ADAPTIVE PROCESS MODEL MATCHING – IMPROVING THE EFFECTIVENESS OF LABEL-BASED MATCHING THROUGH AUTOMATED CONFIGURATION AND EXPERT FEEDBACK

By

Christopher Klinkmüller

Diplom-Wirtschaftsinformatiker (Diploma in Information Systems)
Leipzig University

A THESIS SUBMITTED TO MACQUARIE UNIVERSITY

FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

DEPARTMENT OF COMPUTING

JULY 2017

MACQUARIE
University
SYDNEY·AUSTRALIA

© Christopher Klinkmüller, 2017.

Typeset in LaTeX 2_ε.

Declaration

I certify that the work in this thesis entitled ADAPTIVE PROCESS MODEL MATCHING – IMPROVING THE EFFECTIVENESS OF LABEL-BASED MATCHING THROUGH AUTOMATED CONFIGURATION AND EXPERT FEEDBACK has not previously been submitted for a degree nor has it been submitted as part of the requirements for a degree to any other university or institution other than Macquarie University. I also certify that the thesis is an original piece of research and it has been written by me. Any help and assistance that I have received in my research work and the preparation of the thesis itself have been appropriately acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

Christopher Klinkmüller

Acknowledgements

This thesis is the result of a long journey. In all the years I have benefited from many people and owe them my gratitude. At first, I would like to mention André Ludwig and Bogdan Franczyk who gave me the opportunity to start this project as well as the time, the resources, and the freedom to finish it. I am deeply grateful to them. I am also glad that I was part of their team at Leipzig University. In this respect, I would like to give special thanks to Christoph Augenstein, Steffi Donath, Michael Glöckner, Henrik Kerkhoff, Robert Kunkel, Stefan Mutke, and Martin Roth who have been great colleagues and friends. I also would like to offer my special thanks to Leszek Maciaszek and Mehmet Orgun who made my time at Macquarie University possible. Moreover, they always patiently answered my questions and provided valuable advice.

I would like to express the deepest appreciation to Ingo Weber who I am in great debt to. Without his guidance and persistent help this dissertation would not have been possible. In this context, I also want to thank Florian Daniel, Henrik Leopold, Jan Mendling, and Carlos Rodriguez. Discussions with them have been illuminating and provided inspiration as well as motivation. Additionally, I would like to express my gratitude to Patrick Sievert, Harald Winter, and Anja Michelchen from the AOK Bundesverband as well as to Stefan Zehr, Robert Rieckhoff, and Jochen Riechert from Versicherungsforen Leipzig. Due to their support I had the chance to discuss practical challenges and to transfer my results to practice.

I would like to express my heartfelt appreciation to Julia Bohne and Richard Müller. Their advice and comments have been a great help for the preparation of this thesis. In

this regard, I would also like to show my gratitude to Jan Schreiter and Anrdé Müller with whom I collaboratively created the evaluation datasets.

Finally, I owe a very important debt to my family and Nazrina. They always support me and are there whenever I need them. Even though I do not say it often, their help means more to me than they can imagine. I will always be grateful to have them. Furthermore, I want to thank my friends who had to listen to my complaints for years, but always had an open ear and advice for me. Their support is invaluable to me.

Abstract

Process model matchers automate the detection of activities that represent similar functionality in different models. Thus, they provide support for various tasks related to the management of business processes including model collection management and process design. Yet, prior research primarily demonstrated the matchers' effectiveness, i.e., the accuracy and the completeness of the results. In this context, all data is used for the matcher development and the validity of the design decisions is not studied. A result of these shortcomings is that existing matchers yield a varying and typically low effectiveness when applied to different datasets.

With that in mind, the thesis studies the effectiveness of matchers by separating development from evaluation data and by empirically analyzing the validity and the limitations of design decisions. In more detail, the thesis develops matching techniques that rely on different sources of information. First, the activity labels are considered as natural-language descriptions and the *Bag-of-Words Technique* is introduced. In comparison to the state of the art it achieves a high effectiveness. However, its effectiveness depends on the degree to which the underlying knowledge sources reflect the domain characteristics of the models. Moreover, it needs to be configured for each model collection which can require a huge manual effort. Second, the *Order Preserving Bag-of-Words Technique* analyzes control flow dependencies between activities in order to automatically configure the Bag-of-Words Technique and to maximize its effectiveness. Thus, it relieves experts from manually configuring the matchers. Third, expert feedback is used to adapt the matchers to the domain characteristics of process model collections and to further improve

the effectiveness. Here, the *Adaptive Bag-of-Words Technique* is introduced. It analyzes expert feedback in order to continuously adjust the matching process and yields a strongly improved effectiveness. Consequently, it outperforms state-of-the-art matchers as well as the other matchers from this thesis.

List of Publications

Publications

The thesis has resulted in the following publications. My contribution to those publications and submissions is 75%. If applicable, CORE[1] and ERA[2] rankings are provided.

- C. Klinkmüller, S. Mutke, A. Ludwig, and Bogdan Franczyk. Towards automated logistics service comparison - decision support for logistics network management. In 14th International Conference on Enterprise Information Systems, pp. 259–264 (Wroclaw, Poland, 2012). (Core 2014: C)

- C. Klinkmüller, I. Weber, J. Mendling, H. Leopold, and A. Ludwig. Increasing recall of process model matching by improved activity label matching. In 11th International Conference on Business Process Management, pp. 211–218 (2013). (Core 2014: A)

- C. Klinkmüller, H. Leopold, I. Weber, J. Mendling, and A. Ludwig. Listen to me: Improving process model matching through user feedback. In 10th International Conference on Business Process Management, pp. 84–100 (Eindhoven, The Netherlands, 2014). (Core 2014: A)

- C. Klinkmüller and I. Weber. Analyzing control flow information to improve the effectiveness of process model matching techniques, Decision Support Systems, 2017, ISSN 0167-9236, http://dx.doi.org/10.1016/j.dss.2017.06.002. (ERA 2010: A*)

[1] http://www.core.edu.au/conference-portal, accessed: 13/01/2017
[2] http://portal.core.edu.au/jnl-ranks/, accessed: 13/01/2017

Collaboration Papers

I participated in collaborations that resulted in a professional article and a conference paper. My contribution to the article is 50% and to the conference paper 25%.

- S. Zehr and C. Klinkmüller. Prozesskonsolidierung – ein automatisierbares vorgehensmodell. (English: Process consolidation – an automatable procedure.) Versicherungsforen – Themendossier 14(21), 6 (2014).

- C. Rodríguez, C. Klinkmüller, I. Weber, F. Daniel, and F. Casati. Activity matching with human intelligence. In BPM Forum at the 14th International Conference on Business Process Management (Rio de Janeiro, Brazil, 2016). (Core 2014: A)

Process Model Matching Contest Papers

Finally, I submitted matching techniques to the process model matching contests in 2013 and 2015. The participation in the contests required the preparation of matching results yielded by as well as of a description of the matching techniques. My contribution to the preparation of the techniques was 80%, whereas the overall contribution to the contest papers cannot be properly quantified due to the large number of participants.

- U. Cayoglu, R. Dijkman, M. Dumas, P. Fettke, L. García-Bañuelos, P. Hake, C. Klinkmüller, H. Leopold, A. Ludwig, P. Loos, J. Mendling, A. Oberweis, A. Schoknecht, E. Sheetrit, T. Thaler, M. Ullrich, I. Weber, and M. Weidlich. The process model matching contest 2013. In 3rd International Workshop on Process Model Collections: Management and Reuse, pp. 442–463 (Beijing, China, 2013).

- G. Antunes, M. Bakhshandeh, J. Borbinha, J. Cardoso, S. Dadashnia, C. Di Francescomarino, M. Dragoni, P. Fettke, A. Gal, C. Ghidini, P. Hake, A. Khiat, C. Klinkmüller, E. Kuss, H. Leopold, P. Loos, C. Meilicke, T. Niesen, C. Pesquita, T. Peus, A. Schoknecht, E. Sheetrit, A. Sonntag, H. Stuckenschmidt, T. Thaler, I. Weber, and M. Weidlich. The process model matching contest 2015. In 6th International Workshop on Enterprise Modelling and Information Systems Architectures, pp. 127–155 (Innsbruck, Austria, 2015).

Contents

Declaration iii

Acknowledgements v

Abstract vii

List of Publications ix

List of Figures xv

List of Tables xix

List of Algorithms xxi

1 Introducing the Subject 1

1.1 Motivation 1

1.2 Research Hypotheses 6

1.3 Research Methodology 8

1.4 Solution Details 20

1.5 Structure 22

2 Modeling Business Processes 25

2.1 Business Process Management 26

2.2 Business Process Modeling Techniques 32

2.3 Business Process Modeling Languages 39
2.3.1 The Canonical Business Process Model 41
2.3.2 Business Process Model and Notation 43
2.3.3 Event Driven Process Chain 46
2.3.4 Petri Net 49
2.3.5 Other Notations 51
2.4 Summary 53

3 Matching Business Process Models 55
3.1 Basic Concepts 56
3.2 Application Scenarios 65
3.3 State of the Art 68
3.3.1 Questions 68
3.3.2 Search Strategy 72
3.3.3 Matching Techniques 75
3.3.4 Results 84
3.4 Model Collections 88
3.5 Summary 92

4 Comparing Activity Labels 95
4.1 Basic Label Matching 96
4.2 Label Decomposition 103
4.3 Semantic Comparison of Words 113
4.4 Label Specificity 120
4.5 The Bag-of-Words Matching Technique 128
4.6 Evaluation and Analysis 134
4.6.1 Effectiveness on the Development Datasets 135
4.6.2 Effectiveness on the Evaluation Datasets 137
4.6.3 Semi-manual Configuration 139
4.6.4 Challenge Analysis 140
4.7 Summary 145

5 Analyzing Structure and Behavior **149**

5.1 Multi-Dimensional Classification of Activity Pairs 150

5.1.1 Path Properties 153

5.1.2 Fragment Properties 157

5.1.3 Execution Semantics Properties 161

5.1.4 Suitability Analysis 167

5.2 Patterns for Activity Cluster Detection 171

5.3 Alignment Consistency 178

5.4 The Order Preserving Bag-of-Words-Technique 182

5.5 Evaluation and Analysis 187

5.5.1 Effectiveness on the Development Datasets 187

5.5.2 Effectiveness on the Evaluation Datasets 189

5.5.3 General Validity of the Order Relation Score 190

5.5.4 Portability to Matcher Selection 191

5.6 Summary 192

6 Learning From Expert Feedback **197**

6.1 The Process of Feedback Collection 199

6.2 Word Similarity Adaptation 204

6.3 Transitivity 213

6.4 The Adaptive Bag-of-Words Technique 219

6.5 Evaluation and Analysis 223

6.5.1 Effectiveness on the Development Datasets 224

6.5.2 Effectiveness on the Evaluation Datasets 227

6.5.3 Maximization of the Effectiveness Improvements 228

6.5.4 Reduction of Expert Workload 231

6.5.5 Transitivity in the Evaluation Datasets 234

6.5.6 Limitations of the Feedback Analysis 234

6.6 Summary 236

7 Discussing the Results **239**

7.1 Summary of the Contributions 239

7.2 Threats to Validity 244

7.3 Future Research 246

A Identified Literature **249**

Acronyms **253**

Symbols **257**

Bibliography **261**

References **261**

List of Figures

1.1 The ISR framework 9
1.2 Research design 10
1.3 Literature review process 12
1.4 Literature search process 13
1.5 Qualitative analysis process 16
1.6 Classification of activity pairs with regard to a gold standard 17
1.7 The matching techniques and their dependencies 20

2.1 Levels of business processes 27
2.2 The BPM lifecycle 30
2.3 The information modeling process 34
2.4 Elements of modeling techniques 35
2.5 Elements of modeling languages 39
2.6 Components of the semantics of business process models 40
2.7 BPMN model for the university admission example 43
2.8 Basic BPMN elements 44
2.9 EPC model for the university admission example 47
2.10 EPC elements 47
2.11 Petri net model for the university admission example 49

3.1 An alignment between two university admission process models 57
3.2 General business process model matching workflow 60

3.3 Basic matching sub-workflows 61
3.4 Overview of the search process and the identified papers 75

4.1 Two configurations of the basic label matching algorithm 98
4.2 The feature model for the basic label matching algorithm 101
4.3 The feature model for the bag-of-words matching algorithm 111
4.4 The extended feature model for the bag-of-words matching algorithm . . 118
4.5 Distribution of the label length 121
4.6 The feature model for the bag-of-words matching algorithm with pruning 127
4.7 ROC curves on Birth Registration (BR) (top) and University Admission (UA) (bottom) 132
4.8 PR curves on BR (top) and UA (bottom) 133
4.9 The feature model for the Bag-of-Words Technique 134

5.1 One-dimensional (top) vs. two-dimensional (bottom) classification of activity pairs 151
5.2 Graph structure of the university admission models 154
5.3 RPSTs for the university admission models 159
5.4 Box plots for corresponding (c) and non-corresponding (n) activity pairs . 170
5.5 Examples of the polygon (left) and the sequence pattern (right) 172
5.6 Examples of the path pattern 173
5.7 Examples of the bond (upper left corner) and the partial bond pattern (rest) 174
5.8 Examples of the fragment sequence pattern 175
5.9 Examples of the arbitrarily connected sub-graph pattern 175
5.10 Example of the disconnected sub-graph pattern 176
5.11 Scatter plots for the order relation scores vs the micro f-measure 182
5.12 The OPBOT match workflow 184
5.13 The feature model for the reduced space of BOT configurations 186

6.1 The process of feedback collection 203
6.2 Overview of the effectiveness for the thresholds and word similarities . . . 211

6.3 Average f-measure yielded per model pair with and without the adaptation 213
6.4 Example for transitive correspondences 214
6.5 Example for transitive elementary and complex correspondences 217
6.6 Potentially transitive activity triplets for the example 217
6.7 Possible matching scenarios in the example 218
6.8 The ADBOT workflow 220
6.9 The preparation sub-workflow 221
6.10 The determination sub-workflow 222
6.11 The analysis sub-workflow 223
6.12 Micro f-measures for stopping to collect feedback after a certain iteration 233

List of Tables

3.1 Alignment matrix for the university admission example 57
3.2 Summarized assessment of the approaches from prior research 85
3.3 Descriptive statistics for the process model collections 90
3.4 Descriptive statistics for the gold standards 91
3.5 Results of the matching contests 2013 and 2015 92

4.1 Effectiveness of the basic matching algorithm 102
4.2 Relative frequencies of the activity labeling styles 104
4.3 Frequencies of object and additional information fragments in regular labels 105
4.4 String similarity scores for “accept application” and a second label 106
4.5 Illustration of the bag-of-words similarity using LCS as the word similarity 110
4.6 Bag-of-words similarities for “accept application” and a second label . . . 110
4.7 Effectiveness of the bag-of-words matching algorithm 112
4.8 Effectiveness of the semantic word similarities 119
4.9 Effectiveness of the bag-of-words matching algorithm with pruning 128
4.10 Effectiveness of the optimized BOT configurations on BR and UA 135
4.11 Effectiveness of BOT and the contest matchers on BR and UA 137
4.12 Effectiveness of BOT and the contest matcher on SR and AW 138
4.13 Results of the semi-manual configuration approach 140
4.14 Overview of the false positive challenges 141
4.15 Overview of the false negative challenges 143

5.1 Path position properties for the university admission example 156
5.2 Path neighborhood properties for the university admission example 157
5.3 Fragment properties for the university admission example 162
5.4 Possible execution traces of the university admission process models . . . 163
5.5 Behavioral profiles for the university admission process models 166
5.6 Execution semantics properties for the university admission example . . . 167
5.7 p-values of the Kolmogorov–Smirnov test for BR and UA 168
5.8 Information gain for the selected attributes on BR and UA 170
5.9 Absolute (abs), relative (rel), and cumulative (cul) frequencies of the patterns 177
5.10 Order relation scores of the gold standards on BR and UA 181
5.11 Correlation coefficients on BR and UA 181
5.12 Effectiveness of OPBOT, BOT, and the contest matchers on BR and UA . . 188
5.13 Effectiveness of OPBOT, BOT, and the contest matcher on SR and AW . . 189

6.1 Conceptual overview of design options for feedback collection tasks 201
6.2 Maximum effectiveness of BOT configurations with and without adaptation 209
6.3 Improvements of the micro f-measure 210
6.4 Results of the global clustering coefficient analysis on BR and UA 216
6.5 The local clustering coefficients on BR and UA 219
6.6 Effectiveness of ADBOT and other matchers on BR 225
6.7 Effectiveness of ADBOT and other matchers on UA 226
6.8 Effectiveness of ADBOT and other matchers on SR and AW 228
6.9 Comparison of strategies for the ordering of model pairs 231
6.10 Model collection characteristics vs. improvements gained by analyzing feedback 235

A.1 References identified during the literature search with topic classification and first source of occurrence (part I) 250
A.2 References identified during the literature search with topic classification and first source of occurrence (part II) 251

List of Algorithms

4.1 Basic label matching algorithm 97
4.2 Bag-of-words matching algorithm 108
4.3 Bag-of-words matching algorithm with pruning 125
4.4 Bag-of-words matching algorithm with pruning and filtering 129

6.1 Word similarity adaptation algorithm 206

1

Introducing the Subject

This chapter familiarizes the reader with the topic of this thesis and its underlying research approach. It first introduces the central subject in Section 1.1. In this regard, Section 1.2 provides a more detailed view on the specific research problem. That is, the research hypothesis is introduced and particularized in terms of sub-hypotheses. The approach that was followed to verify these hypotheses is subsequently described in Section 1.3. Following, a summary of the main research contributions is provided in Section 1.4. Finally, the structure of this thesis is outlined in Section 1.5.

1.1 Motivation

Over the last decades business processes have increasingly been recognized as an important element of organizations. In fact, business processes have always existed in

organizations, but were not always perceived as a valuable element. However, with the advent of Information Systems (IS) at the beginning of the 1980's more and more organizations started to automate their processes and to become aware of the importance of their business processes. Organizations recognized that optimizing and automating business processes opens opportunities to increase the efficiency and the effectiveness of businesses. Moreover, they saw the potential to provide services distinguishable from those of competitors by innovating their business processes. To exploit these advantages, organizations conducted large Business Process Reengineering (BPR) projects [1] to optimize their whole business process landscape at once. Such projects were complex, long-running and cost intensive because all processes needed to be analyzed, re-designed and adapted. Here, the analysis phase was typically carried out at the beginning of the projects and the fact that customer requirements and conditions of the market kept evolving during the project was ignored. The result was that many of the re-designed business processes were already outdated at the end of the projects. In order to tackle this problem, a more flexible idea evolved at the beginning of the 2000's. That is, modern Business Process Management (BPM) pursues the continuous analysis and adoption of business processes in focused projects [2]. The benefits of such an ongoing improvement have been recognized by many companies, most notably large and successful organizations, such as those on the Fortune-500 list [3], and BPM has been increasingly adapted.

The basic building block of BPM are business process models as restricted representations of business processes and their environments. Such models serve a multitude of purposes and provide the basis for an extensive number of business related management activities [4, 5]. Respective examples are given in the following. Business process models are used to document and to communicate business processes, e.g., to inform new employees about working procedures that are in place. Furthermore, the automation of business processes through IS can be supported in various ways through models. During requirements analysis those models are employed to capture the demanded workflow that a software system needs to implement. In the development phase these models are iteratively refined and adapted to a specific technical environment. In modern BPM systems technical business process models, usually referred to as workflow models, are

automatically interpreted and executed without the need for manual implementation. Business process models are also valuable for business analysis as they constitute the starting point to identify inefficient activities or steps. When trying to erase identified deficiencies models are often used to evaluate various alternative solutions through simulation. As a consequence of the broad variety of application scenarios model collections within organizations might grow to a size of thousands of models, e.g., the China railway company has more than 200,000 process models [6] and SAP's best practice business process collection exceeds 5,500 models [7].

Another implication of the broad range of usage scenarios is that the same process or sub-process is captured in different models. As models serve different purposes, they comprise different information and focus on different aspects of the same process. The models can focus on control flow aspects including the structure and behavior or involve quantitative metrics that provide information on execution times, costs, or error rates. Furthermore, several models might represent different angles and different levels of abstraction of the same process, e.g., when a process is described from a business and from a technical point of view. Additionally, there might exist variants of the same process that are captured in separate models, e.g., insurance organizations follow the same basic procedure to verify customer claims. However, some checks within this procedure depend on the specific insurance product. Thus, the organizations maintain separate models for each product class and these models are typically characterized by a huge share of identical activities. A result of the fragmented description of processes is the existence of correspondences between models. That is, the same or similar activities occur in various business processes. In this regard, Akkiraju and Ivan [7] report that about 20% of SAP's best practice processes share 50% of their activities with other business processes.

As the creation of business process models is usually a collaborative effort that involves various experts [8–11] these correspondences can be hidden and hard to detect. The reason is that experts have different understandings of the same business process and express their understandings in different ways. Thus, the same fact can be heterogeneously represented in different models [12, 13]. On the one hand, different labeling styles and vocabularies might be used to describe the same activity. On the other hand, different

levels of abstraction might be used or different process layouts can be chosen to express the same behavior. Consequently, correspondences between models cannot always be detected by identifying elements with equal labels. Instead, activities can have heterogeneous labels or are described by a different number of model elements.

In combination with the potentially huge number of models, the model heterogeneity leads to situations where correspondences get indistinct. In this regard, experts expressed their concerns in conversations with the author and stated that they "*drown in their own processes*" and "*need to gain control over their processes again*". The resulting opacity of the process landscape poses a threat to the success of BPM because being unaware of such correspondences decreases the usefulness of the models and aggravates BPM related tasks. The following examples illustrate how knowledge about correspondences can ease BPM related tasks.

First, to prevent business process models from becoming outdated consistency between them must be ensured. In this regard, having a list of correspondences between models helps to transfer updates from the changed model to the related models. For example, when the layout of a process is changed and the according documentation is updated, the changes should also be made in the respective simulation model as the new structure might impact the forecast of execution times, failures, and costs. Second, when a new model is introduced the modeler should be pointed to existing models or parts of them that contain steps similar to the ones introduced in the new model. This way, the reuse of models can be enforced and consistency can be ensured from the beginning [14, 15]. Third, in optimization projects new layouts of a business process are examined to improve the performance of the process. However, as processes are interrelated, changing the layout of the process might impact other processes. Here, correspondences help to determine the influence of a change on the entire business process landscape. Fourth, when updating a technical implementation of a business process the constraints posed by a process model that captures the organizational view should still be satisfied. Again, understanding the correspondences between the technical and the organizational models constitutes a first step in checking whether the technical model is still compliant to the organizational model [16]. Lastly, in business process consolidation projects it is a central task to identify the

most common activities occurring in a set of processes [17, 18]. Such common activities can easily be derived from frequent correspondences.

Although, understanding correspondences between business process models is a key factor in many BPM related tasks, they usually are not explicitly recorded within process model collections. A reason is that collections are often decentralized and it is left to the modelers and departments to maintain their own collections. Furthermore, modeling environments like Signavio[3] or ARIS[4] do not provide sufficient support to maintain correspondences. They only enable the reuse of equally labeled process model elements. However, this requires all experts to use the same modeling environment and to represent models homogeneously. As explained above, this is not always the case.

To ease the experts' jobs and allow them to focus on their actual task, *business process model matching techniques* aim to assist experts by automatically detecting correspondences. The development of such techniques is confronted with the same challenge that experts face: identifying a small portion of correspondences out of a huge number of possible combinations by making sense out of rather restricted and heterogeneous descriptions of business processes. Accordingly, comparative evaluations revealed that state-of-the-art approaches yield a low quality [19, 20], i.e., they detect a small share of the existing correspondences and additionally suggest many non-existing correspondences. Hence, the applicability of the approaches is often limited to model collections with a huge share of correspondences between equally labeled activities.

To this effect, this thesis examines the automated identification of correspondences in collections of heterogeneously modeled business processes. In particular, the thesis focuses on the effectiveness of matching techniques. Here, effectiveness refers to the quality of the results proposed by matching techniques, i.e., the share of correctly identified, falsely suggested and missed correspondences. A matching technique with a high effectiveness reliably suggests correspondences, because it finds many of the truly existing correspondences and only proposes a few correspondences that do not exist. In this

[3] http://www.signavio.com/de/, accessed: 13/01/2017

[4] http://www.softwareag.com/de/products/aris_alfabet/default.asp, accessed: 13/01/2017

context, three sources of information for process model matching are considered for the design of effective matching techniques. First, there are the textual descriptions of activities that encode the purposes of activities. Second, dependencies between activities captured through structural and behavioral relations within process models are examined. Third, the expert feedback in terms of corrections made to automatically suggested correspondences constitutes another source of information. The specific research hypotheses are introduced in the following section.

1.2 Research Hypotheses

The main research objective is to maximize the effectiveness of business process model matching techniques in order to assist experts' in the manual identification of correspondences between process models. This objective is concretized in the following hypothesis which is verified in this thesis.

H0 The adaptation of business process model matching techniques to model collections is necessary to ensure a high effectiveness and the analysis of the control flow as well as of expert feedback provides means to implement this adaptation.

According to this hypothesis, the effectiveness of business process model matching techniques is the primary attribute examined in this thesis. Effectiveness refers to the quality of correspondences proposed by a technique and characterizes the accuracy and completeness of these correspondences. Moreover, the hypothesis states that a high effectiveness which is desirable for practical application requires the adaptation of business process model matching techniques to the characteristics of model collections. This implies that it is not sufficient to rely on universal rules that exploit the textual descriptions encoded in the models. Instead, the hypothesis suggests that the control flow and expert feedback can be exploited to automate the adaptation. In this context, expert feedback is viewed as the manual validation of the suggestions made by a matching technique. That is, experts have to decide whether the classifications proposed by matching techniques hold or not. The main hypothesis is supported by the following sub-hypotheses.

H1 The identification of correspondences between business process models is a challenge for organizations which is not sufficiently supported by existing approaches.

H2 Label-based matching techniques yield a varying and generally insufficient effectiveness.

H3 The maximization of the effectiveness of label-based matching techniques is enabled by the analysis of control flow information.

H4 The effectiveness of matching techniques is improved by the utilization of expert feedback.

Sub-hypothesis H1 emphasizes the practical and scientific relevance of the problem. It views business process model matching as a problem which organizations face in a variety of situations and that requires an enormous manual effort. In this context, automatic decision support in terms of a matching technique bears the potential to minimize the manual effort and to ease the identification of correspondences for experts. Yet, the applicability of existing approaches is limited. Next, sub-hypothesis H2 deals with the textual information in business process models which encodes the purpose of the activities. In order to interpret this information correctly, relations between terms used in the labels must be evaluated. However, universal representations of such term relations are inadequate for an effective matching technique and domain-specific representations are usually not available as they are expensive to create. The control flow information present in business process models is addressed in sub-hypothesis H3. Like the textual descriptions this information is essential for understanding business processes, because it describes the temporal dependencies between the activities. As shown in this thesis, in the context of process model matching control flow information allows for estimating, if a set of proposed correspondences is likely to contain many truly existing correspondences without having knowledge about the truly existing correspondences. This way, it permits the evaluation of the effectiveness of label-based matching techniques in the absence of known correspondences and can be used to automatically configure these techniques in order to maximize their effectiveness. Finally, sub-hypothesis H4 is concerned with the

analysis of feedback provided by experts. Such an analysis allows for deriving domain-specific knowledge that can be used to improve the effectiveness of matching techniques and to achieve practical applicability.

By verifying each of the sub-hypotheses evidence towards the main hypothesis H0 is given. How the evaluation of these sub-hypotheses was carried out methodologically is described in the following section.

1.3 Research Methodology

Business process models play an important role in the design, implementation, and operation of IS and business process model matching techniques support a variety of tasks linked to the management of business processes. Thus, developing business process model matching techniques with a high effectiveness can be classified as design-oriented Information Systems Research (ISR) which goal is "*[...] to develop and provide instructions for action [...] that allow the design and operation of IS and innovative concepts within IS [...]*" [21, p. 2].

Consequently, the methodology underlying this thesis is based on the ISR framework proposed by Hevner et al. [22]. This framework combines *behavioral science* and *design science*. Behavioral science "*[...] seeks to develop and justify theories (i.e., principles and laws) that explain or predict organizational and human phenomena surrounding the analysis, design, implementation, management, and use of information systems.*" [22, p. 76]. In contrast, design science "*[...] seeks to create innovations that define the ideas, practices, technical capabilities, and products through which the analysis, design, implementation, management, and use of information systems can be effectively and efficiently accomplished [...]*" [22, p. 76]. Hevner et al. [22] argue that these two approaches do not exclude but complement each other. While behavioral science aims at revealing the truth, design science puts emphasis on the utility of the designed artifact. Hence, both sciences interact with each other. On the one hand, the design of an artifact relies on the theories discovered within behavioral research. On the other hand, when designing an artifact and focusing on maximizing its utility still unknown truth might be revealed.

Based on this understanding Hevner et al. [22] suggest the ISR framework shown in Figure 1.1. According to this framework ISR is influenced by the environment and the knowledge base. The environment defines the business needs which constitute the requirements that need to be implemented by the designed artifact. These business needs underline the relevance of the research objective and arise from various organizational, human and technical aspects. The knowledge base constitutes the known discovered truth. It contains foundational knowledge that guides the design of the artifact and methodologies that can be applied during the design of the artifact. The environment and the knowledge base establish the frame of ISR.

ISR itself is seen as an iterative approach within this framework. It consists of the *develop/build* and the *justify/evaluate* step. The develop/build step deals with generating artifacts and theories. Whereas in the justify/evaluate step analyses are carried out to back up and assess these artifacts and theories. The ISR process contributes to the environment and the knowledge base. The artifacts are transferred to the environment in order to implement solutions that address the business needs. Additionally, knowledge gained within the ISR process is transferred to the knowledge base and contributes to the scientific state of the art.

Based on the ISR framework the research design outlined in Figure 1.2 was applied. It can be divided into two phases. The first phase consists of the *literature review*. Its purpose

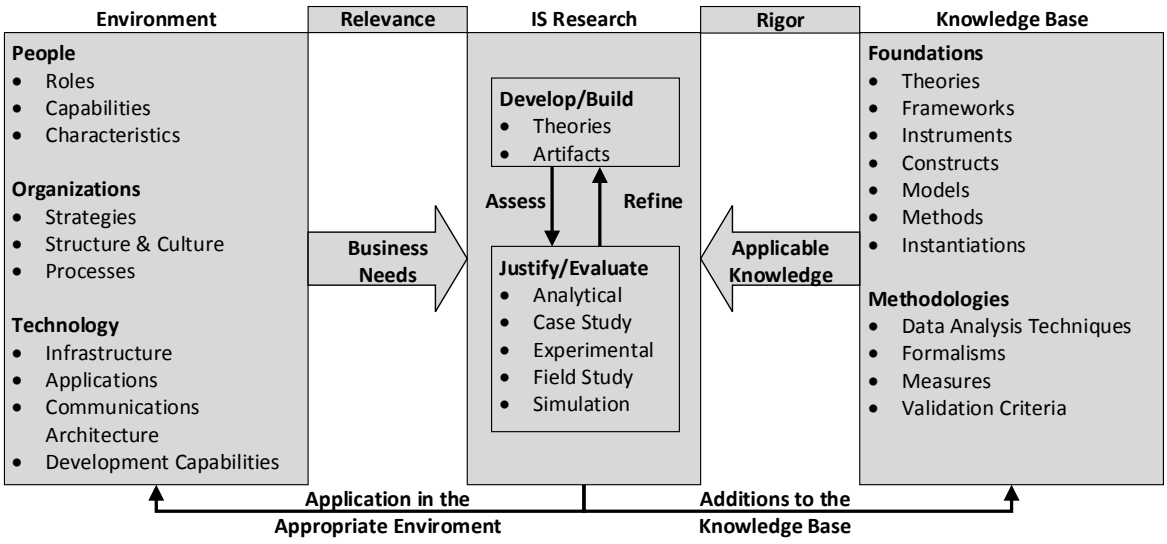

FIGURE 1.1: The ISR framework (cf. [22, p. 80])

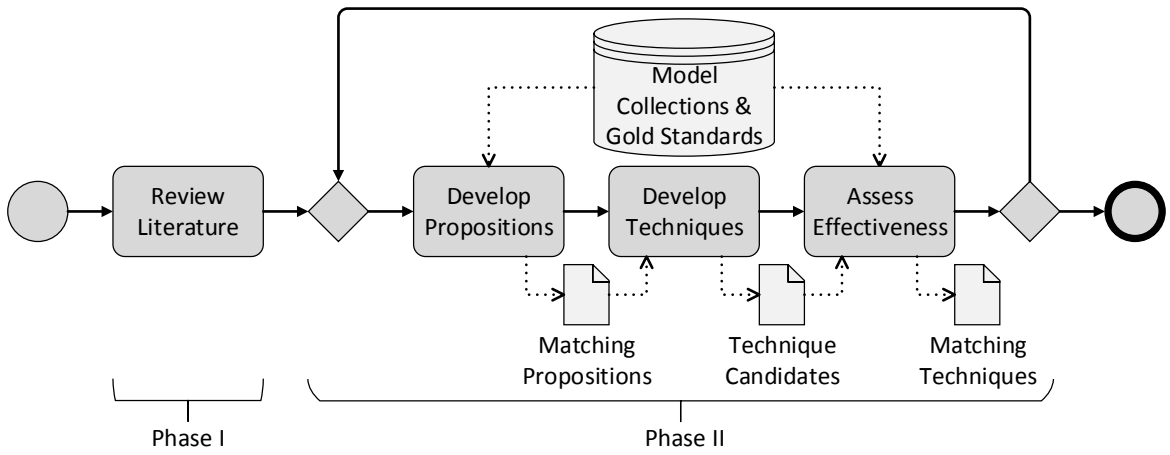

FIGURE 1.2: Research design

was the identification of the research problem and the justification of the scientific as well as the practical relevance. Thus, its results give evidence to sub-hypothesis H1.

In the second phase, the *development of techniques* constitutes the central step. It corresponds to the develop/build step in the ISR framework. Here, *matching technique candidates* were designed. The justify/evaluate step is implemented in two ways. First, in the *effectiveness assessment* matching technique candidates were classified as *matching techniques* or discarded by investigating the degree to which the techniques detect truly existing correspondences. Second, the development was also based on *matching propositions* which are the result of the *development of propositions*. These propositions can be classified as explanation theories [23]. In other words, they provide information on the usefulness of different design options. In accordance with the ISR framework the steps in the second phase are carried out iteratively to develop techniques that justify the sub-hypotheses H2 – H4.

A central component of the second phase are *model collections* and *gold standards*. Whereas a model collection is a set of process model pairs, the according gold standard constitutes the objective truth regarding the correspondences existing in the model collection. More precisely, a gold standard contains correspondences that were identified by experts for each pair of process models in the collection. Together the model collections and gold standards depict the empirical data that was used for two purposes. First, the data was analyzed in order to develop the matching propositions and ground the

design of the matching techniques on empirical evidence. Second, it was used to assess the effectiveness of matching technique candidates. This on the one hand was done to investigate whether the specific utilization of matching propositions within a technique (candidate) yields a good effectiveness. On the other hand, it was carried out to give evidence to the universal applicability and the generalizability of the designed techniques. To this end, assuming a ground truth to exist is a fundamental decision that determines the research methodology as well as the proposed artifacts. While it was argued that different perceptions of whether two elements correspond or not can exist [24, 25], the decision follows the current state of the art in the evaluation of matching techniques [19, 20, 26–30]. Threats to validity arising from this decision are discussed in Section 7.2.

To further substantiate the research design, its most important aspects are discussed in more detail in the following. This comprises the literature review, the proposition development and the effectiveness assessment. The prototype development is not considered here as it deals with formalizing and implementing matching techniques based on the matching proposition that were discovered. It is also guided by the subsequent effectiveness assessment which provides feedback on the design of the matching techniques. Attention is also drawn to the choice of model collections and the definition of gold standards. Note that here the focus is on a basic introduction of methods and concepts relevant to this thesis. Their specific application is outlined in the corresponding chapters and sections throughout the thesis.

Review Literature. A literature review is defined as "*[...] a summary of a subject field that supports the identification of specific research questions.*" [31, p. 31]. Consequently, it is suited as a method to give evidence to sub-hypothesis H1. It helps to review the existing corpus of scientific knowledge and to identify deficiencies of existing approaches. Furthermore, practical application scenarios were derived from the literature in order to underline the practical relevance of the topic.

In this regard, the guidelines suggested by vom Brocke et al. [32] are applied which are the result of an analysis of literature surveys in ISR. They include the review process outlined in Figure 1.3. The iterative layout of the process is affiliated to the continuous updates of the scientific knowledge base due to which reviews become outdated [32].

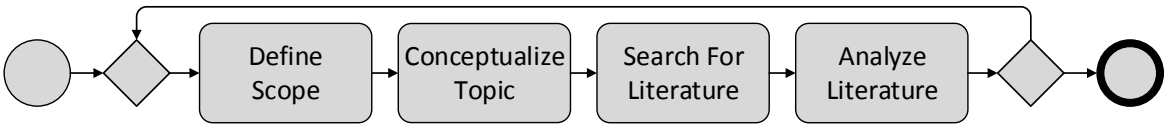

FIGURE 1.3: Literature review process, adopted from [32, p. 2210]

The first step of the review process is the *definition of the review scope*. For this purpose, vom Brocke et al. [32] suggest to follow the taxonomy introduced in [33]. This taxonomy consists of six dimensions. The *focus* refers to the type of artifacts that are examined during the literature review. This includes research outcomes, research methods, theories, and practices or applications. Typical *goals* of a review comprise summary, criticism, or integration of knowledge. The *organization* of a literature review addresses the structure of the review which can be historical, conceptual, or methodological. The *perspective* defines whether the research takes a neutral position or not. The *audience* of a literature review determines the writing style of the author as different audiences require different ways of presenting the research outcomes. The last dimension is the *coverage* which defines to which extent relevant literature is included in the review. The specific configuration of the taxonomy for the purpose of reviewing business process model matching literature is outlined in Chapter 3.

The next step is the *topic conceptualization*. Therefore, known and potentially interesting concepts must be identified and formulated, e.g., by consulting background literature or literature containing a summary of the field of interest. The concepts also serve as input to the literature search as they indicate the relevant issues and can serve as search strings.

Subsequently, the *search for literature* is carried out in order to identify relevant literature with regard to the scope definition. The general search process is depicted in Figure 1.4. It starts with a journal search in order to identify peer-reviewed articles. It can also include proceedings of renowned conferences. Next, appropriate databases are identified to further substantiate the review and a keyword search within these databases is carried out. Finally, to extend the literature review a backward and / or forward search is conducted to identify papers that have been missed so far.

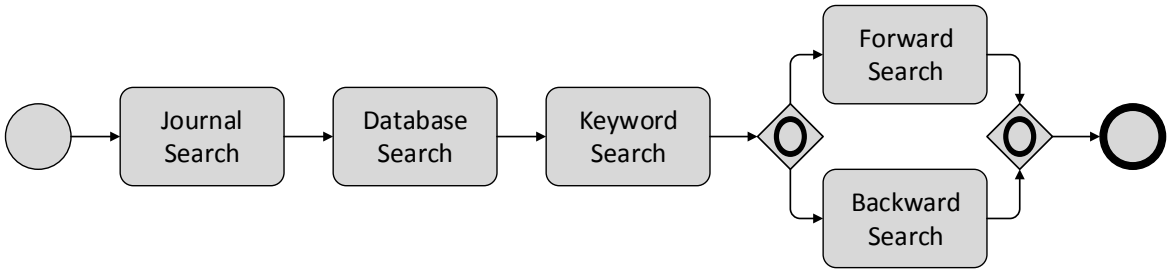

FIGURE 1.4: Literature search process, adopted from [32, p. 2211]

Lastly, the identified papers are examined in the *literature analysis*. This includes the scanning of title, abstract, and content to filter papers that are not relevant. The filtering can already be applied during the search process. Each relevant paper is then assessed with regard to the goal of the literature review. To summarize the results, a concept matrix [34, 35] can be applied. Such a matrix synthesizes the relevant literature and provides the basis for the identification of shortcomings.

Model Collections and Gold Standards. The empirical nature of the research design in this thesis requires data that is used to generate matching propositions and to evaluate the matching techniques. Hence, special attention needs to be drawn to the design of the empirical data collection. Following the classification from [36] the purpose of using the data collection in this thesis is to optimize the matching techniques. Therefore, the data will be used multiple times for analysis and evaluation purposes. In such cases it is recommended to separate training and evaluation data [36, 37]. That is, the training data is used to optimize the techniques while the evaluation data is only used for the final assessment. The idea of this separation is to avoid over-fitting, i.e., to avoid that the techniques perform well on the data, but fail on other data due to a limited generalizability.

With that in mind, four datasets were used in this thesis and separated into *development* and *evaluation* datasets. Note that in this thesis the term development datasets is used instead of training datasets. This is done to avoid confusion as these datasets are not used to train algorithms, but to guide the development of matching techniques. The development datasets comprise two publicly available[5] datasets that were already used for comparative evaluations [19, 20]. Furthermore, there are two evaluation datasets which

[5] http://www.henrikleopold.com/downloads/, accessed: 13/01/2017

are used to finally assess the effectiveness of the proposed techniques and to examine their generalizability. The creation of these datasets was carried out by the author in cooperation with other researchers.

Similar to the evaluation of information retrieval systems [30], a dataset contains a model collection and a gold standard. While the model collection defines the pairs of process models that need to be examined, the gold standard contains the classification of activity pairs contained in the model pairs and thus serves as a baseline for the effectiveness assessment. This classification separates corresponding from non-corresponding activity pairs. The first development dataset contains models dealing with the admission processes at nine different German universities. In particular, they deal with the handling of applications for master courses. The models in the second development dataset are about the registration of newborn children in different countries. The third dataset is the first evaluation dataset and contains models created within the AlmaWeb project at Leipzig University[6]. The project's goal was the unification of processes across all faculties. Finally, the fourth dataset is also solely used for evaluation purposes and consists of selected model pairs from the SAP reference model. This reference model was already subject to scientific analyses [38–40]. A more detailed description of the characteristics of these process model collections can be found in Chapter 3.

While gold standards were already included in the development datasets, they needed to be created for the evaluation datasets. In this regard, two researchers, the author of the thesis and another researcher, manually identified corresponding activities independently. Then, differences were determined automatically, i.e., a software program identified the activity pairs that one of the researchers classified as corresponding and the other one did not. These differences were resolved in a discussion between both experts. Having each activity pair classified by two experts was the result of the limited availability of assessor time. That means, there were only a few experts available which were familiar with the processes or had the time to familiarize themselves with the processes. In such cases, Carterette [41] suggests to carry out a wide and shallow rather than a narrow and deep classification.

[6] http://www.zv.uni-leipzig.de/studium/almaweb.html, accessed: 13/01/2017

Develop Propositions. In this thesis, matching propositions were derived from two sources. First, there is the literature on business process model matching and from related fields including information retrieval as well as schema and ontology matching. As the review of literature was already discussed, the focus is here on the analysis of the empirical data, the second source for the development of propositions.

The development of matching propositions was intended to reveal cause and effect relations that provide reusable explanations for the classification of activity pairs. Therefore, an empirical approach was taken that aimed to derive propositions from the development datasets. In general, there are three types of empirical inquiries: *quantitative*, *qualitative*, and *mixed methods* [42].

"*A quantitative approach is one in which the investigatory primarily uses postpositive claims for developing knowledge [...], employs strategies of inquiry such as experiments and surveys, and collect data on predetermined instruments that yield statistics data.*" [42, p. 18]. In other words, quantitative approaches require the researcher to focus on cause and effect relations that are encoded through variables and theories. These relations are then falsified by carrying out appropriate statistical tests on the data.

While quantitative research focuses on measurements, "*[...] a qualitative approach is one in which the inquirer often makes knowledge claims based primarily on constructivist perspectives [...] or advocacy/participatory perspectives [...] or both [...] [and collects] open-ended, emerging data with the primary intent of developing themes from the data.*" [42, p. 18]. That is, theories are constructed from the data by manually exploring it. Thus, qualitative research is helpful when a deep understanding of a phenomena is needed [23].

In this thesis, a mixed method that utilizes quantitative as well as qualitative approaches was applied. The primary focus was on quantitative analyses of the data to test propositions regarding the identification of correspondences. However, when a more detailed understanding was needed in order to refine propositions or identify patterns, qualitative analyses were carried out.

The quantitative part of the method was carried out by formalizing variables that encoded various properties of activities and activity pairs. Based on these encodings various

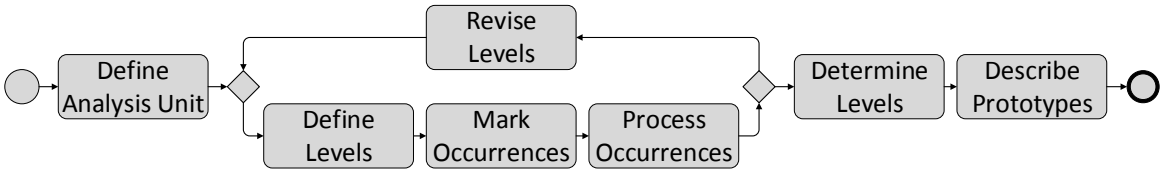

FIGURE 1.5: Qualitative analysis process, adopted from [45, pp. 63]

statistical measurements were applied to check whether these variables are correlated to the classification of activity pairs or not. Amongst others, these measures include the Kolmogorov-Smirnov Test for testing the equality of probability distributions [43] and the information gain for comparing the goodness of classification strategies [44].

The qualitative analysis process applied in the context of this thesis is outlined in Figure 1.5. It is oriented towards the categorizing content analysis [45, 46]. The goal of this method is to examine a specific property and to determine its typical levels. The process starts with the definition of the analysis unit. In this step, the data relevant to the specific property is selected. Next, the data is processed iteratively. First, levels of the property are defined. In this regard, each level is described and concurrent examples are defined. Mayring [45] also recommends to define the rules for marking the occurrences. Following, occurrences of the levels in the data are marked. Then, these occurrences are processed in order to determine the coverage of the levels. In case, not all the data can be classified using the defined levels, the levels are revised. This comprises deleting irrelevant levels as well as adding new levels. Once all data is classified, the typical levels are selected. Here, non-frequent levels are ignored or merged into levels that subsume them. Finally, prototypes for each level are selected in order to provide empirical evidence for their existence.

Assess Effectiveness. The effectiveness assessment constitutes a special quantitative method in this thesis. Its goal is to estimate the quality of the results a matching technique proposes. This is an essential step to select the best matching techniques from those designed in the development step and to examine their generalizability.

To assess the effectiveness three measures well known in information retrieval [30, 47] are applied. These measures are referred to as *precision*, *recall* and *f-measure* [30]. They are also widely adapted in schema matching [28, 29]. Furthermore, these measures are

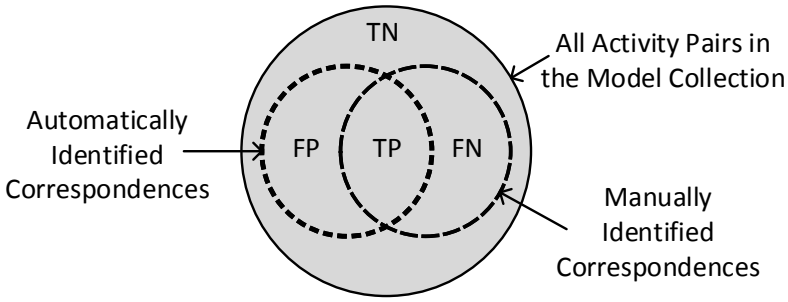

FIGURE 1.6: Classification of activity pairs with regard to a gold standard

proposed by the Ontology Alignment Evaluation Initiative [48] for the assessment of ontology matching algorithms and used in the initiative's annual comparative evaluation, e.g., [26, 27]. Moreover, they were used in comparative evaluations of business process model matching techniques [19, 20].

All three measures rely on the comparison of the classification made by the examined matching technique and the classification suggested by the gold standard as shown in Figure 1.6. A matching technique automatically classifies activity pairs in the collection as corresponding or not. The proposed correspondences are referred to as positives, whereas all other pairs are subsumed as negatives. With regard to the gold standard both sets of activity pairs can be divided into two subsets. The *Set of true positives* (TP) comprises all positives that are truly corresponding with regard to the gold standard, whereas the *Set of false positives* (FP) contains all other positives. Similarly, the negatives are grouped into the *Set of true negatives* (TN) and the *Set of false negatives* (FN). With regard to this classification the three effectiveness measures can be defined.

The precision pr is the share of true positives among all positives. Additionally, the recall re is the ratio of the number of true positives and all activity pairs that truly correspond. Finally, the f-measure F is the harmonic mean of both measures. A mathematical definition of these measures with reference to business process model matching is provided in Chapter 3.

Besides the application of established research methods, the research design was further substantiated by taking the ISR guidelines [22] into account. Adhering to these guidelines guaranteed that internationally accepted research standards in the discipline

were met. In the following, each of the seven guidelines is introduced and it is outlined how these guidelines were implemented in the context of this thesis.

Guideline 1 (Design as an Artifact) demands that ISR must produce viable artifacts which can be constructs, models, methods, and instantiations. The thesis produced three kinds of artifacts. First, the matching propositions constitute constructs that were generated to gain an understanding of cause-effect relations that can be exploited to automatically match process models. Furthermore, the matching techniques proposed in this thesis constitute methods. Lastly, the designed techniques were implemented in a Java library that is an instantiation of the research results.

Guideline 2 (Problem Relevance) addresses the practical relevance of the examined problem. That is, ISR projects have to address problems that organizations face. As already outlined in Section 1.1 business process model matching is a relevant problem in practice. This is illustrated in more detail through the investigation of sub-hypothesis H1. Here, a review of BPM activities in which business process model matching plays a central role provides evidence towards the practical relevance.

Guideline 3 (Design evaluation) expects that the utility, quality, and efficacy of research results is rigorously verified in order for the results to be accepted as artifacts. To this end, the clear separation between development and evaluation data permitted a final assessment of the effectiveness, the important quality criterion of the designed matching techniques. Thus, it provides evidence towards the general applicability of the designed techniques.

Guideline 4 (Research Contribution) refers to the scientific relevance. According to that guideline, the design artifacts must contribute to the relevant research area. In this respect, the research gap was identified through a literature review. In particular, existing approaches were identified and their shortcomings examined to verify sub-hypothesis H1. Furthermore, existing approaches and the designed techniques are compared based on the datasets. This demonstrates that the research results extend and improve the state of the art. Moreover, this thesis explicitly examines cause-effect relations underlying

the matching techniques. In contrast to prior research where matching techniques are introduced as closed entities, this ensures transparency. Thus, it is easier in future work to reuse, build upon, and improve the research results. An overview of the designed techniques is provided in the next section.

Guideline 5 (Research Rigor) postulates that the creation and the evaluation of the research artifacts have to rely on rigorous methods. This guideline was addressed by choosing established research methods widely adopted in ISR to implement the steps of the methodology as outlined in this section.

Guideline 6 (Design as a Search Process) claims that actions and resources have to be iteratively applied to achieve the defined goals under the constraints and laws of the solution space. This guideline is also implemented as the final matching approach is based on continuous analyses and evaluations relying on real world data. On the one hand, analyses were refined and revealed propositions step by step. On the other hand, the concrete application of these propositions within the techniques was guided by evaluations. That way, the techniques could be fine-tuned to maximize their effectiveness.

Guideline 7 (Communication of Research) requires research results to be communicated to scientific as well as technology-oriented and management-oriented audiences. This guideline is not explicitly addressed in the methodology. However, the dissemination was taken care of during the research project. The transfer to the scientific community was ensured through the publication of three conference papers [49–51] and one journal article [52]. Moreover, the author of the thesis contributed to another conference paper [25] and submitted version of the developed matching techniques to two comparative evaluations that were published as workshop papers [19, 20]. Furthermore, the results were successfully disseminated in practice. This was achieved in cooperation with two organizations. First, a tool was developed for and with BPM experts of the AOK Bundesverband GbR. This tool permits a practical application of the developed matching techniques in process consolidation projects. Second, through the support of the Versicherungsforen Leipzig GmbH the research results could be presented to companies from the insurance

domain. In addition to several presentations held for management-oriented audiences, a professional article [53] as well as an interview [54] were published.

1.4 Solution Details

As the sub-hypotheses allude, the designed matching techniques utilize textual and control flow information encoded in business process models as well as expert feedback. In particular, there are three matching techniques which build upon and extend each other as outlined in Figure 1.7. At the center of the techniques there is the Bag-of-Words Technique (BOT). It is the result of the examination of sub-hypothesis H2 and solely evaluates textual information in terms of activity labels in order to detect correspondences. The Order Preserving Bag-of-Words Technique (OPBOT) extends BOT by incorporating control flow information. It originates from the investigations related to sub-hypothesis H3. Finally, both techniques constitute the core of the Adaptive Bag-of-Words Technique (ADBOT) which refers to sub-hypothesis H4. This technique analyzes expert feedback to adapt the matching process and to improve the effectiveness stepwise. Details of these techniques are provided below.

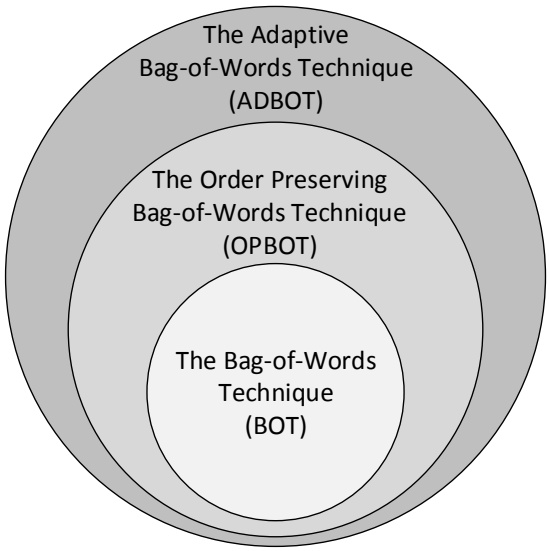

FIGURE 1.7: The matching techniques and their dependencies

The Bag-of-Words Technique (BOT) only considers labels to match business process models. It works by first filtering equally labeled activities and considering them as correspondences. After that, the remaining activity pairs are inspected. Therefore, the labels are decomposed into the individual words, and relations between these words are used to compute a similarity score. If the similarity score indicates that the activities are highly similar, they are considered as correspondences. Basically, BOT is a configurable technique with five features. However, as only a specific feature configuration can be applied by business experts, a default configuration is derived from the evaluation on the development datasets.

The Order Preserving Bag-of-Words Technique (OPBOT) addresses the configuration problem. That is, a BOT configuration which is performing well on one dataset does not necessarily need to yield a high effectiveness on a different dataset. Instead, BOT needs to be optimized on each dataset in order to maximize its effectiveness. This requires knowledge about the truly existing correspondences. However, collecting these correspondences makes the maximization obsolete. To solve this paradox, the *order relation score* is introduced. It is based on structural relations between correspondences and is strongly correlated to the effectiveness of matching techniques. Thus, it can be utilized by OPBOT to predict the effectiveness of BOT configurations without knowing the true correspondences. By this means OPBOT searches the space of BOT configurations in order to detect the most promising configurations. Once they are identified, OPBOT takes their proposals and combines them to a final result.

The Adaptive Bag-of-Words Technique (ADBOT) analyzes feedback provided by experts. In more detail, it determines correspondences for a model pair and presents these model pairs to the experts. Then, the experts are required to correct the suggested correspondences. This means that they remove falsely proposed correspondences and add correspondences that were not detected. This feedback is then used to adapt the matching mechanism and to improve the effectiveness. To this end, the adaptation of BOT configurations is considered. In particular, OPBOT is used to determine the most promising BOT configurations. Then, the feedback is used to adjust the word relations

underlying the BOT configurations to better reflect the characteristics of the domain terminology. Due to this adaptation the effectiveness of the BOT configuration is gradually enhanced. Further improvements are gained by transitively inferring correspondences from other model pairs for which the correspondences are already known. A further part of ADBOT is a strategy to reduce the workload for the experts while maximizing the improvements gained through analyzing the feedback. That is, the model pairs that need to be matched are sorted so that the order in which feedback is collected maximizes the improvements and thus ADBOT's effectiveness. Moreover, it is shown that feedback is only needed for a subset of the model pairs to maximize ADBOT's effectiveness. Thus, the remaining pairs are matched automatically without requiring efforts from the experts.

1.5 Structure

This thesis is organized in seven chapters wich can be divided into three parts. The first part provides the foundations, the second deals with the matching techniques, and the third concludes the thesis. The content of each chapter is briefly outlined in the following.

Chapter 1: Introducing the Subject defines the thesis' scope. It motivates business process model matching as the primary research object and introduces the hypotheses examined in this thesis. In this context, the research methodology applied to verify the hypotheses, the contributions, and the structure of the thesis are outlined.

Chapter 2: Modeling Business Processes narrows the context of this thesis down by discussing BPM and business process modeling. First, a brief overview of BPM is provided in which business process models are the central building block. Next, basic concepts regarding the modeling of business processes are introduced. Finally, a formal definition for business process models is provided and the most widely adopted modeling notations are reviewed with regard to this definition.

Chapter 3: Matching Business Process Models deals with the problem of automatically identifying correspondences between process models. In this regard, business process model matching is formally defined. Next, the use of business process model

matching in BPM is summarized to motivate the need for such techniques in practice. Afterwards, existing literature regarding business process model matching is reviewed in order to substantiate the scientific demand. Then, the empirical data comprising the development and evaluation datasets is introduced. Finally, the findings are summarized and discussed in order to verify sub-hypothesis H1. An initial discussion of the shortcomings of existing approaches was published in [49] and the literature review is part of the appendix of the journal article [52].

Chapter 4: Comparing Activity Labels examines sub-hypothesis H2. More precisely, strategies to exploit the labels of activities are investigated and BOT is introduced. Furthermore, the limitations of label-based matching techniques are discussed. An early version of BOT was published in [50] and was submitted to the process model matching contest 2013 [19].

Chapter 5: Analyzing Structure and Behavior deals with the use of the control flow. Here, properties of activity pairs, patterns of activity clusters, and relations between correspondences are examined. Based on the outcome of these analyses OPBOT is introduced. Altogether, these investigations give evidence towards sub-hypothesis H3. The analysis of the activity properties was published in [51] and a first version of OPBOT was submitted to the process model matching contest 2015 [20]. Moreover, the manuscript that was submitted to Decision Support Systems [52] contains all behavioral analyses regarding the use of control flow information from this chapter as well as the current version of OPBOT.

Chapter 6: Learning From Expert Feedback provides evidence to the last sub-hypothesis H4. The sub-hypothesis is verified through the development of ADBOT. In this regard, adjusting word similarities and transitively inferring correspondences are discussed as strategies to learn from expert feedback. The approach for adjusting the word similarities was published in [51].

Chapter 7: Discussing the Results summarizes the contributions. It also discusses limitations of the thesis and gives directions for future research.

2

Modeling Business Processes

This chapter introduces business process models as the key artifact that matching techniques need to handle and hence helps the reader to comprehend the setting in which process model matching techniques are applied. Following, Section 2.1 gives a general overview on BPM in order to provide an understanding of the context in which business process models are used. Subsequently, Section 2.2 deals with business process modeling techniques which constitute the basis for the creation of business process models. An important aspect in this regard are the modeling languages as they provide means to capture business processes. Thus, Section 2.3 presents a detailed overview of such languages. Moreover, the section introduces a definition of a canonical business process model and relates it to the modeling languages. Relying on this canonical model permits a language-independent definition of the matching techniques and is a prerequisite for a broad applicability. Finally, the chapter is summarized in Section 2.4.

2.1 Business Process Management

Modern organizations execute business processes in order to manufacture products or deliver services to their customers. For example, a university assesses student applications to decide whether a student is qualified to study at the university or not. Therefore, the university generally follows a defined process consisting of various checks. First, the document is formally verified. More precisely, it is investigated if the application was submitted in time and if it comprises all necessary documents. Here, an IS is used to support the verification. Afterwards, the application is assessed by the examination board and the applicant's aptitude is determined. Based on the recommendation the applicant is either accepted or rejected. Finally, the application as well as all documents created during the assessment are archived.

This example illustrates the basic characteristics of business processes as defined by Weske [5]. First of all, they realize business goals. Here, the assessment of an application is related to the general goal of a university to provide higher education. Second, business processes comprise a set of activities that are executed in coordination, e.g., the formal checks are carried out before the application is assessed. Thirdly, business processes are embedded in an organizational and technical environment which amongst others comprises resources, employees, and software. In the above stated example, there are employees of the university's examination office and a special IS. Finally, business processes are always executed within the boundaries of a single organization, but might interact with business processes from other organizations. For example, the task of accepting a student could involve the registration of the student with a public student service. In such a case, the exchange of information about the student's registration constitutes an interaction between the university's and the service's processes. The following definition summarizes these characteristics and depicts the basic understanding adopted in this thesis. Note, that the terms business process and process (in general) are used interchangeably in this thesis.

Definition 2.1 (Business process) *"A business process consists of a set of activities that are performed in coordination in an organizational and technical environment. These activities*

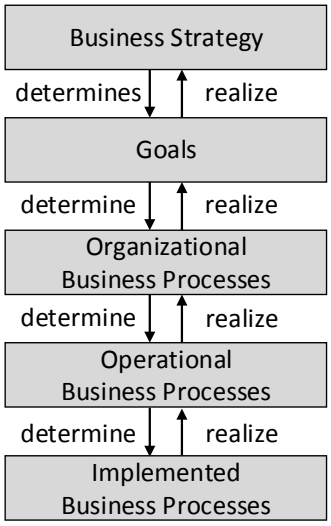

FIGURE 2.1: Levels of business processes, adopted from [5, p. 18]

jointly realize a business goal. Each business process is enacted by a single organization, but it may interact with business processes performed by other organizations." [5, p. 5]

A specific type of business process is called workflow. A workflow automates parts of or the entire business process by relying on defined rules in order to manage the exchange of information between participants [5, 55]. In the example, such workflows might be part of the software responsible for the formal verification of the application or they might be setup to automate the coordination of the involved parties. In this thesis, workflows are not discussed separately. Instead they are subsumed under the term business process.

Typically, business processes exist on various levels with regard to the business goals of an organization. An according classification of such levels proposed by [5] is shown in Figure 2.1. In this classification business processes on higher levels always determine the design of those on lower levels, whereas the business processes on lower levels realize the objectives of the according business processes from higher levels. The top level constitutes the *business strategy*. It comprises the long term goals an organization strives to achieve. With regard to the university example this could be the goal to become one of the most recognized universities. On the second level the strategies are subdivided into *goals* that present a short term perspective. With reference to the example the goal is the provision of higher education. As the provision of education is recognized by the public, it is an important aspect for the implementation of the strategy.

While these two levels deal with objectives an organization aims to achieve, the lower levels focus their implementation in terms of business processes. On the third level there are the *organizational business processes* which represent high level and coarse-grained business functionalities addressing the defined goals. Each organizational business process is refined by a couple of *operational business processes*. Such processes represent a perspective where activities and execution constraints between them are focused. The last level consists of *implemented business processes*. In contrast to organizational business processes these business processes usually contain information specific to the execution of the activities. This comprises policies and guidelines as well as automatically executable pieces of software. The process described in the example constitutes an operational process as it provides a basic overview of the activities that must be carried out during the assessment of an application. This operational process is related to the organizational business process "study management" which again is related to the goal of providing higher education. Instructions for the examination board as well as procedures that are part of the software are implemented business processes.

While all levels are important to the management of businesses, organizational and implemented business processes are not considered in this thesis. The reason is that organizational business processes are too abstract and implemented business processes too specific. More precisely, there are usually only a few organizational business processes within an organization. Thus, identifying correspondences at this level does not require a huge manual effort. Moreover, due to the abstractness of this level, correspondences between organizational business processes do not provide valuable insights for experts. In contrast, implemented business processes are the adaptation of operational business processes to certain IS or working environments and they cover a broad variety of fine-grained steps that are specific to the systems and working environments. As a consequence, there are many small-scale correspondences which are hard to grasp and to manage coherently. Hence, process model matching techniques adresses processes at the operational level in order to support BPM related management activities. For the sake of simplicity and as long as not stated otherwise, the terms business process and process are used in this thesis to refer to operational business processes.

Business processes have a long history. The following summary of the historic development of the concept of business processes is oriented towards the overview provided in [4]. Since the prehistory humans have applied working procedures to build, produce, and create tools, jewelry, buildings, and so on. While at the beginning humans were generalists and able to produce various kinds of goods, they became more and more specialized over time. In the middle ages, this development lead to the establishment of guilds in which craftsmen pursuing a similar profession organized themselves [4]. The specialization of labor was further driven by Taylor [56] who proposed his *principles of scientific management* at the beginning of the 20th century. One of the elements of the scientific management was the precise examination of single production steps and the according development of instructions. As a consequence the functional organization was adopted by most companies and laborers became responsible for single tasks. However, Davenport and Short [57] pointed out that focusing on the optimization of single tasks rather than looking at the entire process potentially causes inefficiencies. As a consequence, Business Process Reengineering (BPR) [1, 58] arose. BPR aimed to apply management concepts in order to restructure the businesses of organizations and to increase the effectiveness and efficiency of their business processes. BPR projects aimed to improve the business process landscape at once, i.e., the whole business was analyzed, redesigned, and changed within a single project. The problem of such large scale projects is that the time span between the beginning of the planning and the end of the implementation is long, usually a few years. Within such a long period market conditions are likely to change and the plans become outdated, so that the improvements are not effective anymore. Thus, a more continuous approach referred to as Business Process Management (BPM) emerged at the end of the 20th century [2]. It was driven by the development of modern IS, like Enterprise Resource Planning (ERP) systems and Workflow Management Systems (WFMS) [4]. While ERP systems allowed to centralize the managment of information, WFMS enabled organizations to automate and flexibly adapt their business processes with regard to changing market conditions. Another driver was the availability of statistical measures that allowed to assess and evaluate business processes as well as to examine alternatives. Based on the early work by Deming [59]

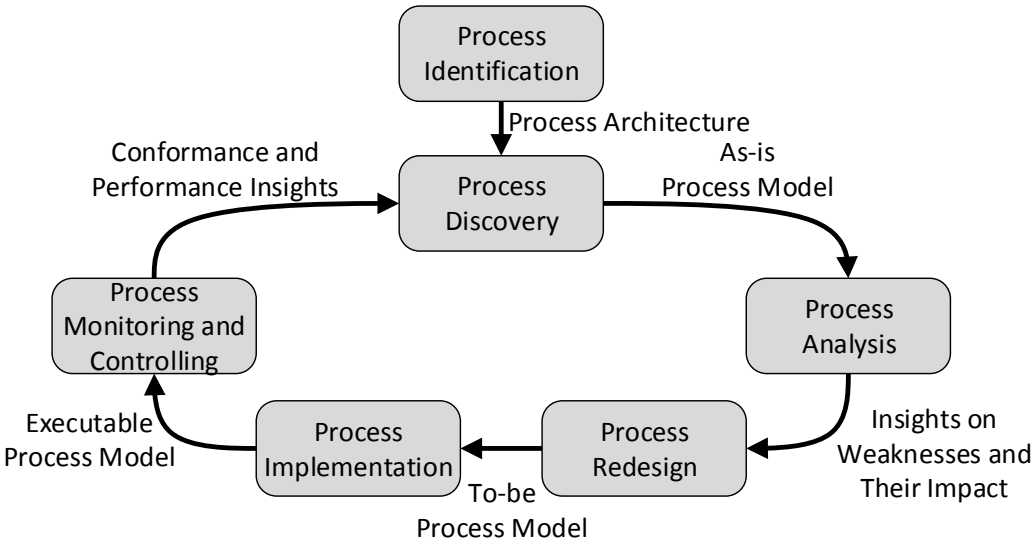

FIGURE 2.2: The BPM lifecycle, adopted from [4, p. 21]

and Shewhart [60] more sophisticated approaches, like six sigma [61], became available. Nowadays, organizations establish BPM in order to continuously improve their business processes. A definition of BPM is presented below.

Definition 2.2 (Business process management) *Business Process Management (BPM) depicts a set of tools to support all activities that aim to continuously improve business processes over their whole lifecycle.*

This definition represents the common understanding of BPM shared by many definitions, e.g., [4, 5, 62]. It basically consists of two ingredients. First, BPM is seen as a toolbox that provides business experts with means to ease their work. Such means typically comprise concepts, methods, techniques, and software. The matching techniques developed in this thesis are part of this toolbox. Second, BPM addresses activities that arise within the lifecycle of business processes. In this regard, various lifecycles have been proposed [5, 62–65]. To provide a more detailed understanding of the activities that occur within the BPM lifecycle, the one introduced by Dumas et al. [4] is taken as a reference here. Its phases and their relations are shown in Figure 2.2.

The iterative design of the lifecycle accounts for a continuous BPM. After the business processes are setup, they are monitored and adopted to react to changing requirements. The lifecycle consists of six phases:

1. *Identification*: In this first phase a business problem is identified and processes needed to solve it are determined. Next, relations between these processes are specified. The outcome of this phase is a process architecture which provides an overview of the organization's process landscape. If a process architecture already exists, it will be updated in this phase.
2. *Discovery*: This phase is about documenting the current state of the processes. Here, as-is models describing this state are created or updated, in case the process is refined.
3. *Analysis*: The examination of the current process implementation is carried out to reveal and record limitations. Here, performance measures are used to quantify the current state of the implementation. The result of this phase is a collection of issues that need to be solved in order to satisfy the business need. Moreover, the analysis phase provides insights into the actual implementation which can be used in the next phase.
4. *Redesign*: Based on the analysis results various alternatives that implement the business needs and solve the issues are developed. In consideration of the analysis results models of alternatives are iteratively evaluated and refined, resulting in a set of to-be models.
5. *Implementation*: After a solution was selected, activities needed to transform the business in order to implement the to-be models are planned and executed. Organizational changes are subject to such activities. Such changes refer to the work of all process participants. They include amongst others the provision of an appropriate working environment and the training of the participants. Moreover, changes to the automation of the processes might be required. This includes the design, creation, and deployment of appropriate IS. Such systems comprise hardware, e.g., computers, machines, robots, as well as software.
6. *Monitoring and controlling*: Finally, performance indicators are measured to assess the quality of the process execution. If there is the need for short term intervention, counter actions are performed in this phase. In cases where fundamental updates to

the process architecture are required a new iteration of the lifecycle will be triggered, beginning with the discovery phase.

These explanations illustrate that business process models are an integral component to BPM. Throughout the lifecycle business process models are used to document, analyze, design, implement, execute, and monitor business processes. As these steps serve different purposes, there are also different information needs in these steps. Consequently, the same process or sub-process is likely to be captured in different models. This fact substantiates the need for matching techniques that help experts to comprehend the relations between the models.

2.2 Business Process Modeling Techniques

Models are an important tool to many professional activities. Architects draw models of buildings that later on are used by construction companies to build it. Dentists create dental impressions to manufacture protheses. Moreover, models are also omnipresent in everyday life. For example, people rely on maps to navigate through cities or children play with toys that are based on real objects.

In the area of computer science and IS respectively, a model is usually conceived as "*[...] a representation of either reality or vision*" [66, p. 162], i.e., it constitutes a *mapping* of an original. According to Stachowiak [67] this mapping property is one of three properties that characterize models. Additionally, a model abstracts from the original as it depicts a subset of the original's attributes. This property is referred to as *reduction*. Moreover, the *pragmatism* property states that a model serves a certain purpose. It is determined by the audience that uses the model, the task supported by the model and the point in time of model creation and usage. Depending on the purpose the model might comprise different attributes of the original.

As outlined in Definition 2.1 a business process consists of a set of activities that are performed in a socio-technical environment. Thus, a model of a business process contains information about such activities, their order of execution and their environment. In more detail, attributes of a business process can be assigned to one of four perspectives

[68, 69]. The *functional* perspective includes attributes regarding the activities that are carried out in a business process. This perspective also comprises the objects that serve as input to or are the output of these activities. The *behavioral* perspective captures the control flow and provides details about the temporal ordering of these activities. Here, structural and behavioral constructs like loops, alternative paths, parallel executions etc. are considered. The characteristics of the objects that are processed by the activities are focused in the *informational* perspective. Finally, the *organizational* perspective deals with people, resources, and roles relevant to the business process.

Schuette and Rotthowe [70] criticize the definition by Stachowiak [67] as it implies that a model is a mapping of the real world. Instead, they emphasize the modeler's role in the process of mapping the original to the model. They comprehend a model as "*[...] the result of a construct done by a modeler who examines the elements of a system for a specific purpose [...] at a given point in time with a specific language [...]*" [70, p. 243]. According to this understanding, a model does not directly represent an original, but is a subjective outcome reflecting the modeler's perception of the original.

Additionally, models are not always created by a single person, but often the creation of the model is a collaborative approach [8–11]. Hence, a model is seen as the subjective outcome of modelers that jointly create the model in this thesis. Based on these positions the following definition of a business process model is introduced which is similar to the definitions in [39, 64] .

Definition 2.3 (Business process model) *A business process model is constructed by one or more modelers and represents their perception of a real-world or fictive business process. It comprises information on the functional, behavioral, informational, and organizational perspectives of a business process that is relevant with regard to a specific purpose.*

Inherent to this definition is that the creation of a model is a process itself. In the domain of IS it is generally referred to as *information modeling* or *conceptual modeling*, respectively. A basic view onto the information modeling process is provided by Frederiks and van der Weide [9] who build their understanding of information modeling on the perception of Burg [71]. According to Frederiks and van der Weide [9] the modeling

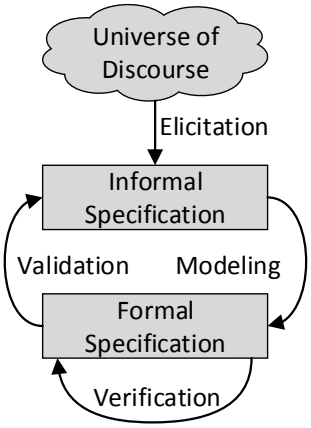

FIGURE 2.3: The information modeling process, adopted from [9, p. 7]

process consists of four phases as shown in Figure 2.3. It was discussed and adopted in the context of business process modeling, e.g., in [4, 72–75].

The first phase is the *elicitation* where the *universe of discourse* is investigated by the modelers. With regard to business process modeling this universe usually contains the business processes that have to be modeled and their environment. As a result of this phase an *informal description* is created. It represents the analysts' understanding of the universe of discourse. The phase can be subdivided into three steps. First, there is the collection of significant information objects that need to be considered in the model. Next, these information objects are verbalized using a natural language. Finally, this specification is reformulated into a unifying format. There is a huge variety of techniques that support this phase including focused observation, case study analysis, questionnaires, or time line analysis [76]. A detailed description on discovering business processes including data collection and the organization of workshops is presented by Sharp and McDermott [77].

In the *modeling* phase the informal specification is transformed into the *formal specification* which represents the model. Therefore, the modelers need to carry out two tasks. First, the modeling concepts needed to express the informal description have to be identified. Second, the informal description must be translated to the model by matching the informal description to the concepts. In this phase, *modeling techniques* provide specific means to solve a certain modeling problem. Such modeling techniques consist of a *modeling language* and a *modeling procedure* [78, 79] as shown in Figure 2.4. While the modeling language defines the concepts that are available to describe the universe

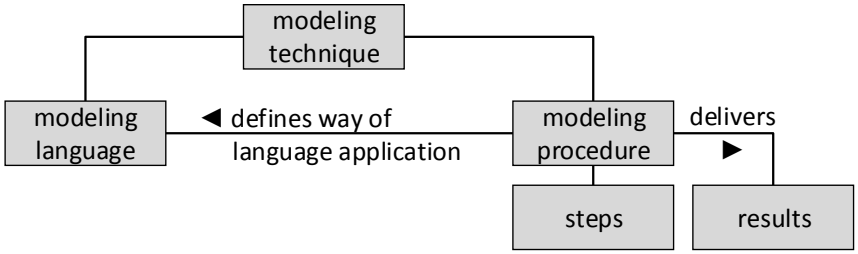

FIGURE 2.4: Elements of modeling techniques, adopted from [78, p. 184]

of discourse, the modeling procedure defines steps that need to be carried out in order to yield the desired result, i.e., the model. A similar view on the elements of a modeling technique is given by the framework for research on conceptual modeling [80].

An overview of modeling languages is provided in the next section. The modeling procedures are usually specific to these modeling languages, but there also exist a variety of language independent guidelines to support modelers. A known set of such guidelines in the context of IS are the guidelines of modeling [70, 81]. These guidelines were discussed in the context of business process modeling by relating them to the management and simulation of workflows in [82]. The six guidelines are briefly outlined in the following.

1. *Construction Adequacy*: This guideline postulates that it is impossible to prove that a model correctly reflects the reality. Instead the modelers and the model users need to agree that the model is adequate with regard to a specific problem. This means that an agreement about the problem as well as about its representation must be reached.

2. *Language Adequacy*: There are two criteria regarding the modeling language used to create the model. First, it has to be suitable, i.e., it must allow the modelers to represent the reality. Second, it must be ensured that the language is used correctly. While the first criterion directly addresses the modeling language, the second criterion refers to its application.

3. *Economic Efficiency*: The creation of a model causes costs, e.g., modelers need time to create it. These costs must be justified by the benefits of the model's use.

4. *Clarity*: A model needs to be comprehensible and explicit. This requires that the model is represented on a suitable level of abstraction which is determined by the

purpose of the model. Further, it must be ensured that the understandability is supported by the graphical arrangement of the model elements and the model should be simple. That is, it should comprise as few information objects as possible. Lastly, the models should be suited to the information needs of its users.

5. *Systematic Design*: The reality is often described in different models. In such cases information objects should be consistently defined and used in all models. Moreover, the relations between all models should be clear and consistent.

6. *Comparability*: If there are different languages used to create models for the same purpose, it must be possible to transform models between these languages. Furthermore, similar issues should be represented in a similar way in all models.

A set of rules overlapping with the guidelines of modeling is discussed in [83]. In accordance with [84], this set focuses the use of conceptual models in the implementation of systems. Whereas these rules and the guidelines of modeling apply to conceptual modeling in general, Mendling et al. [85] proposed the Seven Process Modeling Guidelines (7PMG) that constitute a set of guidelines specific to the domain of business process modeling. This set of rules was derived from an analysis of understandability and error probability in business process models. The 7PMG comprise the following guidelines.

1. *Use as few elements as possible*: As larger models tend to be harder to understand and have a higher probability to contain errors, a model should contain as few elements as possible.

2. *Minimize the routing paths per element*: The higher the degree of incoming or outgoing control flow connections of elements in the process model is, the more cumbersome it is to understand the model. Thus, elements should have as few control flow connections as possible.

3. *Use one start and one end event*: The presence of multiple start or end events has a negative impact on the understandability and the error probability of business process models. Consequently, there should be one start and one end event in each model.

4. *Model as structured as possible*: It is more cumbersome to interpret unstructured models and they also tend to contain more errors than structured models. A model is structured, if all elements that split a path into several paths, are matched by another element that joins all these paths.

5. *Avoid OR routing elements*: Models that contain OR split elements are ambiguous as they usually allow for a variety of combinations of the connected paths to be executed. Thus, models should only contain parallel and alternative split and join elements.

6. *Use verb-object activity labels*: The interpretation of an activity label will in general be easier, if the label consists of a verb and an object, e.g., "evaluate application" instead of "application evaluation".

7. *Decompose a model with more than 50 elements*: This guideline is related to the first guideline. Models with more than 50 elements tend to have an error probability that is up to 50% higher than smaller models. Thus, if a model reaches a size of more than 50 elements it should be split into a number of smaller models.

When modelers have applied the modeling technique including the discussed guidelines and created a model, the *validation* phase is carried out next. Its purpose is to check whether the model represents the informal description. Therefore, the model is again translated to a natural language description and compared to the informal specification. Leopold et al. [86] present a technique to generate natural language documents from business process models. Furthermore, approaches to compliance checking of business process models support modelers in determining whether a business process satisfies regulatory rules [87–89].

In addition to the validation, the *verification* deals with examining whether the model concepts have been applied consistently. For the verification of business process models there exists a variety of approaches in the field of BPM. First, the verification of the soundness of the control flow was examined in a number of papers [90–92]. A business process is called sound, if it is free of anomalies like deadlocks and livelocks. A more

relaxed soundness property that requires each activity to be part of at least one path from the start to the end node was discussed in [93]. The extension of such approaches to cross organizational business processes was discussed in [94, 95]. Furthermore, there are approaches which focus on the verification of business process models with regard to the informational perspective [96, 97].

There exists a broad spectrum of notions of modeling quality. These notions typically cover various aspects of the information modeling process and aim to provide guidelines for the entire modeling process. The SEQUAL framework [98] constitutes such a view on quality aspects of conceptual models. It addresses the syntactic, the semantic, and the pragmatic quality. The framework was subsequently extended [99, 100] by considering more levels of Stamper's semiotic ladder [101]. Consequently, the latest version of the SEQUAL framework also addresses the physical, the empirical, the social and the organizational quality. The Bunge-Wand-Weber ontology [102–104] relies on the scientific ontology proposed by Bunge [105]. It was used to examine the redundancy and the excess of the constructs in modeling languages, see [106] for an overview. Additionally, Wand and Wang [107] use the ontology as a means to compare the modelers' view of the domain to how this view is captured in a model. Finally, the conceptual modeling quality framework [108] combines both – the SEQUAL framework and the Bunge-Wand-Weber ontology.

Although there exist many approaches that support experts in modeling, business process modeling is still a rather creative process. The reason is that the approaches are generic. Thus, the outcome of the modeling process, i.e., the model, relies on the perception of the modelers and their ability to express it in a model. In this regard, Rittgen [11] criticizes that most modeling techniques only focus the correct use of the modeling language concepts, but do not provide any further guidelines on how to derive a model from the informal specification. As a result, models depicting the same business process might be very dissimilar, even if they are created for the same purpose. Consequently, the identification of correspondences between process models can become very cumbersome and time consuming, motivating the development of techniques that automate this comparison.

2.3 Business Process Modeling Languages

In order to capture business processes as models, a plethora of modeling languages has been proposed. Like other modeling languages they define the *syntax* and the *semantics* of the concepts that can be used to create models [109]. Whereas the syntax defines symbols and how they can be combined to create expressions, the semantic assigns meanings to these expressions. Therefore, it comprises the *semantic domain* and the *semantic mapping*. The former defines the relevant concepts and the latter maps these concepts to the syntactic symbols and expressions. In this regard, simpler expressions can also be combined to complex expressions whose meanings depend on the simpler expressions. Karagiannis and Kühn [78] add a further element to the modeling language: the *notation*. It describes how expressions are represented, e.g., by graphical elements or by textual sentences. The relations between these elements are summarized in Figure 2.5.

In this manner, business process modeling languages typically define concepts to depict activities and execution constraints exposed on them, like control flows, gateways, or events. Additionally, there might be concepts to express roles, people, or systems that are responsible for the execution of activities or to express objects or information that is needed and modified during process execution. However, these concepts only allow modelers to define the basic layout and elements of business processes. In order to convey the actual meaning of the elements modelers need to annotate them. This is usually done by defining a label which constitutes a description of the according element. The description highly depends on the domain of the business process, i.e., process models from different domains, like industrial production, university administration, or health

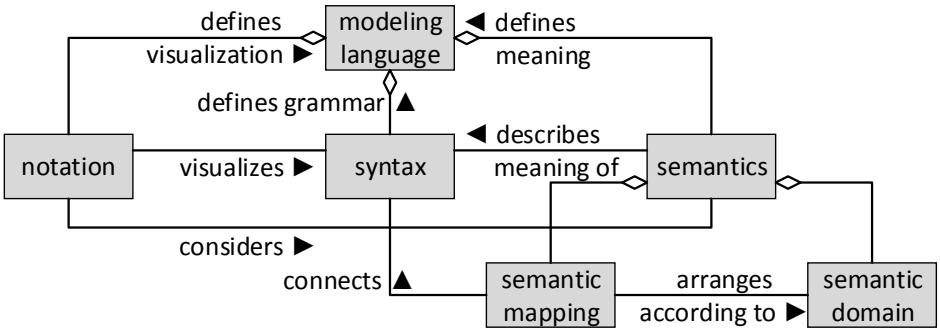

FIGURE 2.5: Elements of modeling languages, adopted from [78, p. 184]

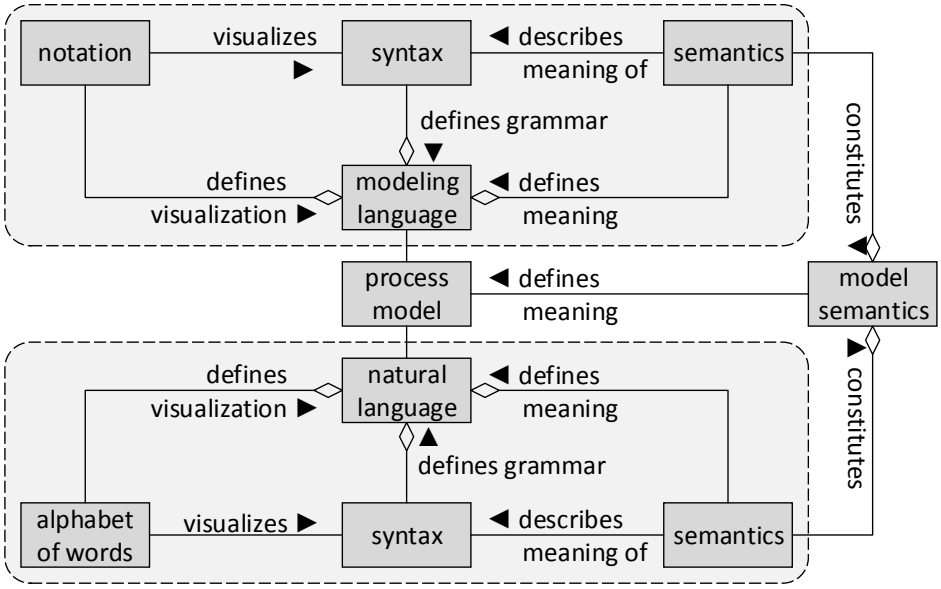

FIGURE 2.6: Components of the semantics of business process models, adopted from [39, p. 12]

insurance, are very likely to contain totally different descriptions. Due to the unlimited variety of possible scenarios, modelers typically use natural language to create such descriptions. Hence, the actual meaning of a business process model does not only depend on the specific modeling language in use, but also on the natural language used to describe the model elements. Leopold [39] summarized this interplay of the modeling language and the natural language in process models as shown in Figure 2.6.

As the actual semantics of the model elements largely depends on the labels, they constitute the primary source for the automatic identification of corresponding activity pairs. However, additional information, like element types, execution constraints etc., are encoded using a business process modeling language. Especially, the relations between the model elements in general and the activities in particular provide another source of information for matching. The reason is that these relations encode the execution semantics of the process model and define constraints that might be exploited to identify correspondences. In the following, a basic understanding of relevant model elements for business process modeling and their relations is provided by the definition of the canonical business process model. As the canonical model permits the representation of models defined with different languages, it serves as the basis for all matching techniques presented in this thesis. This way a broad applicability of these techniques is achieved.

That is, the techniques can be applied regardless of the process modeling language used to capture the business processes and hence many organizations are able to utilize the techniques. In this regard, the Business Process Model and Notation (BPMN), the Event Driven Process Chain (EPC), and Petri nets are outlined. Besides being well-known in academia and practice, these three modeling languages were chosen because the four real-world datasets used for analysis and evaluation in this thesis (see Section 3.4) are based on them. Thus, to ensure comprehensibility and replicability of the techniques and analyses presented in this thesis, the languages are not only introduced, but also formally defined and mapped to the canoncial format. To conclude the discussion of the modeling languages, an overview of further languages is provided.

2.3.1 The Canonical Business Process Model

Various approaches exist that allow to encode business process models in a language-independent format. In this regard, Petri nets [110] have been suggested as such a format, especially in the context of examining the execution semantics of process models. Here, a variety of approaches to map models of commonly used languages to Petri nets exists, e.g., for BPMN [111] and for EPC [112], an overview of mapping approaches is provided in [113].

Furthermore, several abstract business process modeling languages have been defined. The canonical format of the advanced process model repository [114, 115] constitutes such an abstract language whose metamodel is presented in [115]. According to this format, business process models are graphs that consist of nodes and edges connecting these nodes to represent the control flow. The set of nodes is further subdivided into gateways, events, states, or activities. These basic elements allow to capture the functional as well as the behavioral perspective. Additionally, objects or roles can be assigned to the nodes and especially to the activities in order to include the informational and organizational perspective.

The notion of business process graphs is introduced in [116, 117]. In contrast to the canonical format it focuses the functional and behavioral perspective and does not

include the assignment of roles or objects to nodes. Furthermore, it does not explicitly distinguish specific sets of nodes, but assigns types to nodes. Similarly, the jBPT library[7] defines process models based on directed graphs [118].

In this thesis, the notion of business process graphs is adopted. That is because in Petri nets there are only two types of nodes. When transforming a model from a different language to a Petri net, all nodes have to be mapped to these types. This usually leads to a mapping where a variety of different model elements is encoded as the same Petri net concept. Consequently, it is hard to distinguish between activities and other types of business process elements like parallel gateways or events. Furthermore, the matching techniques in this thesis only rely on the functional and organizational perspective (cf. Chapter 3). The reason is that activities are regarded as similar if they are carried out for a similar underlying purpose. To this effect, it does not matter whether an activity is performed by a different role, especially as there might exist different roles in different organizations or units. Additionally, different IS might be in place. Furthermore, objects required for or resulting from the execution of an activity are neglected as they also might be organization-specific and do not influence the reason why an activity is carried out. Instead, the purpose of the activity determines the use of such objects. Another problem in this regard is that roles, objects, etc. might be labeled heterogeneously, too. Thus, to utilize them during matching, they also need to be aligned. However, the computation of such alignments is a different problem, e.g., the alignment of entities in the informational perspective is discussed in the field of schema matching [119, 120].

Definition 2.4 (Canonical business process model) *Let $\mathscr{L}$ be a set of labels, and $\mathscr{T} = \{activity, event, state, xor, and, or\}$ be the set of types. A canonical business process model P is a 5-tuple*

$$(N, A, E, \lambda, \tau)$$

such that

- *N is the set of nodes;*
- *$E \subseteq N \times N$ is the set of edges;*

[7] https://www.openhub.net/p/jbpt, accessed: 13/01/2017

- $\lambda : N \nrightarrow \mathscr{L}$ *is a partial function that maps nodes to labels;*
- $\tau : N \rightarrow \mathscr{T}$ *is a function that maps each node to a type; and*
- $A = \{a | a \in N \wedge \tau(a) = activity\}$ *is the set of activities.*

The canonical model does not provide a notation, but it provides the syntax and semantics to formally define business process models. To outline its application, the example process which was introduced in Section 2.1 is used. This process and accordingly its canonical model comprise six activities that represent the steps executed in the process. This includes checking if the application was submitted in time, checking if the application is complete, assessing the qualification of the student, accepting the student, rejecting the student and archiving the documents. Moreover, it will contain an AND-split as well as an AND-join as both formal checks are carried out in parallel. It also contains a XOR-split and a XOR-join as the student is either accepted or rejected. Finally, there are a number of edges connecting all these nodes. Note that unless stated otherwise the terms process model and business process model are used in this thesis to refer to the canonical business process model.

2.3.2 Business Process Model and Notation

The Business Process Model and Notation (BPMN) is a widely adapted business process modeling language that was initially developed at the beginning of the 21st century and was maintained by the Business Process Management Initiative. Currently, it is a standard managed by the Object Management Group and available in version 2.0 [121]. Figure 2.7 illustrates the BPMN model for the application assessment process.

The process model contains six activities (rounded rectangles) as well as a start and an end event (circles). These events mark the beginning and the end of the process.

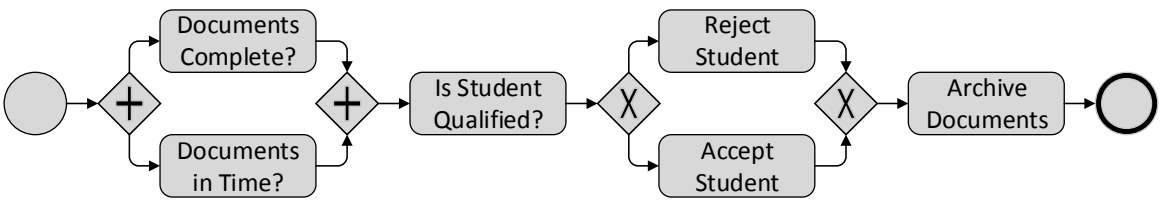

FIGURE 2.7: BPMN model for the university admission example

Furthermore, there are two parallel gateways (diamond shape with a plus) and two exclusive gateways (diamond shape with a cross) to capture the routing behavior of the process. Finally, sequence flows are used to connect these elements and to define the control flow.

This model only contains a small subset from the huge variety of elements BPMN offers. Despite the extensive choice of elements that enables modelers to represent more complex issues, e.g., choreographies, error handling, or data exchange, zur Muehlen and Recker [122] observed that only a small subset of the elements is used in practice. Based on this finding and without loss of generalizability, only a small subset of BPMN elements is considered here. These elements are presented in Figure 2.8 .

In BPMN the task element is used to represent activities. Furthermore, there are three different types of events. While the start event indicates the beginning of an instance, the end event marks its termination. Intermediate events are used to model events that may occur during the execution of an instance. The gateways represent points during process execution where the flow is split into separate paths or where such paths are joined. The exclusive gateway represents a decision where the flow is routed to one of the subsequent paths or where the execution is continued as soon as one of the paths reaches the join. The inclusive gateway marks a decision where a subset of the subsequent paths is chosen for execution or where the execution will continue when all activated paths reach the gateway. The parallel gateway marks a point where all subsequent paths are activated or where execution will be continued as soon as all preceding paths terminate. A pool is

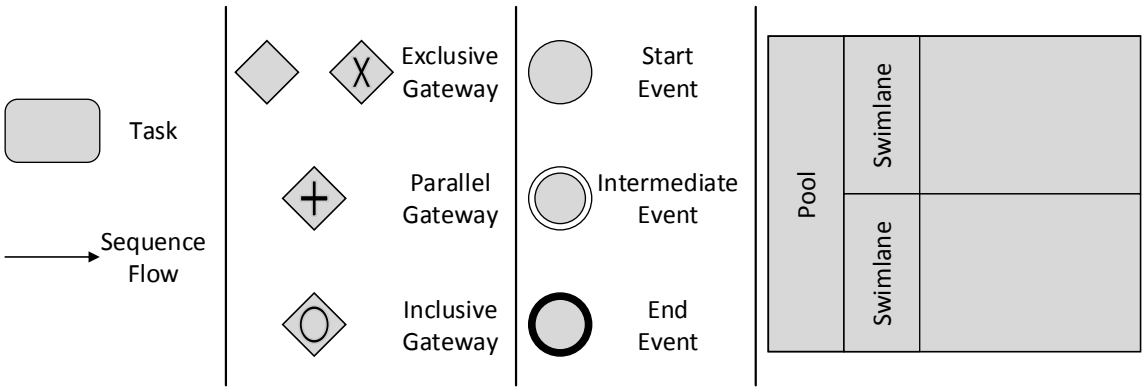

FIGURE 2.8: Basic BPMN elements

used to model systems, roles, or organizations that are responsible for the execution of tasks within them. Pools can be further subdivided into lanes which indicate a department or subsystem. Finally, sequence flows are used to connect the flow elements in a model in order to depict the control flow.

The BPMN standard describes the graphical notation used to model processes, but it does not include a formal definition. However, as such a definition is needed in order to map BPMN models to canonical process models, the formal definition introduced in [123] is adopted here.

Definition 2.5 (BPMN process model) *Given the set of labels $\mathscr{L}$, a BPMN process model P_B is a 13-tuple*

$$(A_B, \mathscr{E}_B, \mathscr{E}_B^S, \mathscr{E}_B^I, \mathscr{E}_B^E, \mathscr{G}_B, \mathscr{G}_B^X, \mathscr{G}_B^I, \mathscr{G}_B^P, E_B, L_B, \iota_B, \lambda_B)$$

such that

- *A_B is the sets of tasks;*
- *$\mathscr{E}_B$ is the set of events that can be partitioned into the disjoint sets of start $\mathscr{E}_B^S$, intermediate $\mathscr{E}_B^I$ and end $\mathscr{E}_B^E$ events;*
- *$\mathscr{G}_B$ is the set of gateways that can be partitioned into the disjoint sets of exclusive $\mathscr{G}_B^X$, inclusive $\mathscr{G}_B^I$ and parallel $\mathscr{G}_B^P$ gateways;*
- *$E_B \subseteq (A_B \cup \mathscr{E}_B \cup \mathscr{G}_B) \times (A_B \cup \mathscr{E}_B \cup \mathscr{G}_B)$ is the set of sequence flows;*
- *L_B is the potentially empty set of lanes;*
- *$\iota_B : N_B \nrightarrow L_B$ is a function that maps nodes to lanes; and*
- *$\lambda_B : (A_B \cup \mathscr{E}_B \cup \mathscr{G}_B) \nrightarrow \mathscr{L}$ is a partial function that maps nodes to labels.*

A BPMN model can be straightforwardly represented as a canonical process model. Here, the set of nodes comprises all tasks, events, and gateways in the BPMN model and the set of activities contains all tasks. The set of edges and the label function are identical to those in the BPMN model. Finally, the type function classifies a node as an activity, if

its BPMN counterpart is a task and as an event, if its counterpart is an event. Similarly, all exclusive gateways are assigned to the xor-type, all parallel gateways to the and-type and all inclusive gateways to the or-type. This mapping is summarized in the following definition.

Definition 2.6 (BPMN to canonical business process model mapping) *Let* $P_B = (A_B, \mathcal{E}_B, \mathcal{E}_B^S, \mathcal{E}_B^I, \mathcal{E}_B^E, \mathcal{G}_B, \mathcal{G}_B^X, \mathcal{G}_B^I, \mathcal{G}_B^P, E_B, L_B, \iota_B, \lambda_B)$ *be a BPMN process model.* P_B *can be represented as a canonical process model* $P = (N, A, E, \lambda, \tau)$ *such that*

- $N = A_B \cup \mathcal{E}_B \cup \mathcal{G}_B$
- $E = E_B$
- $A = A_B$
- $\lambda = \lambda_B$
- $\tau(n) = \begin{cases} activitiy & \text{if } n \in A_B \\ event & \text{if } n \in \mathcal{E}_B \\ xor & \text{if } n \in \mathcal{G}_B^X \\ and & \text{if } n \in \mathcal{G}_B^P \\ or & \text{otherwise} \end{cases}$

2.3.3 Event Driven Process Chain

The Event Driven Process Chain (EPC) was developed in the context of the architecture of integrated information systems which is an approach for the development of information systems that adhere to organizational requirements [124]. In this regard, EPC models serve as a means to capture business processes. Figure 2.9 presents an EPC model for the application assessment process.

Obviously, the model comprises more elements than the corresponding BPMN model. The reason is that in EPC models it is required that on each path from a start to an end event functions (rounded rectangles) and events (hexagons) alternate, i.e., on each

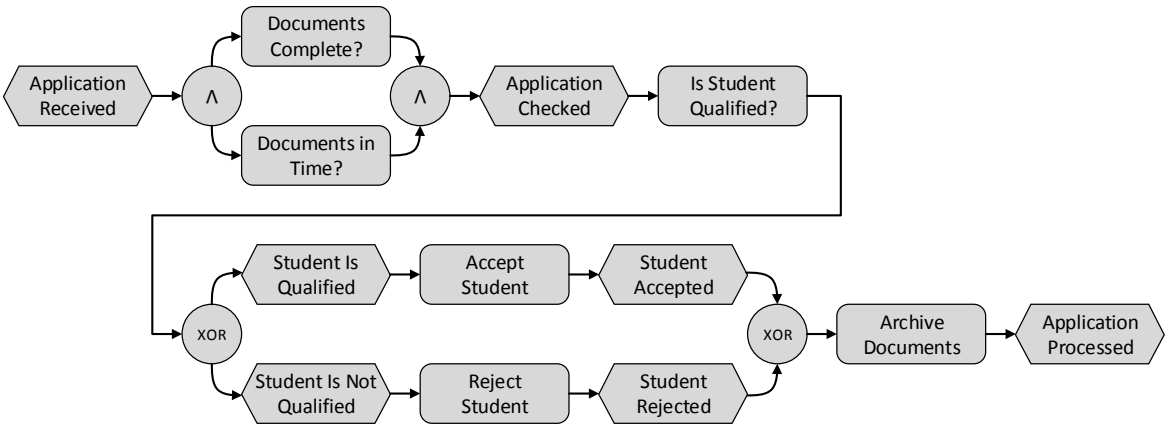

FIGURE 2.9: EPC model for the university admission example

path each function must be followed by an event and each event must be followed by a function when skipping the connectors (circles). Thus, the number of events in an EPC model usually exceeds the number of events in an equivalent BPMN model. The possible elements of an EPC model are shown in Figure 2.10.

As already outlined in the university admission example, EPC comprises functions, activities and events. Moreover, there are three types of connectors which are used to split and join the flow. These types comprise the XOR-connector, the AND-connector, and the OR-connector. But there exists a restriction regarding the use of the XOR-connector and the OR-connector. An event cannot be followed by a split connector of these types, as the split requires a decision that can only be actively made by executing a function. Finally, the elements are connected by control flows.

While the number of EPC elements is small, there exists a number of extensions that add further elements to the notation. For example, the extended EPC provides elements to annotate functions with organizational units that are responsible for their execution and information objects that are required for or the result of the execution of a function. It also comprises elements to hierarchically organize EPC models. Furthermore, there exist

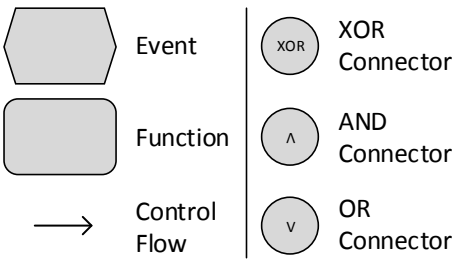

FIGURE 2.10: EPC elements

several variants, e.g., an object-oriented extension [125], the modified EPC [126], and yet another EPC [127]. Sarshar et al. [128] give an overview of variants and additional elements. A formal definition of EPC process models in the context of this work is provided in the following.

Definition 2.7 (EPC process model) *Given the set of labels $\mathcal{L}$, an EPC process model P_E is an 8-tuple*

$$(A_E, \mathcal{E}_E, \mathcal{G}_E, \mathcal{G}_E^X, \mathcal{G}_E^A, \mathcal{G}_E^O, E_E, \lambda_E)$$

such that

- *A_E is the set of functions;*
- *$\mathcal{E}_E$ is the set of events;*
- *$\mathcal{G}_E$ is the set of connectors that can be partitioned into the disjoint sets of xor $\mathcal{G}_E^X$, and $\mathcal{G}_E^A$ as well as or $\mathcal{G}_E^O$ connectors;*
- *$E_B \subseteq (A_E \cup \mathcal{E}_E \cup \mathcal{G}_E) \times (A_E \cup \mathcal{E}_E \cup \mathcal{G}_E)$ is the set of control flows; and*
- *$\lambda_E : (A_E \cup \mathcal{E}_E \cup \mathcal{G}_E) \nrightarrow \mathcal{L}$ is a partial function that maps nodes to labels.*

Similar to BPMN an EPC process model can be mapped to a canonical process model where the set of nodes comprises all functions, events, and gateways. Further, the set of functions constitutes the set of canonical activities and the set of edges corresponds to the set of control flows in the EPC model. Consequently, the labeling functions of both models are identical. Finally, the type function classifies each function as an activity and each event as an event. Elements whose counterpart in the EPC model is a connector are assigned to the respective connector type, e.g., xor-connectors are assigned to the xor-type. The mapping is formally defined in the following.

Definition 2.8 (EPC to canonical business process model mapping) *Let $P_E = (A_E, \mathcal{E}_E, \mathcal{G}_E, \mathcal{G}_E^X, \mathcal{G}_E^A, \mathcal{G}_E^O, E_E, \lambda_E)$ be an EPC process model. P_E can be represented as a canonical process model $P = (N, A, E, \lambda, \tau)$ such that*

- *$N = A_E \cup \mathcal{E}_E \cup \mathcal{G}_E$*

- $E = E_E$
- $A = A_E$
- $\lambda = \lambda_E$
- $\tau(n) = \begin{cases} activitiy & if\ n \in A_E \\ event & if\ n \in \mathscr{E}_E \\ xor & if\ n \in \mathscr{G}_E^X \\ and & if\ n \in \mathscr{G}_E^A \\ or & otherwise \end{cases}$

2.3.4 Petri Net

Based on the notion of finite-state machines Petri [110] developed a first version of the Petri net modeling language to represent concurrency within distributed systems. It is widely adapted in many scientific areas, e.g., computer science and machine engineering. Its use in the field of BPM was amongst others discussed in [129]. The representation of the university admission example as a Petri net is shown in Figure 2.11.

Basically, there are two types of nodes in a Petri net, the transitions (squares) and the places (circles). Places represent conditions that can be activated during the execution of the underlying system. Transitions instead describe the active parts of a system, i.e., the events or functions that modify a system's state. In the context of business processes transitions can be used to depict the process' activities. The actual state of a system is represented by a marking. Such a marking consists of a set of tokens where each token is

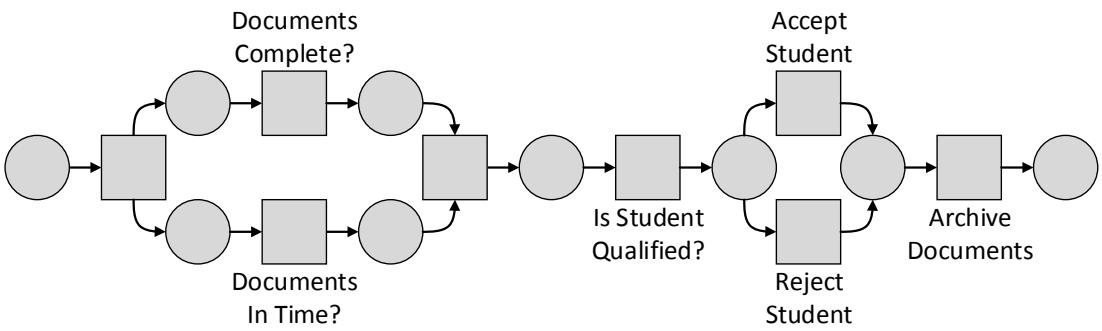

FIGURE 2.11: Petri net model for the university admission example

assigned to a place. Given such a marking it is possible to determine which transitions can be activated next. In other words, it is possible to determine the actions that can occur and that can modify the systems' state. In general, all transitions where each of the preceding places is marked by at least one token are enabled. If an enabled transition is fired, one token is removed from each of its input places and one token is added to each of its output places. Consequently, places can be used to represent XOR-joins or -splits as only one of the following transitions can be activated if the place is marked by a token. Similarly, transitions can be used to represent AND-joins or -splits because a transition is only activated, if all input places are marked and because it marks all output places during activation. Following a widely adapted definition of Petri nets [130] a Petri net process model is seen as a directed graph consisting of transitions and places.

Definition 2.9 (Petri net process model) *Given the set of labels $\mathscr{L}$, a Petri net process model P_P is a tuple*

$$(T_P, \Theta_P, E_P, \lambda_P)$$

such that

- T_P *is the set of transitions;*
- Θ_P *is the set of places;*
- $E_B \subseteq (T_P \times \Theta_P) \cup (\Theta_P \times T_P)$ *is the set of arcs connecting transitions to places and places to transitions; and*
- $\lambda_E : (A_E \cup \mathscr{E}_E \cup \mathscr{G}_E) \nrightarrow \mathscr{L}$ *is a partial function that maps nodes to labels.*

As places and transitions are used to depict various basic process elements [13, 112], transforming a Petri net into a canonical business process model is not as straightforward as it is to map a BPMN or EPC model to the canonical format. In this thesis, transitions that possess a label are transformed to activities. Transitions without a label, also referred to as silent transitions, are treated in different ways. If a silent transition is connected to more than one input or more than one output transition, it constitutes an AND-gateway. All other transitions will be treated as events. Note, that a labeled transition that has

multiple input or output places is treated as an activity. Additionally, places are treated as states, if they have at most one input and at most one output transition. Otherwise, places are classified as XOR-gateways.

Definition 2.10 (Petri net to canonical business process model mapping) *Let* $P_P = (T_P, \Theta_P, E_P, \lambda_P)$ *be a Petri net process model.* P_P *can be represented as a canonical process model* $P = (N, A, E, \lambda, \tau)$ *such that*

- $N = T_P \cup \Theta_P$
- $E = E_P$
- $\lambda = \lambda_P$
- $\tau(n) = \begin{cases} activitiy & if \quad n \in T_P \wedge n \in supp(\lambda_P) \\ and & if \quad n \in T_P \wedge n \notin supp(\lambda_P) \wedge \\ & \quad (\{(n,n')|(n,n') \in E_P\} > 1 \vee \{(n',n)|(n',n) \in E_P\} > 1) \\ event & if \quad n \in T_P \wedge n \notin supp(\lambda_P) \wedge \\ & \quad (\{(n,n')|(n,n') \in E_P\} \leq 1 \wedge \{(n,n')|(n,n') \in E_P\}) \leq 1 \\ xor & if \quad n \in \Theta_P \wedge \\ & \quad (\{(n,n')|(n,n') \in E_P\} > 1 \vee \{(n',n)|(n',n) \in E_P\} > 1) \\ state & otherwise \end{cases}$
- $A = \{a | a \in N \wedge \tau(a) = activity\}$

2.3.5 Other Notations

Besides these three modeling languages and their variants there exist many other business process modeling languages. In the field of software development the Unified Modeling Language (UML) [131] provides a number of modeling languages to specify, visualize, and document software applications. Within this continuum the UML sequence diagram and the UML state machine diagram constitute modeling languages to capture processes within a software system.

In the context of service oriented architectures the Web Services Business Process Execution Language (BPEL) is a language whose purpose is to support the design and execution of business activities and their orchestrations [132]. BPEL is block structured, i.e., the process is basically described as a set of nested blocks where each block contains one start and one end node. The basic block type is the sequence of one or more activities or blocks. Other types include parallel or exclusive branches and loops. This block structure can be straightforwardly transformed into a graph structure.

Both languages can be mapped to a canonical business process model similar to BPMN and EPC models. That is, all nodes and edges are part of the canonical model and nodes are assigned to a respective canonical element type.

All the modeling languages considered so far are imperative business process modeling languages where models based on these modeling languages capture all possible execution scenarios [133]. However, there are scenarios where such a definition of a process yields complex and inflexible models. For example, treatment processes in hospitals highly depend on the specific disease, the circumstances of the patient, and the availability of resources. Capturing the treatments with imperative process modeling languages has no prospect of success due to the enormous amount of possible scenarios. Here, declarative modeling languages that focus on the main characteristics of the process can be used. More precisely, such languages allow to specify constraints that restrict the space of possible scenarios by excluding cases that are prohibited. For example, they might provide means to capture the exclusive use of tasks or the mandatory application of another activity. Declare [134], DCR Graphs [135] and SCIFF [136] are examples of declarative business process modeling languages. An in-depth discussion of the difference between imperative and declarative business process modeling languages can be found in [137].

The functional perspective of declarative business process models can be mapped to the canonical format quite easily, i.e., all activities in the declarative model are transferred to the canonical model. Considering the behavioral perspective is not advisable. The reason is that there are usually only a few but essential characteristics of this perspective depicted in declarative process models. Thus, it is possible to derive structurally different imperative models that adhere to the declarative constraints. As a consequence, only those

matching techniques introduced in this thesis that solely rely on the labels of activities can be applied, if declarative models need to be matched.

2.4 Summary

This chapter introduced BPM as the context of this thesis. In essence, it defined business processes as coordinated executions of activities and BPM as a tool set that supports all phases in the lifecycles of business processes. In this context, the use of business process models which are restricted representations of business processes was motivated. Next, approaches to business process modeling were discussed. This comprised a basic overview of the modeling process including modeling techniques. In this regard, it was pointed out that despite the broad range of approaches that assist modelers' in creating models, business process modeling is a creative process. Consequently, the quality of business process models depends on the modelers' capabilities to capture a business process based on a certain modeling language. As a result, the same process might be represented in different ways impacting the identification of correspondences and motivating the need for process model matching techniques. Lastly, the chapter presented a formal definition of a canonical business process model. This notion provides a basis for all matching techniques in this thesis and permits their application to different modeling languages. In this regard, it was shown how different modeling languages, including BPMN, EPC, and Petri nets, can be represented as canonical business process models.

3

Matching Business Process Models

H1: The identification of correspondences between business process models is a challenge for organizations which is not sufficiently supported by existing approaches.

This chapter introduces business process model matching in more detail. In this regard, elementary concepts are defined and the sub-hypothesis H1 is examined to provide evidence to the practical and the scientific demand. Moreover, the characteristics of the empirical datasets which serve as the basis for the development and the evaluation of the matching techniques are outlined.

The chapter is structured into five sections. A basic understanding of business process model matching is introduced in Section 3.1. This includes illustrative examples and formal definitions. Next, the practical demand is addressed in Section 3.2. Here, an

overview of application scenarios for business process model matching techniques is provided to substantiate the need for such techniques in practice. After that, the scientific demand is discussed based on the results of a literature review in Section 3.3. At this point, the verification of sub-hypothesis H1 concludes and the chapter continues with the description of the datasets in Section 3.4. Lastly, Section 3.5 summarizes this chapter.

3.1 Basic Concepts

In this thesis, the terminology from the field of ontology matching is adopted. Thus and unless stated otherwise, the terminology introduced in this section is oriented towards [138]. Accordingly, business process model matching is seen as the process of identifying an *alignment* between two process models. The alignment refers to the functional perspective or more specifically to the activities in the process models. It consists of *correspondences* which indicate activity sets that represent the same functionality in both process models.

An example of an alignment is shown in Figure 3.1 where an alignment between the application assessment process from the previous chapter ("Process A") and another application assessment process ("Process B") is presented. Although both process models represent the same higher level process, they implement this process in different ways. In both processes, the first step is the formal verification of the student's application. Whereas in "Process A" there are two activities (α_1 and α_2) regarding this step, there is only one activity (β_1) in "Process B". The next step is the assessment of the application which in both processes is represented by a single activity (α_3 and β_2). Lastly, a decision is made and steps to enforce this decision are carried out. This is implemented by two activities (α_4 and α_5) in "Process A" and three activities (β_3, β_4 and β_5) in "Process B". The final archiving of the documents (α_6) is only part of "Process A".

Two kinds of correspondence relations can be distinguished. On the one hand, there are *elementary* or *1:1-correspondences* where one activity from the first process corresponds to exactly one activity in the second process and vice versa. In the example, α_3 and β_2

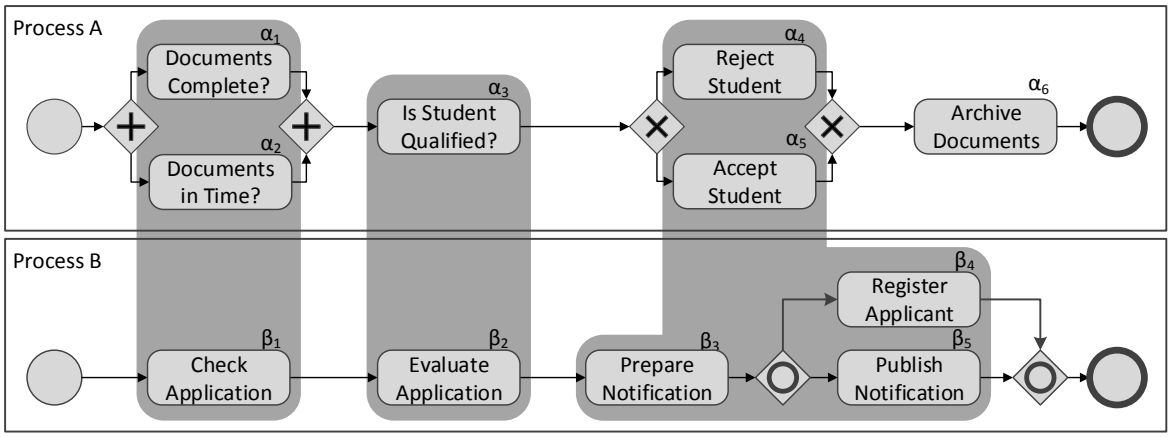

FIGURE 3.1: An alignment between two university admission process models

constitute such an elementary correspondence. On the other hand, *complex correspondences* refer to correspondence relations where there are sets of activities involved. This can be the case if one activity from a process corresponds to a set of activities from the other process and each of these activities only corresponds to the first activity. Those correspondence relations are also referred to as *1:n-correspondences*. In the example α_1, α_2 and β_1 constitute such a correspondence. Another possible scenario is the existence of so called *m:n-correspondences*. That is, sets of activities from both processes have a correspondence relation, e.g., the activities referring to the enforcement of a decision.

In this thesis, an alignment is formally defined as a binary relation over the sets of activities of two processes. In other words, it is a set of activity pairs where activity pairs consist of one activity from each process. This definition allows to represent elementary correspondences as well as complex correspondences. Table 3.1 outlines this representation by presenting the alignment matrix for the example. In the table the

TABLE 3.1: Alignment matrix for the university admission example

	α_1	α_2	α_3	α_4	α_5	α_6
β_1	1	1	0	0	0	0
β_2	0	0	1	0	0	0
β_3	0	0	0	1	1	0
β_4	0	0	0	1	1	0
β_5	0	0	0	1	1	0

activities of "Process A" are represented as columns and those of "Process B" as rows. Each cell contains a value of 1, if the according activity pair corresponds, and a value of 0 otherwise. In case of an activity pair being an elementary correspondence, all cells in the respective row and column contain a value of 0 except for the cell representing the pair. Complex correspondences are encoded by a 1 in each cell that belongs to rows and columns representing activities from the respective sets.

Definition 3.1 (Alignment, Correspondence) *Given two process models* $P = (N, A, E, \lambda, \tau)$ *and* $P' = (N', A', E', \lambda', \tau')$*, an alignment* $\mathscr{A}$ *is a binary relation*

$$\mathscr{A} \subseteq A \times A'$$

where each pair of activities $c = (a, a')$ *with* $a \in A$*,* $a' \in A'$ *is referred to as a correspondence, if it is part of the alignment, i.e.,* $c \in \mathscr{A}$*. Furthermore, the domain* $dom(\mathscr{A}) = A$ *and the co-domain* $cod(\mathscr{A}) = A'$ *of the alignment are used to refer to the sets of activities the alignment is defined over.*

Based on this definition an elementary correspondence can be formally defined as follows.

Definition 3.2 (Elementary correspondence) *Let* $\mathscr{A} \subseteq A \times A'$ *be an alignment over two sets of activities. A correspondence* $c = (a, a')$ *with* $c \in \mathscr{A}$ *is called an elementary or 1:1-correspondence respectively, if both activities do not correspond to any other activity with regard to the alignment, i.e.,* $\{a'' | (a, a'') \in \mathscr{A}\} = \{a'\} \wedge \{a'' | (a'', a') \in \mathscr{A}\} = \{a\}$.

Complex correspondences are formally defined in a similar way.

Definition 3.3 (Complex correspondence) *Let* $\mathscr{A} \subseteq A \times A'$ *be an alignment over two sets of activities. Two subsets of the activity sets* $A_s \subseteq A$*,* $A'_s \subseteq A_s$ *constitute a complex correspondence* (A_s, A'_s)*, if both sets are not empty and in total contain more than two activities, i.e.,* $|A_s| > 0 \wedge |A'_s| > 0 \wedge |A_s| + |A'_s| > 2$*. Furthermore, it is required that for all activities from the first subset the set of corresponding activities with regard to the alignment is the second subset and vice versa, i.e.,* $\forall a \in A_s : \{a'' | (a, a'') \in \mathscr{A}\} = A'_s \wedge \forall a' \in A'_s : \{a'' | (a'', a') \in \mathscr{A}\} = A_s$*. If one of the subsets consists of one activity and the other of more than one*

activity $(|A_s| = 1 \wedge |A'_s| > 1) \vee (|A_s| > 1 \wedge |A'_s| = 1)$*, the complex correspondence is called a 1:n-correspondence. If both subsets contain more than one activity* $(|A_s| > 1 \wedge |A'_s| > 1)$*, the complex correspondence is referred to as an m:n-correspondence. Moreover, each subset with more than one element* $|A_s| > 1$ *is a corresponding activity cluster.*

As stated at the beginning of this section, the identification of an alignment between two process models is a process. It is referred to as the *matching process* and can be carried out by a human who manually identifies the alignment or by a *matching technique* (also *matcher*) which automatically determines an alignment. From an abstract point of view matching processes can be described as a function whose input is a set of process model pairs for which alignments need to be determined. Additionally, a sequence of alignments that might be empty is passed to the matching processes as input. The output of the process consists of an alignment for each input process model pair.

Definition 3.4 (Matching process, Matching technique) *Let* $\mathbb{P}^{in} = (MP_i)_{i=1}^{k}$ *be a sequence of* $k \in \mathbb{N}$ *process model pairs with* $MP_i = (P_i, P'_i)$ *where* $P_i = (N_i, A_i, E_i, \lambda_i, \tau_i)$ *and* $P'_i = (N'_i, A'_i, E'_i, \lambda'_i, \tau'_i)$ *are two process models. Furthermore, let* $\mathbb{A}^{in} = (\mathscr{A}_j)_{j=1}^{l}$ *be a potentially empty sequence of* $l \in \mathbb{N}_0$ *alignments. Then, a matching process is a function*

$$\mathbb{A}^{out} = match(\mathbb{P}^{in}, \mathbb{A}^{in})$$

that determines an alignment for each of the given process model pairs, i.e., $\mathbb{A}^{out} = (\mathscr{A}_{i=1}^{k})$ *with* $\mathscr{A}_i \subseteq A_i \times A'_i$*. To identify these alignments the sequence of alignments provided as input might be exploited. A piece of software that automatically executes matching processes is referred to as a matching technique or matcher, respectively.*

Based on this abstract view there are several design options for matching techniques. First, a matching technique could independently compute an alignment for each of the given process model pairs and ignore the alignments provided as input. For such a technique it does not matter, if it has to process all model pairs at once, or if the technique is applied to each model pair separately. In both cases the alignment for a specific model pair will be the same. Moreover, it might rely on a set of features that contains parameters

to configure the technique and resources that are used to integrate external knowledge. An example for such a technique is BOT which is introduced in Chapter 4.

Second, a matching technique could be designed to process a whole model collection or parts of it at once. In this regard, analyzing characteristics of the process models or pairs could be used for learning. That is, through the inspection of the model collection handed over to the technique, it can derive knowledge that is utilized to determine the alignments. OPBOT that is outlined in Chapter 5 falls into this category.

Thirdly, a matching technique might incorporate expert feedback in terms of manually identified alignments as discussed in the context of schema matching in [139]. There are two basic options in this regard. First, experts might provide complete alignments for a set of process model pairs. A matching technique might analyze these alignments and use the results to determine correspondences for the current process model pairs. Second, the experts might provide incomplete alignments that the matching technique has to complete. This strategy might especially be of interest for model pairs with rather large process models in order to ease the correspondence identification through a stepwise approach. Of course, both options can be combined. ADBOT which is discussed in Chapter 6 constitutes a feedback-based matching technique.

Generally, matching techniques can be implemented through the composition of matching techniques in a *matching workflow*. The general workflow for pairwise schema matching proposed in [140] can be adapted in this regard. Although, it considers the matching of two models, its basic structure can also be used for techniques that process model collections and alignments. Its basic structure is outlined in Figure 3.2

In the general matching workflow there are four different components. First, the *pre-processing* component is used to load and prepare the models for matching. In this regard, there might be a series of possible actions, e.g., the transformation of the models into a canonical format. Afterwards, the models are passed to the *matching sub-workflow* whose task is to suggest correspondences. It is a composition of matching techniques

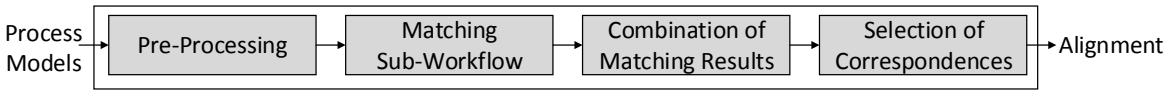

FIGURE 3.2: General business process model matching workflow, adopted from [140, p. 7]

and its general layout adheres to one of the three types shown in Figure 3.3. The *sequential matching sub-workflow* executes a number of matching techniques stepwise. Here, matching techniques might be used to reduce the search space. This includes filtering of non-corresponding activity pairs as well as marking definite correspondences. The *parallel matching sub-workflow* concurrently executes a set of matching techniques. Here, different strategies could be applied that suggest alignments based on different criteria. It is also possible to combine both types into a *mixed strategy*. Once the matching sub-workflow has terminated, a single alignment must be derived from the set of alignments proposed by the sub-workflow. Therefore, the next step is the *combination of matching results*. This, for example, includes the combination of similarity scores through aggregation or by determining the maximum. Lastly, the final alignment is created through the *selection of correspondences*. At this point, activity pairs are classified as corresponding if they are considered similar with regard to the previous results.

The purpose of the application of matching techniques is to ease the work for experts. Consequently, the most important quality dimension of matchers is their effectiveness which characterizes the degree to which a matcher resembles the opinion of experts, i.e., how many mistakes a matcher makes from the perspective of an expert. A further quality criterion is the efficiency of a matching technique [140]. It refers to the time and space complexity of a matcher. However, in this thesis only the effectiveness of matching techniques will be examined. The reason is that the effectiveness is an inevitable prerequisite for practical applicability whereas efficiency is a subordinate feature as illustrated by the following examples. The first example refers to a matching technique that identifies six

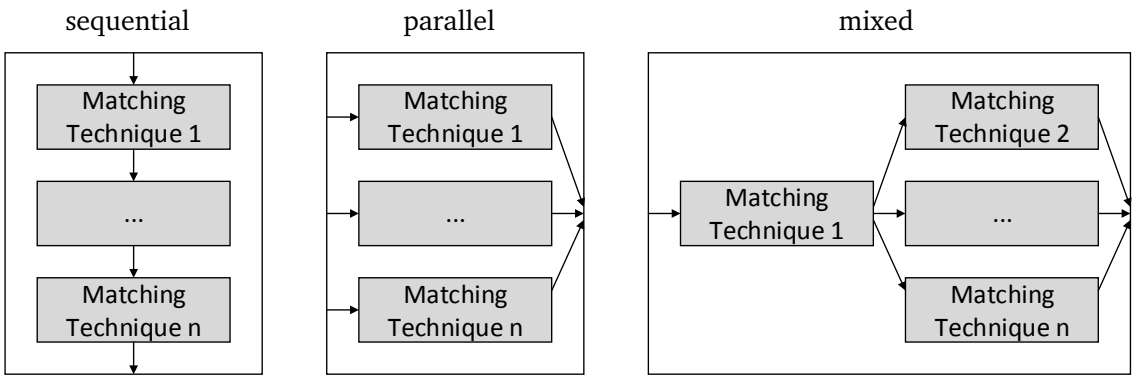

FIGURE 3.3: Basic matching sub-workflows, adopted from [140, p. 7]

correspondences for a given process model, but only three of these correspondences are actually correct. Additionally, there exist another five correspondences that were not identified. In this case, an expert would need to carry out eight operations to correct the alignment. That is, the expert needs to remove the three falsely suggested correspondences and add the five missing correspondences. Given that the expert only needed to add six correspondences at the beginning, the low effectiveness of the matching technique might even increase the workload for experts compared to not using a matching technique. The second example is given by the work of La Rosa et al. [141] who report that three analysts needed 130 man-hours to merge 25% of two process models. As the identification of correspondences is a central task in such projects [141], this example illustrates the enormous efforts linked to the manual matching of process models. The long period of time needed to manually match business process models in contrast to the negative impact on the workload for experts that ineffective techniques have, substantiates the decision to focus on the effectiveness. In the remainder of this work, the term quality is synonymously used for effectiveness.

In order to estimate the effectiveness of a matching technique, it is usually assessed with regard to a number of datasets. This approach was already briefly discussed in Section 1.3 where the effectiveness assessment as part of the research methodology was explained. Basically, it works by applying the matching technique to a set of model pairs for which gold standard alignments exist. Such alignments need to be determined by experts upfront. Given an alignment suggested by the matching technique and an according alignment from the gold standard, each activity pair in the respective model pair is assigned to one of four sets. The *true positives* (TP) comprise all correctly identified correspondences and the *false positives* (FP) all correspondences that were suggested by the technique, but are not among the gold standard correspondences. The *true negatives* (TN) and the *false negatives* (FN) are defined analogously with regard to the activity pairs that were suggested as non-corresponding by the technique.

Definition 3.5 (Activity pair classification) *Let $P = (N, A, E, \lambda, \tau)$ and $P' = (N', A', E', \lambda', \tau')$ be two process models. Further, let $\mathscr{A}^{gs} \subseteq A \times A'$ be the gold standard alignment and $\mathscr{A}^{mt} \subseteq A \times A'$ be an alignment proposed by a matching technique. Then, the sets of true positives TP, false positives FP, false negatives FN, and true negatives TN are defined as*

$$
\begin{aligned}
TP &= \{c | c \in \mathscr{A}^{mt} \wedge c \in \mathscr{A}^{gs} \wedge c \in A \times A'\} \\
FP &= \{c | c \in \mathscr{A}^{mt} \wedge c \notin \mathscr{A}^{gs} \wedge c \in A \times A'\} \\
FN &= \{c | c \notin \mathscr{A}^{mt} \wedge c \in \mathscr{A}^{gs} \wedge c \in A \times A'\} \\
TN &= \{c | c \notin \mathscr{A}^{mt} \wedge c \notin \mathscr{A}^{gs} \wedge c \in A \times A'\}
\end{aligned}
$$

Given this classification, the following indicators for the effectiveness of the technique can be defined. The *precision* provides information on the degree to which a matching technique proposes correspondences that do not exist. In other words, the higher the precision of a technique is the less non-existing correspondences it proposes. The *recall* characterizes the degree to which a matching technique detects correspondences, i.e., the higher the recall the more correspondences that actually exist are proposed by the technique. Finally, the *f-measure* is the harmonic mean of both values. These measures are widely adapted in schema matching [28, 29], information retrieval [30, 47], ontology matching [26, 27, 48], and business process model matching [19, 20].

These measures can be determined in different ways. First, they can be computed for a single pair of process models. Second, a whole dataset that contains several model pairs can be considered. In such situations, the measures can be defined on the macro (M) and the micro level (μ). On the macro level the measures are computed for each model pair in the collection separately. The overall effectiveness scores are then yielded by averaging the model pair scores. On the micro level the sets of true positives are combined and so are the sets of false positives, false negatives and true negatives. Then, the measures are determined with regard to the resulting sets.

Definition 3.6 (Effectiveness measures) *Let* $(\mathscr{A}_i^{mt})_{i=1}^k$ *be a sequence of* $k \in \mathbb{N}$ *alignments proposed by a matching technique and* $(\mathscr{A}_i^{gs})_{i=1}^k$ *be the respective sequence of gold standard alignments where it is required that the ith alignments in both sequences were determined for the same process model pair, i.e.,* $\forall 1 \leq i \leq k : dom(\mathscr{A}_i^{mt}) = dom(\mathscr{A}_i^{gs}) \wedge cod(\mathscr{A}_i^{mt}) = cod(\mathscr{A}_i^{gs})$. *Furthermore, let* $(TP_i)_{i=1}^k$ *denote the according sequence of true positives,* $(FP_i)_{i=1}^k$ *the sequence of false positives, and* $(FN_i)_{i=1}^k$ *the sequence of false negatives. Then, the precision* pr*, the recall* re *and the f-measure* F *can be defined in the following ways:*

$$pr_i = \frac{|TP_i|}{|TP_i|+|FP_i|} \qquad re_i = \frac{|TP_i|}{|TP_i|+|FN_i|} \qquad F_i = 2 \cdot \frac{pr_i \cdot re_i}{pr_i + re_i}$$

$$pr_M = \frac{1}{k}\sum_{j=1}^{k} pr_i \qquad re_M = \frac{1}{k}\sum_{j=1}^{k} re_i \qquad F_M = \frac{1}{k}\sum_{j=1}^{k} F_i$$

$$pr_\mu = \frac{\sum_{j=1}^{k}|TP_i|}{\sum_{j=1}^{k}(|TP_i|+|FP_i|)} \qquad re_\mu = \frac{\sum_{j=1}^{k}|TP_i|}{\sum_{j=1}^{k}(|TP_i|+|FN_i|)} \qquad F_\mu = \frac{pr_\mu \cdot re_\mu}{pr_\mu + re_\mu}$$

where a precision score is set to 1*, if there are no proposed correspondences, i.e., if* $|TP| + |FP| = 0$. *Accordinlgy, a recall score is set to* 1*, if there are no truly corresponding activity pairs, i.e., if* $|TP| + |FN| = 0$.

All measures are bound to the interval $[0,1]$ and the higher the value for a certain effectiveness measure is, the higher is the respective effectiveness dimension of the matcher. In this regard, the f-measure constitutes the most interesting measure because it provides an indication for the overall effectiveness. But, as the same f-measure value might be attributed to different combinations of precision and recall values, these measures are important secondary sources for the effectiveness assessment. In such cases, the recall should be favoured over the precision [142].

In this thesis, the most important set of measures are the micro level measures. The reason is that they characterize the effectiveness with regard to a whole model collection. Moreover, the macro level measures might be distorted in case of a large variance in the number of correspondences per model pair. However, techniques from related work will be considered as a baseline for the matching techniques proposed in this thesis. As for

some techniques there are only macro level measures published, these measures will be reported where such a comparison is carried out.

3.2 Application Scenarios

After having discussed basic concepts regarding business process model matching in the previous section, this section focuses on the demand for such techniques in practice. By reviewing tasks arising from the BPM lifecycle which require the identification of correspondences between business process models, the practical need for matching techniques is demonstrated and the first part of sub-hypothesis H1 is verified. Moreover, the section once more motivates the research in this thesis and also explicates research areas that depend on the results of this thesis.

A first set of such tasks refers to the management of business process model collections. In practice, organizations possess model collections that comprise hundreds or thousands of process models [143]. For example, China railway has to maintain 200,000 business process models [6] and SAP has more than 5,500 best practices process models [7]. Clearly, the large number of process models makes the manual management of such collections cumbersome. Especially, managing versions of processes and avoiding duplicates on activity as well as on process level are central challenges. In this regard, there is a variety of techniques that address the detection and handling of similar (sub-)processes including *process similarity search*, *process model merging*, *clone detection*, and *process model querying*. Often, these approaches assume that correspondences or alignments between process models are available and thus require the application of process model matching techniques. In the following, each of these areas is briefly introduced.

In case there exist similar process models with only a small number of differences it might be necessary to merge these models. A reason therefore is the reduction of the number of models referring to the same process in order to ease the management of these models. In this regard, process model merging techniques, e.g., [17, 141, 144], support the combination of different models. They automatically join two or more process models

and ensure that the original control flow constraints from the various models are captured in the unified model.

The field of process similarity search provides metrics or measures that indicate to which degree process models are similar. These techniques help to identify and cluster versions of processes or processes that overlap. Representative works in this areas comprise [117, 145]. An overview of existing techniques is provided in [146, 147].

Approaches for clone detection, e.g., those introduced in [148–150], aim at detecting equivalent or very similar fragments in process model collections. Such techniques can be used for refactoring. In particular, they are applied to introduce sub-process hierarchies and to maintain frequently occurring fragments in separate process models.

Closely related to the clone detection techniques are process model querying approaches [15, 151, 152]. Here, queries are formulated by a user in order to retrieve processes or process fragments that satisfy these queries. The difference to clone detection algorithms is that the queries do not necessarily need to be formulated using a process modeling language. Instead a query language might be used. Such languages allow to define attributes and relations of activities that process models must satisfy, e.g., that only one of two activities should be present or that the execution of a certain activity must be followed by the execution of another activity.

Another use case for process model matching techniques is the support for modelers in the design of process models. Here, a modeler is pointed to activities or fragments that share characteristics with the currently designed model. Based on these suggestions, the modeler can orient the layout of the new process model towards existing designs. Whereas the techniques referring to the management of modeling collections analyze existing collections, modeling support aims to reuse knowledge from collections and to ensure consistent modeling from the beginning. Awad et al. [14] and Chan et al. [153] introduce approaches that support modelers by presenting alternative modeling options and giving the modeler the chance to select an option. Additionally, Niedermann et al. [154] present the idea of taking existing process analysis results during modeling into account. Therefore, correspondences between the created model and existing models are

used to apply insights from past process executions to the new model. This way process optimization can already be conducted during design time.

A further area that benefits from process model matching techniques is compliance checking. It deals with determining if a process model adheres to rules, e.g., regulatory guidelines induced by the law, or internal standards implemented by a company. In case these rules exist as reference process models, matching techniques can help to check whether all necessary activities are implemented. Process similarity metrics could then be used to validate that control flow constraints are also met. This method could be applied in a variety of scenarios, e.g., when processes of service providers must be compared to customer requirements [49] in the context of service management platforms [155]. Similarly, this method can be of use for the evaluation of standard software [156] where the processes of the software packages are compared to the processes in a company [157]. Moreover, Branco et al. [16] apply business process model matching in the context of model-driven engineering [158, 159]. Their goal is to verify that a software process is compliant to the higher level business process defined in an early phase of the software development project.

There also exist compliance checking algorithms that support scenarios where the regulatory rules are not represented as process models. Instead, they are encoded as logical rules which might be derived from legislative texts or other sources. These rules contain relations between possible states of the process execution. To utilize these rules, the process models must be annotated with the respective states. Then, algorithms are applied to verify whether the constraints imposed by a rule are satisfied in a process model. According techniques include [87–89]. As the annotation of process models requires human effort, correspondences could be used to transfer annotations between models.

The last use case scenario for business process model matching considered here is the consolidation of business processes. It is seen as one of the central use cases for matching [116, 160–162]. In case two companies merge and want to unify their business processes, process model matching techniques help to identify the similarities and differences between process models. In this regard, a further use case is inductive

reference process modeling [18, 163, 164] where reference process models are derived from a set of existing process models.

These use cases illustrate the broad variety of application scenarios for business process model matching techniques in practice. Despite this demand, the support for these use cases that professional process modeling tools offer is insufficient. Tools like Signavio[8], or ARIS[9] do not employ business process model matching techniques, but determine correspondences based on equal attributes, mainly labels and identifiers [116]. As will be discussed in the next section this is clearly not sufficient as process models from practice are often characterized by textual and structural heterogeneity. In summary, the use cases as well as the insufficient support offered by professional tools illustrate the practical need for process model matching techniques.

3.3 State of the Art

The second part of sub-hypothesis H1, the scientific demand, is subject to this section. Its verification is based on a critical review of the current state of the art on business process model matching. Therefore, the particular questions that the review aims to answer are introduced in Section 3.3.1. Next, the search strategy that was applied to identify relevant literature is explained in Section 3.3.2. Then, the identified literature is briefly summarized in Section 3.3.3. Finally, Section 3.3.4 discusses the state of the art based on the questions and identifies the research gap.

3.3.1 Questions

The examination of the state of the art is guided by four questions that view the related work from two angles. First, there are two questions that focus on the matching techniques as the research artifact. To this end, the first question refers to the *applicability* and examines, if the application of matching techniques is restricted to certain matching scenarios.

[8] http://www.signavio.com/, accessed: 13/01/2017

[9] http://www.softwareag.com/de/products/aris_alfabet/default.asp, accessed: 13/01/2017

Additionally, the second question focuses on the evidence that was given towards the *effectiveness* of the matching techniques. In contrast to these two questions, the other two questions investigate the research underlying the development of the techniques. Here, the third question is related to the type of the research *approach* and the fourth question to the *empiricism* in terms of the amount of empirical data the research relies on. All questions are outlined in the following.

Q1 - Applicability: *Is a broad applicability of the matching techniques ensured?* In order to be applicable in as many matching scenarios as possible, the techniques should not pose any restrictions on the process models. The restrictions can refer to a diverse range of characteristics. The following requirements refer to the most important characteristics.

First, there are different modeling languages available (cf. Section 2.3). Consequently, matching techniques must be able to process a diverse range of such languages.

Second, not all modeling languages provide means to capture the informational and organizational perspectives of a business process, e.g., in their basic form EPC and Petri nets do not provide such elements. Even if a language provides appropriate elements it cannot be assumed that they are also used. For example, zur Muehlen et al. and Recker [122] observe that only a small subset of BPMN elements is used in practice with a strong emphasis on functional and behavioral perspective (cf. Section 2.3). Consequently, matching techniques have to be able to compute alignments by only exploiting the activity descriptions and the control flow.

Third, while the labels within a model collection are expected to rely on the same natural language, it cannot be assumed that the process model elements are labeled homogeneously, i.e., with equal labels. This is supported by various empirical observations. Mendling et al. [38] and Leopold [39] observed that labeling styles vary within model collections. Similarly, Pittke et al. [165] revealed that control flow constraints might be encoded in labels rather than being expressed with the according modeling language elements. Weber et al. [166] observed that there are activities with similar purposes, but different labels. Finally, Gottschalk et al. [144] were challenged by the versatile labeling of similar activities when consolidating a set of process models from Dutch municipalities.

Fourth, matchers cannot expect models to be *sound*. Basically, a model is considered to be sound, if for each state during the execution of the process, it is possible to terminate the process [91]. Typical errors that lead to unsound models include deadlocks and livelocks. Again, this requirement is based on various empirical evidence. Fahland et al. [90] report that 49% of 735 models that stem from three IBM libraries are unsound and Mendling [64] discovered that 21% of the 604 EPC models from the SAP reference model are unsound. An extensive overview on quality aspects of business process models that further substantiates the last two requirements can be found in [167].

Lastly, matching techniques must be aware of a varying granularity in process models. Such differences usually result in complex correspondences. In this regard, Dijkman [12, 13] observed differences in the granularity where a set of activities in one process model implements the same or overlapping functionality as a single activity or a set of activities from another model.

Based on these requirements the applicability of a matching technique is estimated to be *high*, if it implements all requirements. Similarly, a matcher is considered to have a *medium* applicability, if one or two of the requirements are violated. In all other cases the applicability is classified as *low*.

Q2 - Effectiveness: *How effective are the matching techniques?* In contrast to the first question which refers to the input of the matching techniques, the second question is related to the output in terms of the effectiveness. In particular, the effectiveness is assessed by investigating the empirical evidence from the literature. That is, the reported effectiveness measures (Section 3.1, Definition 3.6) and especially the f-measure are considered. In cases where only precision and recall values are reported, the f-measure is computed based on these values.

Based on the f-measure, the matchers' effectiveness is classified as *high* ($F \in (.\overline{6}, 1]$), *medium* ($F \in (.\overline{3}, .\overline{6}]$), or *low* ($F \in (0, .\overline{3}]$). This rough classification was chosen to provide an overview of the results achieved in the process matching literature. More details are provided in the context of the summary of the identified techniques and in the respective publications. Note that in cases where a technique was evaluated in multiple publications,

it might be assigned to a range of effectiveness classifications. Moreover, if no evaluation results are presented for a matcher, its effectiveness was *not assessed*.

Q3 - Approach: *What are the limitations of the applied research approaches?* The goal of this question is to examine whether there are any threats that limit the validity of the existing research results. To this end, the research approaches applied in the literature are classified and potential shortcomings are discussed. In this regard, the results of this analysis substantiate the research approach underlying this thesis.

In particular, this thesis distinguishes between five categories of approaches. First, the *proposition* approach characterizes literature that introduces a design for a matching technique, but does not present any empirical evidence that supports the underlying assumptions or that demonstrates the technique's effectiveness. Clearly, such works do not allow to assess the utility of the design choices. Second, papers might present an *illustration* of the proposed technique. While this approach enables readers to comprehend the intended use and functioning of the technique, there is still no quantifiable evidence that supports the technique's design. Third, the *evaluation* of a technique provides information regarding the effectiveness of the technique. Here, a technique is applied to a set of model collections, its suggestions are compared to a predefined gold standard, and respective effectiveness measures are reported. In contrast to the first two categories, papers that follow this approach provide quantifiable evidence. Yet, the limitation of this approach is that there is no baseline that allows for assessing whether the achieved effectiveness can be considered to advance existing techniques or not. Fourth, papers might comprise a *comparison* of different techniques to overcome the weaknesses of the evaluation approach and to put the achieved effectiveness into context. Here, one or more techniques from prior work serve as a baseline and the comparison of the effectiveness measures allows for a more detailed assessment of the contributions of the introduced matcher. Yet, as discussed in [168, 169], such comparisons need to be treated with care, as the results might have been observed by chance. Fifth, the thesis thus also distinguishes *analysis* approaches where an assessment of the basic cause-and-effect relationships is conducted. By going beyond the pure evaluation of matchers and studying assumptions and design

options, such an approach provides the most solid foundation for future improvements. That is because it makes the utility of the design choices explicit and hence fosters the reuse of them in future work.

Q4 - Empiricism: *How many model pairs are typically used in the evaluation?* In addition to the third question, the fourth question addresses the size of the empirical data used in the literature. To this end, the number of process model pairs is used as an indicator for the size of the empirical data. The purpose of this question is twofold. On the one hand, it helps to assess the external validity of the proposed approaches, i.e., to which degree the achieved effectiveness can be generalized. Clearly, the higher the amount of the model pairs, the more generalizable the results are. On the other hand, it puts the extent of empirical data that this thesis relies on into perspective and justifies the choice of four real-world datasets as the empirical base for this thesis.

3.3.2 Search Strategy

Business process model matching is closely related to the area of ontology and schema matching where techniques for the comparison of database schemas and ontologies are developed. An overview of schema and ontology matching algorithms is provided in [119, 120, 170, 171]. Basic concepts and techniques are summarized in [138, 172]. Although the research in the context of schema and ontology matching provides a valuable pool of concepts for process model matching, respective approaches are excluded from the literature review. The reason is that in comparison to process models ontologies and database schemas are characterized by different types of textual and structural information. For example, entities in database schemas are usually labeled with terms, e.g., there might be tables like ‘sales order’ and attributes like ‘order item’. Furthermore, typical relations in database schemata include generalizations or aggregations. In contrast, activity labels in process models contain phrases consisting of several words that describe an action and the relations between these activities refer to temporal dependencies. Consequently, Dijkman [116] observed that the quality of process model alignments yielded by the similarity

flooding algorithm for schema matching [173] is poor. Additionally, the AML ontology matcher [174] with good results in comparative ontology matching evaluations [26, 174] yielded poor results on two out of three datasets in the process model matching contest of 2015 [20].

Consequently, the literature search was limited to the field of process model matching and its goal was to provide a representative collection of techniques from this field. This also means that research areas in the field of BPM that are related to process model matching, e.g., process similarity search, process querying and clone detection (cf. Section 3.2), were excluded from the search. The reason is that techniques in this field aim to measure similarity at the process or fragment level rather than at activity level. Thus, they do not necessarily determine an alignment between process models. With that in mind, the focus was on related work where a process model matching technique is introduced.

To identify such related work, the search strategy suggested by vom Brocke et al. [32] (cf. Section 1.3) was adapted here. The basic structure of this strategy is to carry out a journal search and refine it by a database search which finally is completed by a backward and forward search. Here, the journal search was skipped and replaced by an analysis of the two process model matching contests from 2013 and 2015 [19, 20] as the contests were considered to be representative of the state of the art. The contests were carried out in the context of the International Conference on Business Process Management and invited researchers to submit their matching techniques. Therefore, the researchers were required to provide a short description of their matcher as well as matching results for various model collections. A brief overview of the two editions is provided in the next section. In addition to the respective publications of the contest results in [19, 20], another six papers that dealt with matching techniques were derived from the respective reference lists.

Based on the eight publications that were identified so far a database search was prepared by deriving search terms to query the databases. Basically, each of these terms consisted of two terms. The first term referred to process models as the central artifact. In particular, this comprised *"process model"* and *"business process"*. Here, the quotes indicate that both words in the term needed to occur in the paper. The second term was related

to matching including the terms *match**, *map**, and *align**. The asterisk indicates that different declinations are included, e.g., "match", "matches", "matching" or "matched" satisfy the search term "match*". As a result there were six search strings ("process model" match*; "process model" map*; "process model" align*; "business process" match*; "business process" map*; "business process" align*) which were seperately used to query the databases. A further query constraint referred to the publication date of the papers. In this regard, only papers that were published between 2000 and May 2016 were considered. The latter date corresponds to the time at which the literature search was carried out. The former was chosen, because modern BPM together with an increased usage of process models arose at beginning of the 2000s [2].

To finalize the preparation of the database search, relevant databases were selected. The focus was on databases that contain papers that are written in English, peer-reviewed, and were published in well-known journals and conferences from the IS-domain. Thus, the following databases were chosen: the IEEE Xplore Digital Library[10], the ACM digital library[11], Science Direct[12], Springer Link[13], Google Scholar[14] and Emerald Insight[15].

During the database search the result lists for each database and search string combination were examined in order to assess the relevance of the proposed papers and to exclude papers that were out of scope. Therefore, the titles and the abstracts were scanned. In most of the cases, the result lists comprised hundreds or thousands of papers where papers with a low position were likely to be irrelevant. Thus, the result lists were scanned stepwise starting from the highest position. For each result list the first 50 papers were examined. Afterwards, papers were scanned until the distance to the last relevant paper exceeded 20. Once all result lists were filtered, duplicates were removed from the identified set of papers. At this point, the eight initial papers were used to validate the database search. That is, it was checked, if the results contained these eight papers. Except for the publication of the results of the second matching contest [20] all papers

[10] http://ieeexplore.ieee.org/Xplore/home.jsp, accessed: 13/01/2017
[11] http://dl.acm.org, accessed: 13/01/2017
[12] http://www.sciencedirect.com, accessed: 13/01/2017
[13] http://link.springer.com, accessed: 13/01/2017
[14] http://scholar.google.com, accessed: 13/01/2017
[15] http://www.emeraldinsight.com, accessed: 13/01/2017

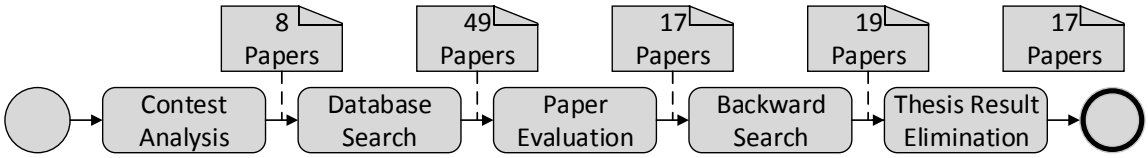

FIGURE 3.4: Overview of the search process and the identified papers

were present. The reason for the absence of this paper is that it was published in the Lecture Notes in Informatics by the German society for computer science (Gesellschaft für Informatik) which were not indexed by any of the databases at this time. However, this result was considered to verify our strategy. Nevertheless, the completeness of the search is limited by the completeness of the employed databases. Thus, to mitigate the risk of overlooking papers, the database search was finally complemented by a backward search over the references in the identified papers.

All identified papers were evaluated and only those papers that were relevant with respect to the questions were selected. Here, the inclusion criterion was that the papers introduced a process model matcher. In contrast, papers that (i) discussed process model matching (e.g., [175, 176]) ; (ii) addressed aspects of model collection management (e.g., [177–180]) ; (iii) discussed support for process model design (e.g., [153, 181]); or (iv) referred to Business-IT alignment ([182]) were excluded.

The final set of relevant literature contained 19 papers where two of the papers [50, 51] were co-authored by the author of the thesis. As these two papers include results that are discussed in this thesis, they were removed. Thus, the literature search revealed a total of 17 papers. The search is summarized in Figure 3.4 and an overview of all identified papers is provided in Appendix A.

3.3.3 Matching Techniques

Next, all techniques that were identified during the literature review are briefly summarized. To ensure that relations between these techniques are comprehensible, they are presented in historical order.

Semantic Alignment of Business Processes [160] A generic approach to identify elementary correspondences between elements of Pr/T nets [183] which are a specialization

of Petri nets is proposed in [160]. Besides correspondences between transitions, the approach also suggests correspondences between other elements. The authors later adapted the approach for process similarity search [145]. In the approach the types of elements that should be matched and the relevant properties of these elements are determined first. Then, for all possible property pairs similarity scores are calculated based on manually created ontologies and aggregated in order to yield a similarity score per element pair. Based on the scores, corresponding element pairs are selected and another iteration might be triggered to refine the results. The approach is not evaluated.

Matching Statecharts Specifications [161] A matcher that is tailored to dialects of statecharts [184] and especially to ECharts [185] is presented in [161]. The approach focuses on matching states. Transitions are not matched, but analyzed during the matching process. The authors propose two types of sub-matchers. Static matchers investigate labels and positions of states. Behavioral matchers examine whether two states depend on or transition into similar states. In an evaluation based on three statechart pairs it is shown that the approach achieves recall values between .81 and 1.0 as well as precision values between .51 and .55.

Aligning Business Process Models [116] A configurable matching technique is introduced in [116]. To measure the similarity of activities, a syntactic measure is applied to the labels. Additionally, the labels can be harmonized before the score is calculated. Based on the scores there are two ways to determine an alignment. First, activity pairs that have a label similarity score higher than a threshold are classified as a correspondence. Second, alignments can be identified by optimizing an overall similarity score for the process models which is also used for similarity search in [146]. This score is based on the graph edit distance [186] and besides the label similarity score also takes the number of matched activities and edges into account. To construct an alignment, correspondences are added to the empty alignment as long as the score can be improved. Therefore, a greedy strategy and the A-star heuristic [187, 188] are used. Finally, a post-process step can be activated in order to also yield complex correspondences by extending elementary correspondences. Different variants are evaluated on a dataset that comprises 17 model

pairs from Dutch municipalities. Here, the macro f-measures of the variants range in between .66 and .72.

Complex Mapping Discovery [189] Another approach that relies on a graph edit distance is proposed in [189]. As the approach is applied for Web Service retrieval, the identification of correspondences relies on the comparison of the labels as well as of annotated input and output objects. First, all activity pairs with compatible input and output are determined and ranked with regard to a weighted similarity score that is based on the labels and the input and output objects. From this set of potential elementary correspondences a set of 1:n-correspondence candidates is derived. Here, elementary correspondences are composed, if the activities from the n-side occur in the same sequence, parallel or exclusive block, and if the similarity score of the combination is sufficiently high. Finally, all candidates are considered to construct an alignment that maximizes a graph edit distance. The matcher is not evaluated.

The ICoP framework [162] The *ICoP framework* [162] provides components for the detection of elementary and 1:n-correspondences. Like the general matching workflow [140], it constitutes a configurable and extendable infrastructure that allows users to select components according to their needs. *Searchers* constitute the first set of components that are used to identify potential correspondences by applying similarity measures and heuristics. The result of the searcher execution is a multi-set of correspondences from which a set of correspondences can be constructed through the application of *boosters*. In this regard, correspondences are removed or aggregated and similarity scores are adapted. Finally, *selectors* construct an alignment by selecting correspondences from the set. Therefore, a selector can either rely on the determined similarity scores or on an *evaluator*. Evaluators calculate overall scores for potential alignments and might rely on properties derived from the process models. The ICoP framework also provides a couple of implementations for all of these components. Different configurations of the framework are evaluated using the 17 model pairs from [116] as well as three additional model pairs. Moreover, they are compared to a variant of the matcher from [116]. The f-measures for all matchers differ only marginally and are located close to a value of .6. Weidlich et al.

[162] observed low f-measures of about .3 for the three new model pairs. Due to these model pairs, the overall f-measures were lower than those reported in [116].

Summary-Based Process Model Matching [190] A refined version of their previous work [189] is discussed by Gater et al. [190]. The matcher also considers input and output objects that are annotated to the activities. It first summarizes process models, by composing parallel and alternative blocks as well as sequences into a single activity and deriving a label as well as input and output annotations from the activities in the block. Then, the summarized versions of the process models are matched using a graph edit distance approach. The alignment is then refined by adding unmatched activities to correspondences in their neighborhoods. Moreover, m:n-correspondences are broken down to elementary and 1:n-correspondences. Based on an evaluation of 1,200 model pairs, an overall f-measure of .84 is reported.

Precise Mappings in Versioning Scenarios [191] The matching technique from [191, 192] determines correspondences between activities as well as between edges and fragments. The approach supports a specific scenario where an original and two of its versions are compared. The basic idea is to establish an alignment whenever a new version of a model is created by copying the original and to automatically update it whenever changes are made to the new version. As the updates might not cover all changes, missing correspondences are identified automatically. First, the labels of all activity pairs are compared to identify missing correspondences between activities. Then, missing correspondences between edges are identified by checking whether there are edges with corresponding sources and targets. Finally, fragments are derived through a structural decomposition of the models. To compare two fragments Gerth et al. [191] rely on descriptions that combine the labels within the fragment as introduced in [193]. Alignments between versions of the same original are initially inferred from the alignments between the versions and the original. Then, they are completed by applying the same procedure for completing alignments between the original and its versions. Due to a missing evaluation, the effectiveness requirements cannot be assessed.

Matching Processes Across Abstraction Layers [16] The matcher from [16] is designed to support compliance checks in model driven engineering projects where abstract process models are compared to refined and more fine-grained models. The approach first classifies all node pairs with equal labels and types as elementary correspondences. As the scenario suggests the existence of many complex correspondences, a structural decompositions of the process model in terms of fragment hierarchies are used to identify complex correspondences. For each fragment the labels of its activities are combined and a syntactic label similarity measure is used to compare the combined elements. Correspondences in the fragment hierarchies are identified by a top-down traversal. Branco et al. [16] present an evaluation of the approach based on 110 model pairs from the Bank of Northeast Brazil. The overall macro f-measure is .81. However, while the approach detects 400 of the 416 elementary correspondences, it only identifies 38 out of the 222 complex correspondences.

Semantic Process Model Matching [194] Labels typically contain different components that describe the action and the business object or that provide additional information. Accordingly, Leopold et al. [194] compare labels based on the components that they derive by applying their own component detection algorithm [195]. Based on a weighted component similarity score, they use Markov logic networks [196] to construct alignments. In this regard, they further define a set of constraints that alignments must satisfy. These constraints refer to the inclusion of complex correspondences as well as to the consistency of behavioral dependencies. The evaluation is based on 36 model pairs and a configuration of the ICoP framework serves as a baseline. Here, the approach by Leopold et al. [194] slightly outperforms the ICoP framework in terms of the f-measure (.318 vs. .294).

The Prediction of Matching Quality [197] A flexible approach to process model matching is discussed in [197]. The basic idea is to select the most promising matcher for a given process model pair. Therefore, a prediction model is trained on a set of known alignments. This model identifies correlations between the effectiveness of matchers and characteristics of process models as well as of activities. Once the prediction model was learned, it can be used to identify matchers with a high effectiveness for a given

process model. While a set of measures to assess characteristics of models and activities is introduced, the framework does not comprise any specific matching techniques and was not evaluated.

Matching Based on Positional Language Models [198] Process models often do not exist in isolation, but are accompanied by documentations. Accordingly, Weidlich et al. [198] propose a technique that allows for integrating additional documents into the matching process. The matcher first derives a document for each process model. Therefore, it traverses a structural decomposition of the model to transform it into a sequence of activities. Then, each activity is transformed into a passage that contains the label and additional documentation, if it exists. Afterwards, similarity scores are computed for each activity pair by comparing the respective passages. Therefore, the probability of terms to occur in the passages is determined based on the work by Lv and Zhai [199]. Then, a similarity score is computed using these probabilities by applying the approach from [200]. Finally, the alignment is identified by selecting the most similar activity pairs. The evaluation comprises four different sets of model pairs from [16] and [162]. The f-measure varies between .18 and .33 on these sets.

The Process Model Matching Contest 2013 [19] Besides the approaches from [116, 162, 197] and a technique developed by the author of this thesis (cf. Chapter 4) three additional approaches that were not published anywhere else were submitted to the contest. The *Triple-S* technique measures the similarity of activities based on their labels as well as the number of incoming and outgoing edges. All activity pairs whose respective similarity score is higher than a predefined threshold are suggested as correspondences. The *RefMod-Mine/NSCM* (RMM/NSCM) technique is tailored towards Petri nets. In the first step, it identifies transitions in the Petri nets whose labels indicate that these transitions are actually places. The detection of such transitions is based on the analysis of the form and position of verbs and nouns in the labels. All identified transitions are ignored during matching. Contrary to other approaches from related work RMM/NSCM matches all models in a model collection at once. Therefore, all transitions are compared to each other and a single similarity score is computed for each transition pair. This score solely

relies on the transitions' labels. In particular, it splits the labels into sets of words and removes stop words. Next, it computes the stem for each word. Then, the score is the percentage of stems that occur in both labels. If the score is below a predefined threshold, it is set to 0. In all other cases it is checked whether the labels contain antonyms and thus can be considered totally disimilar. If there are antonyms, the score is set to 0. Based on the similarity scores for all transition pairs, hierarchical agglomerative clustering is applied to cluster the transitions. At the beginning of the clustering a cluster is created for each transition. Then, the clusters are stepwise merged by determining the two clusters containing the most similar transitions and merging these clusters. The clustering stops when the maximum score for a transition pair with two transitions from different clusters is 0. Finally, the identified clusters are translated to correspondences, i.e., each pair of transitions from the same cluster is considered a correspondence. The *RefMod-Mine/ESGM* (RMM/ESGM) also filters and harmonizes activities. The selection of correspondences follows the approach by [116], but exploits dictionary lookups and syntactical similarity measures to compare labels. Lastly, the alignment is completed by adding activity pairs with a similarity score higher than a predefined threshold to the alignment. In addition to the dataset from [194], the evaluation of the matchers comprised another set of 36 model pairs. All approaches yielded a low effectiveness with the highest f-measures at about .4.

Multi-Perspective Matching [201] Another variant of the approach by Dijkman et al. [116] is presented in [201]. The extension is intended to identify complex correspondences, but is limited to process models that represent sequences. In contrast to [116] the similarity score for two activities is not only computed with regard to the labels. Instead, it also considers the order of the activities in relation to all other correspondences, the ratio of data objects shared by the activities, and the roles responsible for the execution of the activities. The approach is not evaluated.

Semantic Model Alignment [202] Fengel [202] presents an approach to model matching that only relies on the labels. In a pre-processing step the process models are transformed into a common format which is based on the web ontology language [203]. Next, a label similarity score is determined for each activity pair. This

score considers equal labels, the number of shared words and synonyms, the existence of negation words (e.g., "not"), and a label based similarity. Given the similarity scores, each activity pair is classified as an exact, a close, a loose, or a low correspondence. Based on an evaluation on eight model pairs, a macro f-measure of .89 is reported.

Fast Discovery of Complex Matches [204] The next approach is again a variant of the approach by Dijkman et al. [116]. However, Ling et al. [204] do not match activities, but activity groups that are derived from structural decompositions of the process models. First, the set of activity groups is determined for each model. Then, a similarity score is computed for each pair of activity groups. In this regard, Ling et al. [204] do not explain how this score is computed. Each group pair gp for which there is no other group pair gp' that comprises subsets of the groups in gp and yields a higher similarity than gp is a potential correspondence. The alignment is then derived from the set of potential correspondences through the application of the greedy strategy introduced by Dijkman et al. [116]. Therefore, the overall alignment similarity is adapted to consider sets of substituted and of skipped activity groups as well as of skipped edges. Additionally, a further component is added to account for the corresponding edges within pairs of activity groups. The authors conduct an assessment of the effectiveness based on 20 model pairs. Here, the approach achieves an f-measure .73.

Resource-Aware Process Matching [205] Baumann et al. [205] refine their own work from [201]. In particular, they extend their approach through a more fine-grain assessment of the organizational perspective. Therefore, they introduce different approaches to compute an activity similarity score based on the roles assigned to the activities. This score distinguishes between human and non-human roles as well as resources. Although the authors discuss practical limitations of their approach, they do not provide any evaluation results.

The Process Model Matching Contest 2015 [20] The second edition of the matching contest concludes the presentation of matching techniques from prior research. In this edition of the contest twelve matchers were evaluated. Besides matchers from [19, 198] and the technique from Chapter 5 there were nine additional techniques which are briefly

summarized in the following. The *AML-PM* matcher is a based on the AML ontology matcher [174]. It compares two process models by first transforming the models into an ontological representation. AML-PM loads the ontological representations and preprocesses them. That is, it transforms the ontologies into an efficient data structure that comprises lexical, structural, and semantic information from the models. It also contains a lexicon that contains the labels and automatically derives synonyms. Next, AML-PM detects all activity pairs with the same labels and adds them to a set of potential correspondences. All remaining activity pairs are then processed by the string matcher which computes a string similarity score per pair. In this step, all pairs with a similarity score above a predefined threshold are added to the set of potential correspondences. In the final step, the word matcher is applied to all activity pairs that are not in the set of potential correspondences. This matcher is based on the Jaccard index and it is run twice: first without label modifications and in the second run all words are stemmed. Again, activities with scores above a threshold are added to the set of potential correspondences. In the last step, the alignment is constructed by ranking the potential correspondences in descending order with regard to the similarity scores. Then, the correspondences are stepwise added to the alignment. A correspondence is only added, if none of the two activities is already part of a selected correspondence. Thus, the matcher only suggests 1:1-correspondences. For each activity the *KnoMa-Proc* matcher extracts its neighboring activities and joins their and the activity's labels. It then uses the joined labels to determine correspondence candidates for each activity based on a non-specified approach. From the set of candidates an alignment is constructed by considering the confidence in the correspondences. The *Match-SSS* and the *Know-Match-SSS* techniques compute similarity scores based on the words in the labels and select correspondences with high scores. The approaches differ with regard to the applied word similarities. The *RefMod-Mine/VM*^{2} (RMM/VM^2) matcher first identifies all activity pairs with equal labels. Next, activity pairs with similar words in different orders are determined. Lastly, correspondences are added based on a label similarity score that utilizes statistics on the occurrence of words in the model pair. The *RefMod-Mine/NCHM* (RMM/NCHM) matcher is an extension of the RMM/NSCM technique from the first contest [19]. In contrast to the first version,

the RMM/NCHM incorporates a post-processing step to filter activity pairs with different roles. The *RefMod-Mine/NLM* (RMM/NLM) matcher computes label similarity scores based on word relations in a dictionary. It selects all activity pairs whose similarity score is considered to be high. The *RefMod-Mine/SMSL* (RMM/SMSL) is also based on the analysis of word relations. However, it optimizes the similarity scores based on gold standard alignments. Lastly, the *pPalm-DS* matcher also solely relies on labels. It computes a label similarity score for each activity pair in the two process models being matched. If the score of an activity pair is higher than or equal to a predefined threshold, it is proposed as a correspondence. In contrast to many other approaches pPalm-DS relies on corpora to compare the words in the labels. It relies on the Gigaword corpus and a collection of English Wikipedia articles. Both corpora are processed using the *word2vec*[16] library, in order to extract semantic context vectors with 300 elements for the words in the corpora. Based on the vectors a label similarity score is computed as follows. First, the labels are split and the sets of words are derived. Then, for each word the context vector is retrieved and per label the respective context vectors are summed up to yield the context vector for the label. Finally, the similarity score is the cosine similarity of the two label vectors. Similar to RefMod-Mine/VM^2, it computes label similarities based on word occurrences. To determine occurrence counts, it does not consider the process models, but Wikipedia[17]. There are three datasets used to evaluate the matchers. The best f-measure scores on each dataset rank in between 0.54 and 0.68.

3.3.4 Results

To identify the research gap, the identified literature is now examined. Therefore, each publication is characterized with regard to the four questions. In the following, the focus is first on the applicability (Q1) and the effectiveness (Q2) of the matching techniques. After that, the underlying research approaches (Q3) and the empiricism (Q4) are discussed. Table 3.2 summarizes the assessment of all identified publications.

[16] https://code.google.com/p/word2vec/
[17] https://en.wikipedia.org/, accessed: 13/01/2017

TABLE 3.2: Summarized assessment of the approaches from prior research

Source	*Q1 - Applicability*	*Q2 - Effectiveness*	*Q3 - Approach*	*Q4 - Empiricism*
[19]	high	low - medium	Evaluation	72
[20]	high	low - high	Evaluation	108
[160]	medium	not assessed	Illustration	
[161]	medium	medium - high	Comparison	3
[116]	high	low - medium	Comparison	17
[189]	medium	not assessed	Proposition	
[162]	high	medium - high	Comparison	20
[190]	medium	high	Comparison	1200
[191]	medium	not assessed	Proposition	
[16]	medium	high	Comparison	110
[194]	medium	low	Comparison	26
[197]	high	not assessed	Proposition	
[198]	high	medium	Comparison	130
[201]	medium	not assessed	Illustration	
[202]	high	high	Evaluation	8
[204]	high	high	Evaluation	20
[205]	medium	not assessed	Proposition	

That matchers need to be applicable in a broad variety of scenarios is widely acknowledged in the literature. That is because there are no restrictions imposed on the applicability in 8 of the 17 publications and thus the applicability of the respective matchers is considered to be high. Moreover, the remaining publications only introduce one or two restrictions, which still allows for a broad application of the matchers. Consequently, none of the approaches from the literature has a low applicability.

In contrast to the applicability, the effectiveness of the matchers is generally insufficient. Admittedly, there are results that give evidence towards a high effectiveness, e.g.,

the matchers in [16, 190, 202, 204] achieve a high effectiveness on the entire dataset. However, the according publications are typically characterized by a small size of the dataset as in [202, 204] or by restrictions regarding the applicability as in [16, 190]. In the rest of the publications that presented evaluation results, evidence is given that the effectiveness of the approaches varies including a low effectiveness on parts of the data. These observations suggest that the difficulty of the datasets varies and that further research to improve the effectiveness of matching techniques as postulated by the research hypothesis H1 is required.

A further shortcoming of the identified publications is related to the research approaches which can be assigned to one out of four classes. First, there are three papers that only *propose* matchers but provide no empirical evidence. Another two papers use one synthetic example to *illustrate* how the matcher is supposed to work. Papers falling into these classes do not provide any evidence towards the proposed ideas and concepts. Among the remaining twelve publications, there are three papers [19, 20, 204] that *evaluate* matchers as black boxes. Such an evaluation assesses the effectiveness of the entire matchers, but an analysis of the influence of the matchers' components is not conducted. Thus, the re-use of these matchers' components in the design of the more effective matchers is not enforced. For example, the matcher in [204] comprises components that compute label similarity scores, investigate the graph neighborhood, detect fragments, and check the consistency. Clearly, the reported overall effectiveness provides no insights into the contribution of each component. Similarly, the contests [19, 20] compare the effectiveness of various matchers, but not the influence of their respective components. Another seven papers *compare* the effectiveness of different matcher variants. However, as e.g., discussed in [168, 169] such results need to be interpreted with care and typically have a limited validity. That is because without further statistical analyses differences might have been observed simply by chance – especially as the reported difference are rather small, e.g., the f-measures in [116] differ by about .06 and in [162] by $\approx$ .05. Moreover, the results of all variants are typically dependent on a basic variant. This entails the risk that the relative performance of the variants and thus the contribution of the components changes, if the basic variant is modified. For example, the approaches in [194]

and [162] analyze structural relations between correspondences to assess the consistency of the correspondences. While in [194] this assessment improves the effectiveness of a basic variant, it reduces the quality in [162]. This shows that even those papers that compare variants are characterized by a limited validity regarding the contribution of the matchers' components. Finally, there is no paper that analyzes the results and techniques in more detail.

In summary, the majority of the works focuses on the evaluation of the matchers' effectiveness, but does not study whether the separate design decisions have a generally positive or negative effect on the identification of correspondences. Thus, the re-use of proposed concepts is not enforced. This observation also motivates the research approach in this thesis which explicitly incorporates the examination of matching propositions to understand which design decisions have the potential to improve the matching. Moreover, almost no publication distinguishes between data that is used to develop the matcher and data that is used to evaluate it. Yet, this is necessary to avoid fitting the matchers to the data [37]. Here, the matching contests [19, 20] constitute an exception, as researchers were provided with an excerpt of the data in order for them to finetune their approaches. All the data was only used by the organizers to evaluate all submitted approaches. As a consequence, the results presented in the related work are likely to draw an overly optimistic picture of the effectiveness. This substantiates the decision to study the general validity of the matching techniques on separate datasets in this thesis.

Considering that companies maintain model collections with up to hundreds or thousand of models, the extent of the empirical data in the literature is rather small. Only four publications comprise more than 100 different model pairs. In this regard, [190] is a notable exception as 1,200 model pairs are used. The overall limited extent of data was considered during the collection of empirical data in this thesis. More details regarding the data are given in the next section.

In summary, the assessment of the four questions justifies the research in this thesis. On the one hand, this pertains the improvement of the existing matching techniques. While the applicability of the matchers is generally high, it was shown that their overall effectiveness is insufficient. On the other hand, the research approaches in the literature focused on the

evaluation of the matchers. But, the effects of the underlying design decisions are rarely studied. Moreover, development and evaluation data is rarely separated impacting the generalizability of the findings. These two issues substantiate the research design chosen in this thesis. Moreover, the amount of empirical data used in prior work is generally small and hence limits the general validity of the results. This warrants a broader evaluation which is hampered by the unavailability of datasets. Thus, as outlined in the next section the author of this thesis aimed to improve the situation by collecting additional datasets and making them (partly) available to the community.

3.4 Model Collections

As outlined in the previous section, prior work primarily focused on evaluating the effectiveness of matching techniques. Due to the lack of detailed analyses the validity and the limitations of assumptions and design decisions are rarely studied. In order to overcome this problem, this thesis utilizes four real-world datasets for analysis and evaluation. In particular, two datasets are used during *development* to study fundamental design decisions based on behavioral analyses as well as to evaluate and fine-tune designs of matching techniques. Using all data during development entails the risk to fit the matchers to the data and to over-estimate the validity of design decisions [37]. Consequently, two *evaluation* datasets are exclusively used to exmaine the general validity and the limitations of the matchers. Contrary to prior work, this approach allows for explicating the limitations and the general validity and hence also fosters reuse. In the following all four datasets are introduced and characterized with regard to their models and gold standards. Moreover, as some of the datasets were used in the matching contests of 2013 and 2015 [19, 20], the respective results are introduced as a baseline for the evaluation of the matching techniques.

The University Admission (UA) dataset was introduced by Leopold et al. [194] and also used in the process model matching contest 2013 and 2015 [19, 20]. It contains nine Petri net models that describe admission processes of nine different German universities. The process models were created by students in the context of business process modeling

lectures at Humboldt University of Berlin. The gold standard was created by three experts. Two of the experts created alignments for all of the 36 model pairs manually. Afterwards, these alignments were merged by the third expert who dissolved differences. Note that in the second matching contest [20] a different version of this dataset including BPMN models and a new gold standard was used. In contrast to the original gold standard from [19, 194] the new version is based on the assumption that correspondences only exists between activities with the same or similar roles. As outlined in Section 3.3.1 it cannot safely be assumed that such information is present in the models. Moreover, the alignment of roles is a separate problem and the selection of correspondences based on role similarity can be implemented as a post-processing step where the alignment between roles is used to filter the previously identified correspondences. For these reasons, the first version of the dataset from [194] along with the evaluation results from [19] is used in this thesis. This version is publicly available[18].

The second dataset is the Birth Registration (BR) dataset which was introduced in the context of the process model matching contest 2013 [19]. It also comprises nine Petri net models which describe processes for birth registration in Germany, Russia, South Africa and the Netherlands. Whereas four models were again created by students at Humboldt University of Berlin, five of the models stem from a process analysis project at Dutch municipalities. The creation of the gold standard followed the same procedure that was applied for the UA dataset.

The UA and the BR datasets were chosen as development datasets because they were available at the beginning of the research project and guided the development of earlier versions of the presented techniques [50, 51]. The models in both datasets were (partly) created by students. Considering such models as real-world data is justified by the observation that the modeling performance of students is similar to the performance of experts [40]. However, to also include model collections from a professional background, the SAP Reference Model (SR) and the Alma Web (AW) dataset were developed in a later phase of the research project and used as evaluation datasets.

[18] http://www.henrikleopold.com/downloads, accessed: 13/01/2017

TABLE 3.3: Descriptive statistics for the process model collections

Dataset		*Models*			*Activities*			
		#	*Pairs*	*Language*	*Min*	*Max*	*Ø*	*Σ*
Alma Web	AW	9	36	German	3	22	7.4	67
Birth Registration	BR	9	36	English	9	25	19.3	174
SAP Reference Model	SR	72	36	English	1	43	9.3	667
University Admission	UA	9	36	English	13	48	27.6	248

The SAP Reference Model (SR) dataset is based on the SAP Reference model which was discussed in the literature [64, 85, 146]. It contains process models related to finance and accounting. The dataset was created by the author based on the similarity search evaluation in [146]. It comprises 36 model pairs, but in contrast to the UA and the BR datasets these model pairs comprise 72 different EPC models. Furthermore, the dataset covers a broad variety of scenarios. There are model pairs with almost identical models and some model pairs comprise models that do not share any correspondences. The rest of the model pairs is somewhere in between. The gold standard was created by two experts including the author that independently identified gold standards. These gold standards were automatically merged and the differences were dissolved in a discussion between both experts. The gold standard was provided to the process model matching contest 2015 [20] and was published[19] by the organizers of the second contest together with the BR dataset.

The last dataset is the Alma Web (AW) dataset which contains nine BPMN process models from different faculties of the Leipzig University. The process models were created within the AlmaWeb project [20] and deal with the examination management at the faculties. While all other datasets were created in English, this dataset contains labels in German. For the creation of the gold standard the author applied the same procedure as for the SR dataset. This dataset is not publicly available.

[19] https://ai.wu.ac.at/emisa2015/contest.php, accessed: 13/01/2017
[20] https://almaweb.uni-leipzig.de, accessed: 13/01/2017

TABLE 3.4: Descriptive statistics for the gold standards

				Activity Pairs	
Dataset	*1:1*	*1:n*	*m:n*	*Corresponding*	*Total*
AW	27	53	25	375	1,866
BR	156	95	13	584	13,358
SR	137	16	3	218	4,559
UA	251	77	1	531	26,853

Table 3.3 provides an overview of the process models that are part of the datasets. While all datasets comprise 36 model pairs, the size of the process models differs as indicated by the average number of activities. On average the AW dataset contains the smallest process models, followed by the SR dataset. In contrast, the models of the other two other datasets contain more activities.

Furthermore, descriptive statistics of the gold standards are provided in Table 3.4. The distribution of correspondences is different in all datasets. Due to the variety of the matching scenarios the SR dataset has the smallest number of correspondences. Most of these correspondences constitute elementary correspondences. The AW dataset has the second smallest number of correspondences. However, from a relative perspecitve 20% of the activity pairs correspond. This is the highest value among all datasets. Additionally, only a small share of the correspondences comprises elementary correspondences. The UA dataset has the lowest share of corresponding activity pairs and similar to the SR dataset a large amount of the correspondences are elementary correspondences. Finally, the BR dataset contains the most correspondences. Like the AW dataset it is also characterized by a huge share of complex correspondences. In summary, the datasets cover a broad variety of characteristics with regard to the model collections and the gold standards.

Three of the datasets were used in at least one of the two matching contests. Thus, the corresponding results are used as a baseline in this thesis to compare the proposed techniques to the state of the art. In this regard, the best technique in terms of the f-measure was chosen for each dataset. As explained in Section 3.1 the micro f-measure is in the focus of this thesis. Thus, it is used to select the best performing matchers from

TABLE 3.5: Results of the matching contests 2013 and 2015 (cf. [19, 20])

Dataset	*Matcher*	pr_μ	re_μ	F_μ	pr_M	re_M	F_M
BR	RMM/NSCM	-	-	-	.68	.33	.45
	pPalm-DS	.502	.422	.459	.499	.429	.426
UA	RMM/NSCM	-	-	-	.37	.39	.38
SR	AML-PM	.786	.595	.677	.664	.635	.480

the contests. However, in the first edition of the contest [19] only macro level measures were used to evaluate the matchers. Thus, macro f-measures are considered where no micro level f-measures are available. Moreover, techniques developed by the author were excluded in order to achieve a comparison to the state of the art. The best results for each dataset are summarized in Table 3.5.

As the UA dataset was only used in the first contest, the results of the RMM/NSCM matcher which yielded the best f-measure are considered. The BR dataset was used in both contest editions. Here, the RMM/NSCM matcher also performed best in the first contest [19] and pPalm-DS in the second [20]. As both matchers achieve a similar macro f-measure and micro f-measures are available for pPalm-DS, the results of RMM/NSCM are discarded and only those of pPalm-DS will be used as a baseline. Finally, the AML-PM achieves the best performance on SR [20].

3.5 Summary

This chapter dealt with the topic of business process model matching. Hence, it first introduced the basic terminology and formal definitions. In this regard, a basic understanding of matching techniques was provided. Additionally, it was shown how the effectiveness can be assessed based on a set of model pairs and a respective gold standard that comprises manually identified correspondences. Here, the (micro) f-measure was identified as the primary effectiveness indicator.

Furthermore, the chapter discussed sub-hypothesis H1 and gave evidence to the practical and scientific demand for further research on business process model matching techniques. The practical need for matching techniques was demonstrated by reviewing a

variety of use case scenarios. Therefore, an overview of approaches from the literature that support various tasks in BPM and for which the availability of correspondences is a necessary prerequisite was presented. Next, the scientific demand was substantiated through a critical literature survey. In this survey 17 publications that introduced business process model matching techniques were examined. It was shown that a broad applicability of the techniques in different matching scenarios is generally ensured, but that their effectiveness is insufficient. Moreover, the literature analysis revealed that the research design in prior work is a further limiting factor. On the one hand, prior research primarily focused on the evaluation of the effectiveness, but did not analyze the limitations and the validity of the design decisions. Thus, reuse of matching techniques and design decisions is not enforced. On the other hand, the amount of empirical data is typically small and all data is used for the development which usually leads to an optimistic view onto the evaluation results [37]. Although the survey only focused on the matching techniques and ignored research from related fields, these findings are considered to draw a representative picture for work on business process model matching. Hence, they justify the research for more effective matching techniques and also back up the research approach in this thesis.

Finally, the empirical data used in this thesis was introduced. Here, the four datasets which were divided into two evaluation and two development datasets were described. In this regard, the origin of the models and the creation of the gold standards was discussed. Based on descriptive statistics it was shown that the four datasets cover a variety of scenarios. Lastly, evaluation results for three of the four datasets from the matching contests in 2013 and 2015 [19, 20] were summarized. These results serve as a baseline for techniques developed in this thesis.

4

Comparing Activity Labels

H2: Label-based matching techniques yield a varying and generally insufficient effectiveness.

While the model elements of a process modeling language provide means to capture relevant aspects of processes and relate them to each other, the labels which are brief descriptions expressed in a natural language assign meanings to these elements. Hence, they constitute the primary source of information to determine the similarity of two activities. This chapter draws on the importance of labels for business process model matching and examines sub-hypothesis H2 by developing the Bag-of-Words Technique (BOT), a matching technique solely relying on labels. To this end, the matching technique is iteratively refined by evaluating and comparing the effectiveness of different versions on the development datasets. First, a basic matching algorithm is introduced in Section 4.1.

This algorithm considers labels as strings of characters. Subsequently, Section 4.2 refines the algorithm by extracting words from labels and assessing the similarity of labels through a comparison of the words. Whereas the first two variants compute similarity scores at the syntactic level, Section 4.3 further extends the algorithm and examines measures to evaluate the semantic relatedness of words. That is, instead of comparing words based on how they are composed of single characters, the similarity of words is assessed with regard to their meanings. Following, the resolution of differences in label specificity is discussed in Section 4.4. Such differences occur when labels provide different levels of detail. Then, Section 4.5 presents BOT which is based on the introduced matching algorithms. BOT is configurable and comprises various features for which different manifestations are provided. Next, Section 4.6 analyzes BOT. In this regard, the maximum effectiveness yielded by the BOT configuration is assessed on the development and evaluation datasets. In this context, a default configuration that can directly be applied is derived from the evaluation on the development datasets and a semi-manual configuration approach that allows experts to configure BOT with regard to model collection characteristics is examined. Furthermore, a qualitative analysis of BOT's results is carried out. Together these analysis results give evidence to sub-hypothesis H2. Finally, Section 4.7 concludes the chapter.

4.1 Basic Label Matching

To construct an alignment for two given process models P, P' and their respective sets of activities A, A', correspondences need to be extracted from the set of all activity pairs $A \times A'$. In this regard, the basic matching algorithm outlined in Algorithm 4.1 constitutes a simple strategy to distinguish corresponding from non-corresponding activity pairs.

The algorithm takes two process models and iterates over the set of all activity pairs that can be constructed from the pair of process models (lines 2 to 11). For each activity it determines the normalized label by applying the label normalization function $norm$ (lines 3 and 5). This is done to convert the labels to a common syntactic format. Here, the following normalization techniques from [138] are applied. First, *case normalization* is applied to transform all capital alphabetic characters into their lower case counterpart.

Algorithm 4.1: Basic label matching algorithm

```
   Input: P = (N, E, λ, τ, A), P' = (N', E', λ', τ', A')
   Output: 𝒜
1  𝒜 = ∅;
2  foreach a ∈ A do
3      label = norm(λ(a));
4      foreach a' ∈ A' do
5          label' = norm(λ'(a'));
6          similarity = σ.λ(label, label');
7          if similarity ≥ ϑ then
8              𝒜 = 𝒜 ∪ {(a, a')};
9          end
10     end
11 end
```

Second, *punctuation elimination* is carried out to replace any punctuation sign with a single blank character. Third, all links between two words are converted to a blank character. This step is referred to as *link stripping*. Next, through *digit suppression* all numerical digits are removed. Finally, *blank normalization* replaces blank characters, like tabulation or carriage return, with a single blank character. Such harmonization techniques are also part of the matching technique developed by Dijkman et al. [116] and RMM/NSCM from the matching contest [19].

Definition 4.1 (Label normalization) *Given the set of all labels $\mathscr{L}$, the function*

$$norm : \mathscr{L} \rightarrow \mathscr{L}$$

returns the normalized version of a given label by applying case normalization, punctuation elimination, link stripping, digit suppression as well as blank normalization.

Based on the normalized labels, a similarity score for the activity pair is computed (line 6). Therefore, a label similarity function $\sigma.\lambda$ is applied. Such a similarity function returns a score on the interval $[0, 1]$. The rationale is that high values suggest a strong similarity between the activities and low values indicate differences.

Definition 4.2 (Label similarity) *Given the set of labels $\mathscr{L}$, the label similarity function $\sigma.\lambda$ is defined as*

$$\sigma.\lambda : \mathscr{L} \times \mathscr{L} \rightarrow [0,1]$$

where a value of 1 *indicates equality, a value of* 0 *total dissimilarity and values in between are interpreted as degrees of similarity.*

The last step in the classification of an activity pair is the evaluation of the similarity score yielded by applying the label similarity function. That is, if the similarity score is higher than or equal to a predefined threshold $\vartheta \in [0,1]$ (line 7), the activity pair is classified as a correspondence and added to the alignment (line 8).

The basic label matching approach is a generic classification mechanism in which the threshold is used to decide, if activity pairs are similar enough with respect to a label similarity function in order to be considered as correspondences. Thus, as illustrated in Figure 4.1, the effectiveness of the algorithm is determined by the specific label similarity function and the value to which the threshold parameter is set. Here, two different instances of the algorithm are applied to ten activity pairs (circles) among which three correspond (black circles). The first instance (top) relies on the label similarity function $\sigma.\lambda_1$. The proposed alignment (grey background) contains many non-corresponding activity pairs yielding a low effectiveness. This can be traced back to $\sigma.\lambda_1$ which poorly separates corresponding from non-corresponding activity pairs. That is, the three corresponding activity pairs are spread over the entire interval of $[0,1]$ and so are the non-corresponding pairs. Thus, introducing a threshold yields two sub-sets where at least one of them contains corresponding and non-corresponding pairs. In contrast, $\sigma.\lambda_2$ (bottom) achieves a perfect f-measure of 1. Here, the function yields high similarity scores for all correspondences,

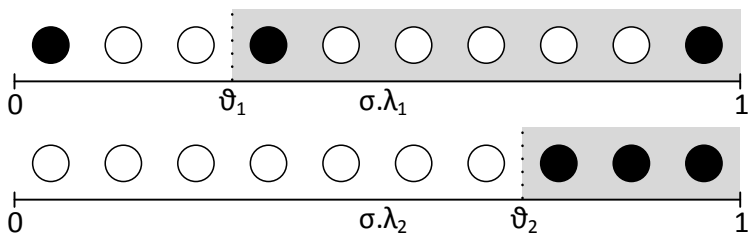

FIGURE 4.1: Two configurations of the basic label matching algorithm

whereas all non-corresponding activity pairs are assigned to low values. Consequently, a threshold can be introduced that perfectly separates the set of activity pairs. Based on these considerations the remainder of this chapter is devoted to the maximization of the effectiveness of the label-based matching algorithm by improving the assessment of the label similarity.

A first strategy that constitutes the starting point of this development is to consider activity pairs as corresponding, if their labels are equal. To adapt this strategy, the Equal String Similarity (EQL) is introduced. The function returns a value of 1, if the two labels are equal and 0 otherwise. When using the label equality function in the basic matching technique the threshold parameter is set to 1. That way all activity pairs with equal labels are considered as correspondences whereas all other activity pairs are neglected. The matching technique by Branco et al. [16] incorporates label equality to detect elementary correspondences, too.

A drawback of requiring label equality is that minor differences in the labels already have a big impact on the recall. Labels might differ due to spelling errors, different word forms, etc. In such cases EQL classifies labels that clearly express the same functionality as non-corresponding. In this respect, string similarity measures provide a more differentiated assessment of the similarity of labels. They consider labels as compositions of characters and investigate to which degree these compositions overlap. Hence, they are less susceptible to minor differences. In the following, a set of well-known string similarity measures [138] is introduced.

The first measure is the Normalized Hamming Similarity (HAM) which is based on the *Hamming distance* [206]. First, the number of positions with different characters in both strings and the difference of the strings' lengths are computed. Then, HAM is the normalized sum of these values. Here, the normalization is achieved by dividing the sum with the maximum length of the strings. Finally, the distance value is transformed into a similarity value by subtracting it from 1.

The Sub-String Similarity (SUB) [138] relies on the longest sub-string that appears in both strings. It is defined as the ratio of twice the length of the longest sub-string and the sum of the lengths of the strings.

In contrast to SUB, the Longest Common Sub-Sequence Similarity (LCS) [207] takes sub-sequences rather than sub-strings into account. The characters of a sub-sequence do not need to consecutively occur in the string. Instead, all characters of a sub-sequence only need to appear in the same order in the string. Thus, a sub-string is a special kind of sub-sequence. For example, consider the strings "rejecting application", "reject" and "reject application". While "reject" is both a sub-sequence and a sub-string of "rejecting application", "reject application" is a sub-sequence, but not a sub-string of "rejecting application". Given the longest common sub-sequence of two strings, LCS is the ratio of twice the length of this sub-sequence and the sum of lengths of the strings.

Next, there are similarity measures that determine all *n-grams*, i.e., sub-strings of length n, in both strings [138]. With regard to the n-grams a similarity score is calculated as the fraction of the number of n-grams appearing in both strings and the number of n-grams in the shorter string. Here, the Bigram Similarity (2G), the Trigram Similarity (3G), and the Quadrigram Similarity (4G) are considered.

Another way to compare two strings is to assess the costs of transforming one string into the other. Those measures are called edit distances [138] and the *Levenshtein distance* [208] is a well-known measure of this class. It defines the edit costs as the minimal number of operations needed to transform a string into the other. These operations include the insertion, the deletion, and the substitution of a character. The Levenshtein Similarity (LEV) is defined as the fraction of the Levenshtein distance and the length of the longer string subtracted from one. Dijkman et al. [116] apply this measure to compute a label similarity score.

Finally, the *Jaro measure* [209] considers the number of equal characters on the same position as well as transposed characters. Here, the Jaro Winkler Measure (J/W) as a refined version of the Jaro measure is considered [210]. In contrast to the Jaro measure, it additionally takes prefixes into account.

These nine label similarity measures and the threshold parameter ϑ span the configuration space of the basic label matching algorithm. Figure 4.2 summarizes this space by showing the feature model [211, 212] for the algorithm.

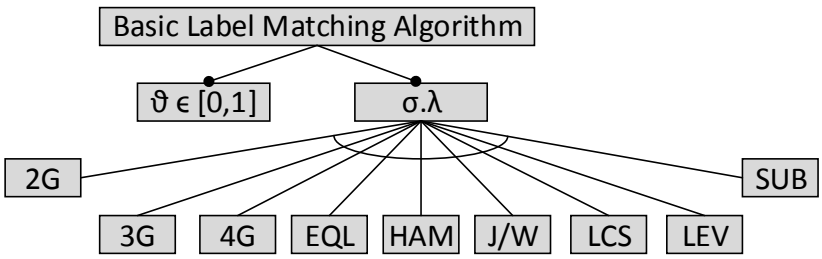

FIGURE 4.2: The feature model for the basic label matching algorithm

In order to apply the algorithm, a label similarity measure must be chosen and the threshold parameter must be set to a specific value. Whereas the determination of the threshold parameter for the label equality function EQL is straightforward due to its binary nature, this is not the case for the string similarities. In conformance with the definition of the label similarity function high scores yielded by string similarity functions are seen as an indication for the similarity relation between activities. Yet, there exists no universal threshold parameter that maximizes the effectiveness for a string similarity function, i.e., that yields the best possible separation of corresponding and non-corresponding activity pairs. In fact, it will be shown that the optimal threshold value varies from similarity to similarity and across the datasets.

Thus, the threshold parameter is optimized for each string similarity on each of the development datasets. Given a string similarity and a dataset, the set of threshold candidates comprises all distinct similarity scores yielded by applying the string similarity to all activity pairs in the dataset. For each of the threshold candidates the effectiveness is measured. Therefore, the alignments resulting from the application of the threshold candidate are determined and compared to the gold standard. Then, the candidate with the highest micro f-measure is chosen as the optimal threshold for the string similarity with regard to the dataset. Table 4.1 summarizes the effectiveness of all label similarity functions in combination with their optimal threshold parameters.

The results show that relying on label equality or utilizing string similarity measures does not guarantee practical applicability as the effectiveness is rather low. In terms of the micro f-measure the best results are yielded by LCS on both datasets. However, it only achieves a micro f-measure of .415 on BR and of .361 on UA. The main reason for the poor effectiveness is the overall low recall. Here, the maximum value on BR is .442

TABLE 4.1: Effectiveness of the basic matching algorithm

	BR				*UA*			
$\sigma.\lambda$	ϑ	pr_μ	re_μ	F_μ	ϑ	pr_μ	re_μ	F_μ
EQL	1.00	**.855**	.161	.271	1.00	**.782**	.162	.268
LCS	.640	.490	.360	**.415**	.737	.531	.273	**.361**
SUB	.462	.447	.373	.407	.692	.595	.213	.313
LEV	.440	.364	**.442**	.399	.583	.367	.288	.323
2G	.625	.678	.274	.390	.528	.326	.328	.327
3G	.608	.640	.274	.384	.522	.321	.330	.325
4G	.592	.598	.272	.374	.494	.305	.341	.322
J/W	.907	.842	.238	.371	.780	.338	.335	.336
HAM	.345	.421	.293	.345	.268	.262	**.345**	.298

for LEV and on UA .345 for HAM. That is, for each label similarity less then 45% of all correspondences are detected. Additionally, the precision varies strongly. While in some cases the precision is very low, e.g., it is .364 for LEV on BR and .262 for HAM on UA, there are also high values. Here, EQL achieves the best results with .855 on BR and .782 on UA. The high precision shows that label equality can basically be considered as an indicator for a correspondence relation between activities. That is because a high share of the equally labeled activity pairs actually corresponds, but exceptions must be tolerated. In the development datasets such exceptions primarily include activity pairs where the activities are carried out at different points in the processes. Thus, they might either be carried out by different, but implicit roles or in different contexts for slightly different purposes. However, label equality is clearly not a sufficient criteria as only a small share of the correspondences has equal labels. The recall of *EQL* is approximately .16 on both datasets.

The effectiveness is not only low, but also varying across the similarities and the datasets. In this regard, all similarities achieve a better micro f-measure on BR than on UA. Moreover, the relative performance of the label similarities varies. For instance, SUB ranks second on BR, but only seventh on UA. Similarly, J/W ranks seventh on BR and second

on UA. Additionally, the optimal threshold values for the label similarities differ across the datasets. On average the difference between the optimal threshold values per string similarity is .119 with LEV yielding the biggest difference ($|\vartheta_{BR} - \vartheta_{UA}| = |.440 - .583| = .143$). Note that EQL was excluded as its threshold parameter is fixed. These observations provide first evidence that label-based matching techniques yield a varying effectiveness across model collections. Thus, they need to be adapted to the domain specifics of the model collections in order to optimize and stabilize their effectiveness. Furthermore, they illustrate that the heterogeneity of the labels and thus the difficulty to automatically detect correspondences are likely to differ across datasets.

Finally, the table shows that many of the optimal thresholds are fairly low. In total there are four similarities with an optimal threshold below .6 on BR and five on UA. That is, in 50% of the cases the optimal threshold violates the definition of the label similarity where low values are required to indicate a dissimilarity. HAM even takes an optimal threshold value of .268 on UA. Thus, the optimal threshold values do not provide an indication for the degree to which activity pairs can safely be considered to correspond. Instead, they are optimized values for which the most effective separation of corresponding and non-corresponding activity pairs was observed. In consequence, these low values provide further evidence that applying string similarities to the entire labels is a generally insufficient strategy.

4.2 Label Decomposition

Although labels usually consist of several words, they have been treated as a single sequence of characters so far. Mendling et al. [38] argued that there are three classes of words in a label: an activity label typically comprises an *action* that is performed on an *object* and it might provide *additional information*, like roles responsible to perform the operation or conditions that need to be met. Moreover, natural languages typically allow to compose words in different ways. Accordingly, in an empirical analysis Leopold [39] observed four labeling styles for activities. The Verb-Object labeling style (VO) characterizes labels where the action is expressed by a verb, e.g., “accept application”. Further,

TABLE 4.2: Relative frequencies of the activity labeling styles

Style	*BR*	*UA*
VO	95.4%	75.0%
AN	2.3%	4.0%
DES	1.7%	0.0%
NA	0.6%	21.0%

the Activity-Noun labeling style (AN) refers to labels where the action is represented as a noun, e.g., "application acceptance" or "accepting application". Moreover, labels adhere to the descriptive labeling style (DES), if they contain a role and the action is expressed by a verb in the third person form, e.g., "faculty accepts application". Labels based on these styles provide information on the action performed through an activity. Thus, these styles constitute regular labeling styles. In contrast, the No-Action labeling style (NA) subsumes all labels that do not contain an action and can be considered as irregular or anomalous, like "accepted" or "application". Leopold [39] further reports the number of occurrences of these styles within the SAP Reference Model and two other process model collections. The collections consist of 328 to 604 models and each collection contains more than 2,400 activities. Whereas 81% of the activity labels in the SAP Reference Model are classified as AN, in the two other collections 74% and 80% of the activity labels are assigned to VO. Additionally, only a small share of activities adheres to DES. Interestingly, about 10% of all activity labels in each collection belong to NA. As shown in Table 4.2 there is also one dominant labeling style in the development datasets. On both datasets VO is the most frequent style with 95.4% of all activities belonging to this class on BR and 75% on UA. While due to the high frequency of VO BR can be considered as very homogeneous, UA is also characterized by a large share of irregular activity labels (21%).

Whereas Leopold [39] defines the labeling styles with regard to the action and thus primarily addresses the use of action fragments, a more diversified picture of the labeling styles can be gained by also looking at the frequencies of the other two classes (business object and additional information). Table 4.3 presents these frequencies for the development datasets. Note that because of the anomaly of the AN labels, only the regular labels

TABLE 4.3: Frequencies of object and additional information fragments in regular labels

Object	*Add. Information*	*BR*	*UA*
not in label	not in label	2.3%	4.0%
occurs in label	not in label	76.4%	53.6%
occurs in label	occurs in label	16.1%	14.1%
not in label	occurs in label	4.6%	7.3%

are considered. In BR (UA) 76.4% (53.6%) of the regular labels contain an object, but no additional information. Another 16.1% (14.1%) of the activities also contain additional information. Furthermore only 2.3% (4.0%) of the activities contain only an action and 4.6% (7.3%) consist of an action and additional information.

Overall, these descriptive statistics show that labeling styles are typically inconsistently applied within model collections and that labels might only contain a subset of the three classes which might be composed in different ways. Accordingly, matching techniques must be prepared for varying labeling styles. That matching techniques which consider labels as single strings do not address the problem of heterogeneous labeling styles is shown in Table 4.4. Here, three string similarities are applied to three activity pairs where each pair consists of the activity "accept application" and another activity. First, there is the activity "accept" which does not contain an object or additional information. Although, depending on the context this label might be similar to "accept application", the string-based label similarities yield low values. This is because the difference in the label length distorts the similarity calculation. The same effect can be observed for the activity "accept application if requirements are met" where the label contains an object and additional information. Finally, there is the label "reject application". It yields the highest similarity scores among all three labels although it has the opposite meaning. The reason is that both labels have a length of 18, but only the first five letters differ.

The varying use of labeling styles typically impacts the similarity assessment for a huge share of activity pairs, even for model collections with a rather homogeneous use of labeling styles. This can best be illustrated with regard to the BR dataset where the values from Table 4.4 are considered to reflect the actual distribution of labeling styles. Although

TABLE 4.4: String similarity scores for "accept application" and a second label

Second Label	*LCS*	*LEV*	*3G*
accept	.500	.333	.333
accept application if requirements are met	.600	.429	.429
reject application	.722	.778	.722

these values indicate a rather homogeneous labeling style, only $(2.3\%)^2 + (76.4\%)^2 + (16.1\%)^2 + (4.6\%)^2 = 61.2\%$ of all activity pairs have the same labeling style.

To better address varying labeling styles, a refined version of the basic label matching algorithm is introduced in the following. It breaks labels down into sets of words in order to compute a similarity score. Therefore, it relies on the *bags-of-words* model which has been adopted in linguistic contexts [213, 214] and is also widely used in the field of object recognition, e.g., [215]. Rather than considering texts as sequences of words with a certain order, the bag-of-words model omits the structure and represents texts as multi-sets of words. Such a multi-set contains the words from the respective text and provides information on how often these words occur within the text. As labels are rather short texts, it might be assumed that they do not contain words more than once and can thus be represented as sets of words. However, there are counterexamples that violate this assumption. For example, Pittke et al. [165] report that labels sometimes contain descriptions of two or more distinct activities, like "evaluate application and check application". Similarly, labels might contain an additional information fragment that represents a condition for the activity execution, e.g., "accept applicant if the applicant is qualified". Although model collections usually only contain a small number of such labels, the bag-of-words model is adapted here to account for the general case.

Definition 4.3 (Bag-of-words) *Given the set of words $\mathscr{W}$, a bag-of-words*

$$\varpi : \mathscr{W} \rightarrow \mathbb{N}$$

is a multi-set that returns the number of occurrences for a given word in a text document. The support of the bag-of-words $supp(\varpi)$ comprises all distinct words that occur in the text document, i.e., $\forall w \in \mathscr{W} : w \in supp(\varpi) \Leftrightarrow \varpi(w) > 0$. Additionally, the total number

of words in the text document is the cardinality of the bag-of-words which is defined as $|\varpi| := \sum_{w \in supp(\varpi)} \varpi(w) := \sum_{w \in \mathscr{W}} \varpi(w)$.

The decomposition of labels into bag-of-words is referred to as tokenization. During tokenization the words are extracted from a label and *stop words* are removed. Such words are function words of a natural language that only carry little semantic meaning [214], like "a", "be" or "could"[21]. To transform a label into a bag-of-words the tokenization function tok is used in this thesis.

Definition 4.4 (Tokenization) *Given the set of labels* $\mathscr{L}$ *and the set of all bag-of words* O^{∞} *the tokenization function*

$$tok : \mathscr{L} \rightarrow O^{\infty}$$

returns the bag-of-words for a label by splitting the label into individual words and removing the stop words.

Based on the bag-of-words model and the tokenization function, the basic matching algorithm can now be refined in terms of the bag-of-words matching algorithm. As shown in Algorithm 4.2, it also iterates over the set of all activity pairs (lines 2 to 13). For each activity it determines the normalized label (lines 3 and 6) and the according bag-of-words (lines 4 and 7). If the similarity score determined for an activity pair is higher than or equal to the predefined threshold ϑ, it classifies the according pair as a correspondence and adds it to the alignment (lines 9 and 10). In contrast to the basic matching algorithm, the similarity score is determined by applying the bag-of-words similarity $\sigma.\varpi$ rather than a label similarity function $\sigma.\lambda$ (line 8).

The most important part of the algorithm is the bag-of-words similarity because it breaks the comparison of two labels down into the comparison of their words. To determine a similarity score the bag-of-words similarity first applies a *stemming* algorithm to each word. Similar to the tokenization which eliminates differences in the label structure, i.e., it omits information about the position of words, stemming is carried out

[21] In this thesis the default English stop word list and the German stop word list from `http://www.ranks.nl/stopwords` (accessed: 13/01/2017) are used.

Algorithm 4.2: Bag-of-words matching algorithm

```
Input: P = (N, E, λ, τ, A), P' = (N', E', λ', τ', A')
Output: 𝒜
1  𝒜 = ∅;
2  foreach a ∈ A do
3      label = norm(λ(a));
4      ϖ = tok(label);
5      foreach a' ∈ A' do
6          label' = norm(λ'(a'));
7          ϖ' = tok(label');
8          similarity = σ.ϖ(ϖ, ϖ');
9          if similarity ≥ ϑ then
10             𝒜 = 𝒜 ∪ {(a, a')};
11         end
12     end
13 end
```

to erase effects arising from different labeling styles. In particular, stemming aims to harmonize the word forms. That is, it strips off affixes in order to find a word's basic form [214]. The labels "accept application" and "accepting application" constitute an example where stemming can help to improve the comparison of the labels. Here, stemming can be used to harmonize the words and to reduce "accepting" to its basic form "accept". To incorporate stemming algorithms the bag-of-words similarity relies on a stemming function st that returns a set of possible stems for a word.

Definition 4.5 (Stemming) *Given the set of words $\mathscr{W}$ a stemming function*

$$st : \mathscr{W} \rightarrow \mathscr{P}(\mathscr{W})$$

returns a set of words which comprises possible stems of the word.

There is a plethora of stemming algorithms available. Some algorithms utilize a set of predefined rules or suffices to stem words, e.g., [216, 217]. Other algorithms are based on statistical measures and corpora analysis [218–220]. In this thesis, two stemming algorithms are considered. First, there is the Porter Stemming Algorithm (PSA) [221]

which is a common rule based matcher [214]. Here, the implementation by Porter[22] which is also available for languages other than English including German, Russian, and French is utilized. Second, the WordNet Stemming Algorithm (WSA) provided by the MIT Java Wordnet Interface (JWI) library[23] is applied. It aims to reduce a word to its stem based on dictionary lookups. In particular, it relies on *WordNet* which is a widely used lexical database for English [222] (cf. Section 4.3). The JWI library was suggested for accessing WordNet based on a comparison of different libraries [223]. Moreover, stemming can be deactivated. At implementation level this is achieved by using a stemming function that returns an empty set for each word.

Furthermore, the bag-of-words similarity needs to assess the similarity of words, in order to compute an overall similarity score for two activities. Therefore, it incorporates a word similarity function $\sigma.w$.

Definition 4.6 (Word similarity) *Given the set of words* $\mathscr{W}$ *a word similarity function*

$$\sigma.w : \mathscr{W} \times \mathscr{W} \rightarrow [0,1]$$

returns a similarity score for a given pair of words where a value of 1 *indicates equality, a value of* 0 *total dissimilarity and values in between are interpreted as degrees of similarity.*

As words are also sequences of characters, the label similarity functions introduced in the previous section can be directly applied here. As outlined, these similarities compare words on the syntactical level, i.e., words are considered similar, if they share a large portion of characters and these characters appear in a similar order.

The bag-of-words similarity $\sigma.\varpi$ then computes a similarity as outlined in Table 4.5. Based on the bag-of-words {“accept”, “application”} and {“reject”, “application”}, a word similarity function (here LCS) is applied to determine a similarity score for each word pair that consists of one word from each bag-of-words. Therefore, for each word the set of stems that includes the word itself is determined. For a given word pair the word similarity function is then applied to all possible combinations of the stems and the maximum similarity score is yielded. In the next step, for each word in the bag-of-words

[22] `http://snowball.tartarus.org`, accessed: 13/01/2017
[23] `http://projects.csail.mit.edu/jwi/`, accessed: 13/01/2017

TABLE 4.5: Illustration of the bag-of-words similarity using LCS as the word similarity

	reject	*application*	*max*
accept	.333	.353	.353
application	.235	1.000	1.000
max	.333	1.000	$\sigma.\varpi = .672$

the maximum similarity score yielded in the previous step is then determined. Finally, the overall score is the average of all these maximum scores. In the example this score is now .672. This value is lower than the score yielded by the basic label matching algorithm (.722). Thus, it better reflects the relation between the labels.

Definition 4.7 (Bag-of-words similarity) *Let ϖ, ϖ' be two bag-of-words and $\Omega = supp(\varpi)$, $\Omega' = supp(\varpi')$ be the words occurring in these bag-of-words. Given a stemming function st and a word similarity $\sigma.w$, the bag-of-words similarity $\sigma.\varpi$ is defined as:*

$$\sigma.\varpi(\varpi, \varpi') := \frac{\sum_{w \in \Omega} \varpi(w) \cdot \max_{w' \in \Omega'} \sigma.st(w, w') + \sum_{w' \in \Omega'} \varpi'(w') \cdot \max_{w \in \Omega} \sigma.st(w', w)}{|\varpi| + |\varpi'|}$$

with

$$\sigma.st(w, w') = \max_{s \in \{w\} \cup st(w)} [\max_{s' \in \{w'\} \cup st(w')} \sigma.w(s, s')]$$

The effect of the bag-of-words similarity is further illustrated in Table 4.6. In this table the bag-of-word similarity scores for the same activity pairs and the same syntactic similarity measures as in Table 4.4 are presented. In contrast to considering labels as strings, the bag-of-words similarity separates the non-corresponding and corresponding activity pairs better. The similarity score for the activity pair "accept application" and "reject application" remains high (.65 on average) due to "application" being part of both

TABLE 4.6: Bag-of-words similarities for "accept application" and a second label

Label	*LCS*	*LEV*	*3G*
accept	.784	.727	.742
accept application if requirements are met	.778	.750	.704
reject application	.672	.667	.612

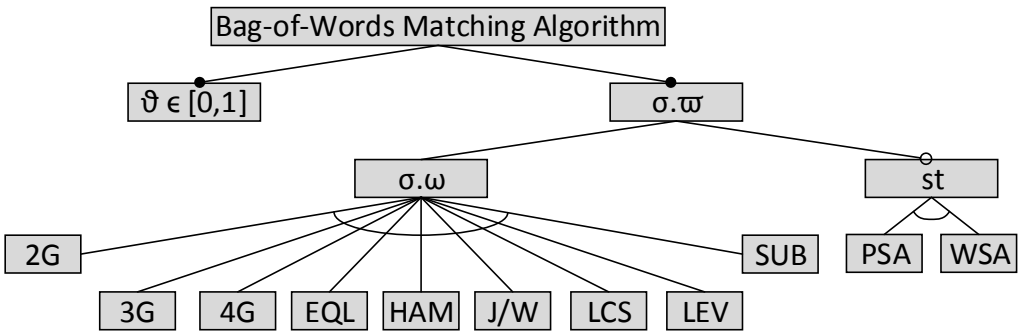

FIGURE 4.3: The feature model for the bag-of-words matching algorithm

labels. However, the scores for the corresponding activity pairs are improved and take a value of .75 on average. The reason is that in both cases all words from the shorter label also occur in the longer label and the effect of differences in the label length is reduced.

As outlined by the feature model in Figure 4.3 the space of possible configurations of the bag-of-words algorithm is larger than that for the basic label matching algorithm. While the application of the bag-of-words algorithm still requires to set a specific value for the threshold parameter ϑ, the computation of the label-based similarity score $\sigma.\varpi$ is more complex. That is, in addition to the selection of the string-based similarity measures to compare words $\sigma.w$, stemming can be activated or not. Moreover, in case it is activated a stemming function st needs to be selected.

To assess the effect of these configuration options in combination with the bag-of-words model, the bag-of-words matching algorithm was also evaluated on the development datasets. In this regard, all combinations of stemming options and word similarity functions were considered and the optimal threshold value was determined the same way as it was for the label similarities in Section 4.1. Table 4.7 summarizes the effectiveness for the combinations that comprise either HAM or SUB. Both word similarities yield the highest micro f-measure on one of the datasets when stemming is deactivated.

On both datasets the bag-of-words matching algorithm improves the maximum micro f-measure achieved by the basic label matching algorithm. HAM yields the highest value in combination with PSA on BR where it increases the prior maximum (.466 > .415). Similarly, SUB outperforms the basic label matching algorithm on UA where the maximum micro f-measure is yielded in combination with WSA (.430 > .361). Whereas SUB's performance is similar on both datasets, HAM yields a better effectiveness on BR.

TABLE 4.7: Effectiveness of the bag-of-words matching algorithm

		BR				*UA*			
st	$\sigma.\varpi$	ϑ	pr_μ	re_μ	F_μ	ϑ	pr_μ	re_μ	F_μ
-	HAM	.541	**.569**	.387	.461	.650	.524	.330	.405
	SUB	.521	.345	**.519**	.414	.708	.541	**.339**	.417
PSA	HAM	.515	.477	.455	**.466**	.767	.652	.303	.414
	SUB	.640	.500	.349	.411	.737	.549	.337	.418
WSA	HAM	.543	.568	.387	.460	.733	.606	.333	**.430**
	SUB	.532	.351	.514	.417	.758	**.726**	.284	.409

The reason for the increased effectiveness on BR is a higher recall. Here, a value of .519 constitutes a clear improvement of the best recall for the basic label matching algorithm (.442). On the contrary, the bag-of-words matching algorithm improves the precision on UA with a maximum of .726. Admittedly, for the basic matching algorithm EQL yielded a precision of .782. However, the respective recall is very low (.162). The remaining similarities achieved recall values similar to that of the bag-of-words algorithm, but their maximum precision is .595. This analysis provides evidence that the bag-of-words matching algorithm is to be preferred over the basic label matching algorithm. However, the improvements are modest and still do not permit a practical application.

According to the evaluation results, the stemming algorithms marginally impact the effectiveness. PSA improves the values yielded without stemming in three out of four times. Only for SUB on UA it yields a lower micro f-measure. WSA increases the micro f-measure twice: for SUB on BR and for HAM on UA. Overall, the effects are rather small as the maximum improvement yielded by PSA is .009 for HAM on UA and by WSA it is .025 for HAM on UA. This analysis indicates that stemming does not strongly impact the effectiveness and that there is almost no difference between PSA and WSA.

Lastly, the evaluation results confirm the observation that the effectiveness of algorithms differs across datasets and configurations. That is, the optimal threshold and effectiveness values vary for all combinations of word similarity and stemming functions.

4.3 Semantic Comparison of Words

Usually business process models are created by a group of modelers (cf. Section 2.2). As a consequence, the vocabulary within a model collection is likely to comprise a broad range of terms, especially when the model collection comprises models of different organizations. Such differences repose on the versatility and ambiguity of natural languages. That is, different words of a natural language might refer to the same sense or a single word might have different senses. Consequently, the same meaning can be expressed in different ways. For example, "send application electronically" and "submit application online" are different labels, but refer to the same activity. Additionally, syntactically identical expressions might convey different meanings, e.g., the label "prepare application" might refer to the creation of documents in order to reply to a job offer. In a different context, it also might be used to address the configuration of a piece of software. In this light, the string similarity functions seem to be inadequate as they compare the syntax, but not the semantics of words, i.e., the meaning and senses conveyed by words.

In the field of linguistics two classes of *sense relations* are distinguished: *paradigmatic* and *syntagmatic* relations [224]. Paradigmatic relations refer to the senses that can be assigned to words. They are general relations between words that exist regardless of the specific use of the words and that provide options to choose words in a certain context. There are six relations of this type: *hyponymy*, *meronymy*, *synonymy*, *incompatibility*, *co-meronymy*, and *opposites* [224]. Hyponymy and meronymy are relations referring to the inclusion of words. The former represents so called "is-a" relations where one word subsumes another, e.g., an application is a document. The latter refers to "part-whole" relations where a word is a member of another, like a chair is part of a faculty which again is part of a university. Synonymy is a relation that holds between words, if they represent the same sense, e.g., "assess" and "evaluate" are two verbs describing the act of judging the value or the worth of something. Similar to hyponymy and meronymy, incompatibility and co-meronymy refer to the exclusion of words. Incompatibility is a relation that represents the mutual exclusion of words. In other words, there is nothing that can simultaneously be part of both classes, e.g., there is nothing that can be a confirmation and a refusal.

Co-meronymy is characterized by words being part of the same whole, but not having any substance in common, e.g., the database and the graphical user interface of an enterprise application are separate parts of the application. Finally, opposites are pairs of words that logically belong together but represent incompatibles, like "open" and "closed", or "increase" and "decrease".

Syntagmatic relations refer to the appearance of words in the same context [224], e.g., in a sentence or phrase. Such relations provide options for chaining words in a sentence. On the one hand, there might be relations that are independent from the grammar and hold between words that might have a rather long distance between them. That is, words might "go together" in a specific context or not. On the other hand, there are relations between words that are situated close to each other and are part of the same grammatically well-formed construction. Such relations typically comprise normal ("drink water"), redundant ("female aunt"), or semantically clashing ("drink rock") uses of words in the same context.

To consider sense relations between words, measures for the semantic relatedness of words are utilized here. Such measures are based on Word Sense Disambiguation (WSD) which constitutes "... *the ability to computationally determine which sense of a word is activated by its use in a particular context.*" [225]. In the context of process model matching, WSD can be used to check whether two words in a label constitute the same or a similar meaning. Hence, it can contribute to the comparison of the meaning of the activity descriptions, i.e., the actual purposes of activities.

In order to associate words with senses WSD utilizes external information. It is derived from knowledge sources that can basically be divided into structured and unstructured sources [225]. Thesauri and machine readable dictionaries are structured resources that contain words and sense relations between them. Roget's thesaurus [226] and the Macquarie thesaurus [227] constitute widely adapted thesauri for natural language processing and WSD. WordNet [222, 228] is a prominent machine readable dictionary with wide application in WSD. There also exist dictionaries for other languages, e.g., GermaNet [229, 230] for German, WoNeF [231] for French, or EuroWordNet as a multilingual database for some European languages [232]. BabelNet [233] is a multilingual dictionary

that was initially created through the integration of Wikipedia[24] and WordNet. Thus, it builds on lexicographic and encyclopedic knowledge. Furthermore, ontologies as explicit specifications of conceptualizations [234] are another type of structured knowledge sources. They usually contain a specification of terminology and the hierarchical classification of the terms along with semantic relations between them. The suggested upper merged ontology and its domain ontologies [235] are examples of ontologies.

Unstructured resources include corpora that represent collections of texts. Whereas, sense-annotated corpora also include information regarding the senses of (a subset of) the words, raw corpora only contain texts. The Brown Corpus [236] and the British National Corpus [237] are well-known raw corpora in national language processing. Examples of sense-annotated corpora include SemCor [238], the line-hard-serve corpus [239], and the Open Mind Word Expert corpus [240]. Stopword lists, like the ones introduced in Section 4.2, are also unstructured knowledge resources for WSD. Furthermore, collocation resources provide information on the co-occurrences of words. The collocations in the British National Corpus and the Web1T corpus [241] are examples for such knowledge sources. A more detailed overview of knowledge sources for WSD is provided in [242].

WSD methods exploit such sources in order to derive information on the semantic relatedness of words. On an abstract level two strategies can be distinguished: *supervised* and *unsupervised* methods [225]. Supervised methods exploit sense-annotated corpora to train a classifier through the application of machine learning. Thus, to apply such methods there is usually some manual effort needed to provide the training data. Unsupervised methods do not incorporate any sense-tags and can be further subdivided into *corpus-based* and *knowledge-based* techniques [225]. Whereas, corpus-based methods rely on raw corpora, knowledge-based techniques utilize structured resources.

To integrate approaches that measure the semantic relatedness of words into the bag-of-words matching algorithm, several semantic word similarity functions that rely on unsupervised methods are introduced in the following, whereas supervised methods are discarded. The reason is that here the focus is on automatic matching techniques that determine alignments between process models without requiring experts to interfere

[24] http://www.wikipedia.org, accessed: 13/01/2017

because they aim to ease the experts' job. Demanding additional input from the experts violates this goal. In this regard, involving the experts is discussed in Chapter 6 where ADBOT is introduced. This matcher comprises a strategy to adapt sense relation measures by analyzing alignments provided by experts and can thus be considered to be a supervised WSD method.

The first set of semantic word similarity functions comprises paradigmatic relatedness measures based on WordNet. As already outlined before, WordNet is a lexical database for English. It contains words and paradigmatic sense relations between them. The set of words only comprises open-class words, i.e., verbs, nouns, adjectives, and adverbs, whereas closed-class word categories, like pronouns and prepositions, are not included [222]. The words in WordNet are assigned to synsets which represent specific concepts [222]. As each synset comprises words that represent the according concept, the synsets encode synonymy relations between words. WordNet further distinguishes between lexical and semantic relations [225]. Lexical relations exist between words and include amongst others opposite meanings. Semantic relations instead exist between synsets and include meronymy and hyponymy. According to the WordNet statistics[25] it contains about 155,000 words and approximately 118,000 synsets.

There is a plethora of word similarity functions that exploit the semantic relations in WordNet. Here, the measures included in the WordNet::Similarity module[26] [243, 244] are applied. For implementation purposes, the Java implementation[27] of this module is used. The module provides popular measures for paradigmatic sense relations and makes them available for the use with WordNet.

On a high level these measures essentially follow the same procedure to determine a similarity score for two words. First, they determine all synsets for each of the two given words. Then, they exploit various relations to calculate a score between each possible combination of synsets, where there is one synset for each of the two words. The maximal score yielded for the synset pairs constitutes the similarity score for the pair of words.

[25] http://wordnet.princeton.edu/wordnet/man/wnstats.7WN.html, accessed: 13/01/2017
[26] http://wn-similarity.sourceforge.net, accessed: 13/01/2017
[27] https://code.google.com/p/ws4j/, accessed: 13/01/2017

The Lesk Similarity (LESK) is based on the algorithm proposed by Lesk [245]. Given two synsets it determines a score based on the overlap of words in the descriptions of their senses. Here, the extended notion by Banerjee and Pedersen [246] that also considers semantic relations to other synsets is used.

There is a series of measures based on hyponymy relations between synsets. The Leacock-Chodorow Similarity (L/C) [247] utilizes the distance of synsets based on hyponymy relations. Similarly, the Resnik Similarity (RES) [248] is based on the lowest common ancestor of two synsets in the hyponomy hierarchy. The deeper the lowest common ancestor is located in the hierarchy, the more semantically related the according synsets are. The Wu-Palmer Similarity (W/P) [249], the Jiang-Conrath Similarity (J/C) [250] and the Lin Similarity (LIN) [251] also consider the depth of lowest common ancestor. But, in contrast to RES they further consider the depth of the two synsets. All three measures combine the three depth values based on different mathematical formulas.

The Hirst-St.Onge Similarity (H/S) [252] is based on a graph distance. It determines the shortest path between two synsets and considers all possible relations. The measure takes the distance of this path into account and penalizes turns in the path. In essence, a turn occurs when a relation is followed by an opposite relation, e.g., when a generalization relation is followed by a specialization relation.

The second set of semantic word similarity functions exploits syntagmatic sense relations between words. Basically, such relations are defined upon statistical measures regarding co-occurrences of words in a corpora. The rationale is that the more often two words occur in the same context, the higher their semantic relatedness. The *contextual similarity* [253, 254] is such a measure. Given two words, it is defined as the cosine of the angle between the context vectors of these words. The context vectors are determined with regard to a set of context words. For each of the context words there is a vector element. Such an element represents the co-occurrence count of the context word and the word the vector is defined for. Consequently, words that tend to occur in the same contexts will have a contextual similarity close to 1 and are considered to be syntagmatically related.

To apply the contextual similarity to business process model matching, a corpus as well as a way to determine the context vectors must be defined. Regarding the former,

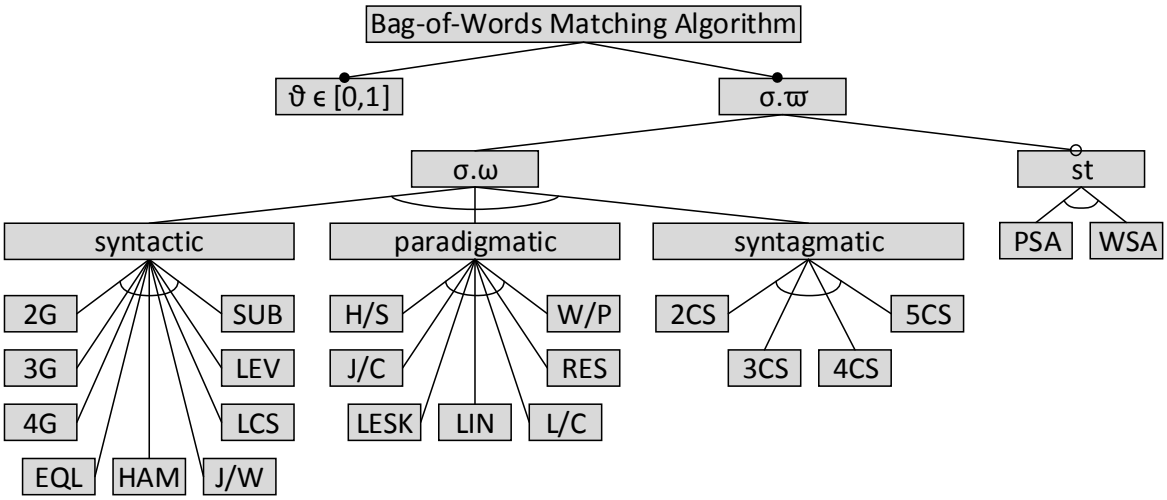

FIGURE 4.4: The extended feature model for the bag-of-words matching algorithm

the model collections are used as corpora in this thesis. That is, each activity label from each model is taken as a text document and added to the corpus. The rationale is that the model collections reflect the domain characteristics.

Given this corpus the following strategy is applied to determine the context vectors for a pair of words. First, the set of the n most frequently co-occurring context words is determined for each of the two words. Then, these two sets are merged and the resulting set comprises the elements of the context vectors. For each of the two words the corresponding context vector then contains the co-occurrence counts for the word and the context vector elements. Finally, the cosine of the angle between these two context vectors constitutes the similarity score for the two words.

Based on the number of context words n that are considered for each word, different contextual similarities can be defined. In this thesis, numbers on the interval $[2, 5]$ are considered as possible values for n. Accordingly, the set of syntagmatic word similarity functions comprises the Two Words Contextual Similarity (2CS), the Three Words Contextual Similarity (3CS), the Four Words Contextual Similarity (4CS), and the Five Words Contextual Similarity (5CS).

All these word similarity measures add further configuration options to the bag-of-words matching algorithm as shown in Figure 4.4. To assess the effect of relying on the semantic word similarities, these functions are also evaluated on the development datasets. In this regard, stemming is deactivated in order to enable an unbiased comparison with

TABLE 4.8: Effectiveness of the semantic word similarities

	BR				*UA*			
$\sigma.w$	ϑ	pr_μ	re_μ	F_μ	ϑ	pr_μ	re_μ	F_μ
J/C	.517	.546	.342	.421	.551	.488	.309	**.378**
LESK	.556	**.627**	.300	.406	.545	.455	**.316**	.373
LIN	.521	.519	.358	.424	.667	.505	.288	.367
W/P	.621	.425	.443	.434	.877	**.777**	.217	.339
2CS	.812	.444	.442	.443	1.00	.688	.166	.267
3CS	.783	.469	.445	.457	.917	.322	.245	.278
4CS	.769	.475	**.449**	**.461**	.897	.345	.252	.292
5CS	.761	.475	**.449**	**.461**	.904	.391	.241	.298

the string similarities. Moreover, the threshold parameter is optimized the same way as it was in the previous evaluations. Table 4.8 summarizes the paradigmatic and syntagmatic word similarities that perform best with regard to the micro f-measure.

Contrary to the presumed necessity for a semantic comparison of words, the considered semantic word similarities do not improve the effectiveness of the bag-of-words matching algorithm. In fact, the paradigmatic word similarities yield lower micro f-measures than the best string similarity function on both datasets. Here, W/P achieves a micro f-measure of .434 ($<$ .461) on BR . Similarly, J/C is the best performing paradigmatic word similarity on UA with a micro f-measure .378 ($<$ .417).

With regard to the syntagmatic word similarities the results are different. 4CS and 5CS achieve the same micro f-measure like HAM (.461) at a higher recall (.449 $>$ .387). In contrast, the performance of these similarities is poor on UA (max = .298). This is not only worse than the performance of all syntactic similarities in combination with the bag-of-words matching algorithm, but it is also worse than the performance of the majority of the syntactic similarities in combination with the basic label matching algorithm. Here, only EQL results in a lower micro f-measure of .268.

These findings indicate that the incorporation of universal word similarity functions does not guarantee a high effectiveness. Instead, they seem to be inappropriate for a

general application as the evaluation revealed a low performance in comparison to the string similarities on the development datasets. The reason is that the paradigmatic word similarities do not integrate domain specific knowledge instead they rely on WordNet, a dictionary for Standard English. In contrast, the syntagmatic similarities are based on domain specific knowledge which is derived from the model collections. However, as they exploit occurrence statistics their quality depends on the availability of a sufficient amount of text. In this regard, the evaluation results indicate that model collections can generally not be considered to comprise enough data. These shortcomings are discussed in more detail in Section 4.6.

4.4 Label Specificity

A further problem that label-based matching techniques face is the varying *label specificity* within model collections. Here, label specificity refers to the level of detail of a label. In general, it is assumed that the higher the specificity, the more precise the information; and the lower the specificity, the more abstract the information.

A factor that influences the label specificity was already discussed in the context of the semantic relations between words in Section 4.3. Hyponymy and meronymy relations between words provide modelers with options to choose from a variety of words with different levels of abstractions. For example, the label “check application” is more precise than “check request”. The reason is that “request” is an abstract term that in different contexts might refer to other concepts, e.g., a request for money. Moreover, “evaluate application” is less specific than “evaluate cv” as the latter addresses the part of the application which is relevant for the evaluation. Whereas such relations are addressed by the integration of paradigmatic word similarities, another factor that influences the label specificity has not been addressed yet. This factor is the length of labels. The rationale is that the more words a label contains, the more specific is its information.

To examine the variety of the label length within model collections, Figure 4.5 shows the number of labels with a certain label length within the development datasets. In this thesis, the length of a label corresponds to the cardinality of its bag-of-words. According

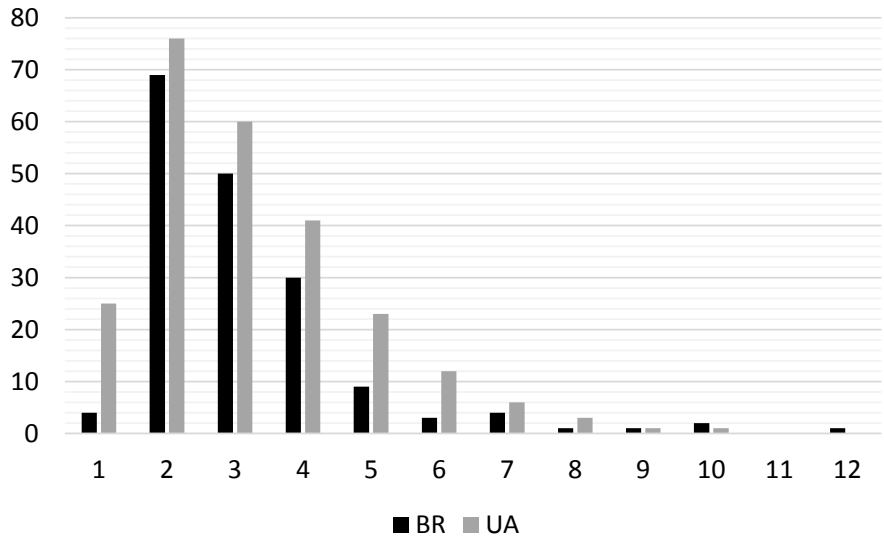

FIGURE 4.5: Distribution of the label length

to the figure, the label length can take a broad range of values within model collections. In this regard, the distribution of the label length in BR is comparable to that in UA. Most of the labels consist of two words and labels with a length of three and four words rank second and third, respectively. The average label length on both datasets is 3.2, whereas the longest label in BR comprises twelve and on UA ten words. Moreover, there are also labels with a length of one in both datasets. Overall, a huge percentage of activity pairs in both datasets is characterized by different lengths of the labels. On BR 64.5% of the activity pairs and on UA 71.7% are impacted.

Differences in the label length can arise from inconsistent labeling styles. As outlined at the beginning of Section 4.2, there are precise labels which contain the action, the object, and additional information. In contrast, other labels are abstract and simply describe the basic action. The impact of a varying label length was already illustrated by the exemplary application of the bag-of-words similarity in Table 4.6. That is, even small differences, e.g., a label contains one word more than the other, can lower the overall bag-of-words similarity score. That is because some words from the longer label typically do not have a counterpart in the shorter label, e.g., the condition “if requirements are met” of the label “accept application if requirements are met” is not represented in the label “accept application”. Yet, those words are considered in the calculation of the bag-of-words similarity and lead to a low similarity score.

To overcome this problem, the matching technique proposed by Leopold et al. [194] separates the words referring to the action from those referring to the object or the additional information by applying the algorithm from [39, 195]. During the computation of label similarity scores only words that belong to the same class are compared. However, natural languages provide versatile options to express the functionality of activities as illustrated by the labels "email application" and "apply via email". Clearly, the meanings are very similar – if not the same – and the same terminology is used in both labels. Yet, the strategy by Leopold et al. [194] yields a similarity score of 0. The reason is that "email" is used to express the action in the first label, whereas it is the object in the second label. Similarly, "application" is the object in the first and "apply" the action in the second label. As this example illustrates, aspects relevant to the assessment of the similarity of activities can be encoded in different label fragments. Thus, the decomposition of labels can be misleading and is considered as inappropriate to solve differences in label specificity.

The label length is also impacted by the use of *collocations* which are arbitrary and recurrent word combinations [255]. Examples include "letter of acceptance" as a specialization of "letter" or "make a decision" as a synonym of "decide". There is a broad range of approaches to the automated extraction of collocations from documents. The comprehensive overview by Seretan [256] served as a basis for the following summary of the field. The determination of collocations is basically carried out in two steps. First, a list of candidates is derived from the documents and the list might be filtered to reduce the number of candidates, as amongst others suggested by Justeson and Katz [257]. Therefore, the words in a text are annotated with *part-of-speech tags*, i.e., with a syntactical word class, like noun, verb, adjective etc. This can be automatically achieved by *part-of-speech taggers* such as those from [258, 259]. Then, only word combinations which adhere to a promising part-of-speech pattern are further considered. Similarly, *part-of-speech parsers* can be used to annotate the words in texts. In contrast to the part-of-speech taggers, parsers also account for syntactical links between words when annotating a text. Thus, their results are considered more reliable [256]. Examples of parsers include those presented in [260, 261]. Once the list of candidates is determined, the second step deals with the inspection of the candidates in order to identify collocations. At this point,

statistical tests based on frequency and co-occurrence counts are carried out to verify that a word combination is actually a collocation. In this regard, the z-score [262], the log-likelihood ratio [263], and the pointwise mutual information [264] were suggested.

The integration of such extraction methods could be used to adjust the levels of abstraction of two labels. For example, when comparing "send letter" and "send letter of acceptance", "letter of acceptance" could be considered as a specialization of "letter" in the first label. This way, the overall similarity score for the labels would be increased and the true relation between the two activities would be better reflected. However, there are also problems connected to the implementation of this idea. Collocation extraction methods are generally not able to detect less frequent collocations [265]. Moreover, they require corpora of a sufficient size in order to produce reliable results. However, business process model collections comprise a rather small amount of short texts and can thus be considered as insufficient corpora. In this regard, the evaluation results of the paradigmatic word similarities indicated that the amount of text in model collections is typically too low to yield reliable results (cf. Section 4.3).

Even if such extraction methods could reliably detect collocations, they would not solve the label specificity problem entirely. Consider the label "notify applicant of acceptance in writing" in which "notify in writing" constitutes a collocation. When this label is compared to the label "send letter of acceptance", "notify in writing" should be matched to "send letter" and "acceptance" should be matched to "acceptance", as these word groups express the same meaning. Yet, there is still a difference in the specificity or label lengths, respectively. That is because "notify applicant of acceptance in writing" contains the word "applicant" while "send letter of acceptance" does not.

Due to these reasons the use of methods for collocation extraction is not further pursued here. Instead, a different approach to the harmonization of the label length is taken. It is referred to as *pruning*. Here, the idea is to remove those words from the longer label that are considered to not have a counterpart in the shorter label. Once these words were removed, the bag-of-words similarity is computed. To implement this idea, the pruning function $prune$ is introduced. It takes two bag-of-words and cuts the first bag-of-words to the size of the second, in case the first is larger than the second.

Definition 4.8 (Pruning) *Given two bag-of-words ϖ,ϖ', a pruning function $prune$ is defined as:*

$$prune(\varpi, \varpi') := \begin{cases} \varpi & \text{if } |\varpi| \leq |\varpi'| \\ \varpi^* & \text{else} \end{cases}$$

where ϖ^* is a subset of ϖ that must have the same cardinality as ϖ' and all words in ϖ^* must appear in ϖ, i.e., $|\varpi^*| = |\varpi'| \wedge supp(\varpi^*) \subseteq supp(\varpi)$.

Based on the pruning function the *bag-of-words matching algorithm with pruning* is introduced in Algorithm 4.3. It is a refined version of the bag-of-words matching algorithm and also iterates over the set of all activity pairs (lines 2 to 15). For each pair it harmonizes the label of each activity (lines 3 and 6) and determines the according bag-of-words (lines 4 and 7). In contrast to the bag-of-words matching algorithm it then unifies the label length by applying the pruning function to the bag-of-words (line 8 to 9). Here, the pruning function needs to be applied twice as each of the two bag-of-words could be the larger one. Based on the pruned bag-of-words the bag-of-words similarity is computed (line 10) and the score is compared to the threshold (line 11) in order to decide if the according activity pair is added to the alignment (line 12).

In addition to the specific word similarity and the stemming functions used to calculate the bag-of-words similarity scores, the effectiveness of the bag-of-words algorithm with pruning also depends on the pruning function applied to harmonize the length of labels. In particular, three pruning functions are considered in the following. At heart, all three functions follow the same procedure. To prune a bag-of-words, the functions first transform it into a list. Therefore, each word is added to the list as often as it occurs in the bag-of-words. Next, the list is sorted in descending order with regard to one or more criteria. At this point, the functions distinguish themselves from one another by relying on different sort keys. Finally, the pruned bag-of-words is created. Therefore, the first n words of the sorted lists are selected and added to the pruned bag-of-words. Here, n is equal to the number of words in the shorter bag-of-words.

The Maximum Pruning Function (MaxPF) utilizes the word similarity and stemming functions. For each word from the larger bag-of-words it determines the maximum word

Algorithm 4.3: Bag-of-words matching algorithm with pruning

Input: $P = (N, E, \lambda, \tau, A)$, $P' = (N', E', \lambda', \tau', A')$

Output: $\mathscr{A}$

```
1  𝒜 = ∅;
2  foreach a ∈ A do
3      label = norm(λ(a));
4      ϖ = tok(label);
5      foreach a' ∈ A' do
6          label' = norm(λ'(a'));
7          ϖ' = tok(label');
8          ϖ_p = prune(ϖ, ϖ');
9          ϖ'_p = prune(ϖ', ϖ);
10         similarity = σ.ϖ(ϖ_p, ϖ'_p);
11         if similarity ≥ ϑ then
12             𝒜 = 𝒜 ∪ {(a, a')};
13         end
14     end
15 end
```

similarity score yielded by comparing the word and its stems to each of the words and their possible stems in the smaller bag-of-words. The maximum scores are then taken as the sort criteria.

The other two pruning functions are inspired by the *term frequency / inverse document frequency* weighting which is used to assess the relevancy of a word for a given document within a document collection [266]. It takes the term frequency into account, i.e., the number of occurrences of a word in a document. That is, the higher the number of occurrences is, the more relevant is the word for the document. However, words that frequently occur in the collection have no or only little discriminating power because all documents would be similarly relevant with regard to these words [30]. Consequently, such words distort the ranking of the documents. Therefore, the inverse document frequency [267] is used to weight the term frequency. The inverse document frequency is inversely proportional to the number of documents that contain the word. That means, the higher the number of documents in which the word occurs, the less important it is.

In this spirit, the Frequency Pruning Function (FreqPF) is suggested as a pruning function. Here, words are ranked with regard to their frequency in the model collection. The frequency of a particular word is the number of activities in the model collection whose label contains the word. Then, the words are ranked in descending order with regard to their frequency or relevance, respectively. If two words have the same frequency, the maximum similarity is used as a second sort criteria. In contrast to the inverse document frequency this strategy favors frequently occurring words over rare words. That is because contrary to information retrieval pruning is applied to balance the label specificity by bringing the more specific label to the specificity level of the more abstract label. Hence, the focus is on relevant aspects, e.g., the main actions, rather than less relevant aspects, e.g., conditions related to a small number of actions.

As opposed to FreqPF, the Co-occurrence Pruning Function (CoPF) does not consider the relevance of words with regard to the entire collection. Instead, the current context is focused, i.e, the smaller bag-of-words. Thus, words that are more likely to co-occur in the same context as those in the other bag-of-words are selected. Therefore, the words are ranked with regard to their overall co-occurrence count. For a specific word from the larger bag-of-words this count is calculated by summing up the co-occurrence counts of the considered word and all words from the shorter bag-of-words. Here, the co-occurrence counts are equal to those in the context of the syntagmatic word similarities (cf. Section 4.3) which are defined with regard to the entire model collection. Similar to FreqPF, CoPF utilizes the maximum similarity as a subordinate sorting criteria.

As a consequence of integrating pruning into the matching process, the space of possible configurations grew again. As shown in Figure 4.6 the feature model now contains the additional prune feature which can be activated or not. Similar to the deactivation of stemming the deactivation of pruning can be achieved through the implementation of a function that always returns the first bag-of-words without pruning it. To enable pruning one of the three options needs to be selected.

Like the other matching algorithms, the bag-of-words matching algorithm with pruning is evaluated on the development datasets. To assess the effect of the pruning function, stemming is neglected. Similar to the evaluation of the stemming algorithms HAM and

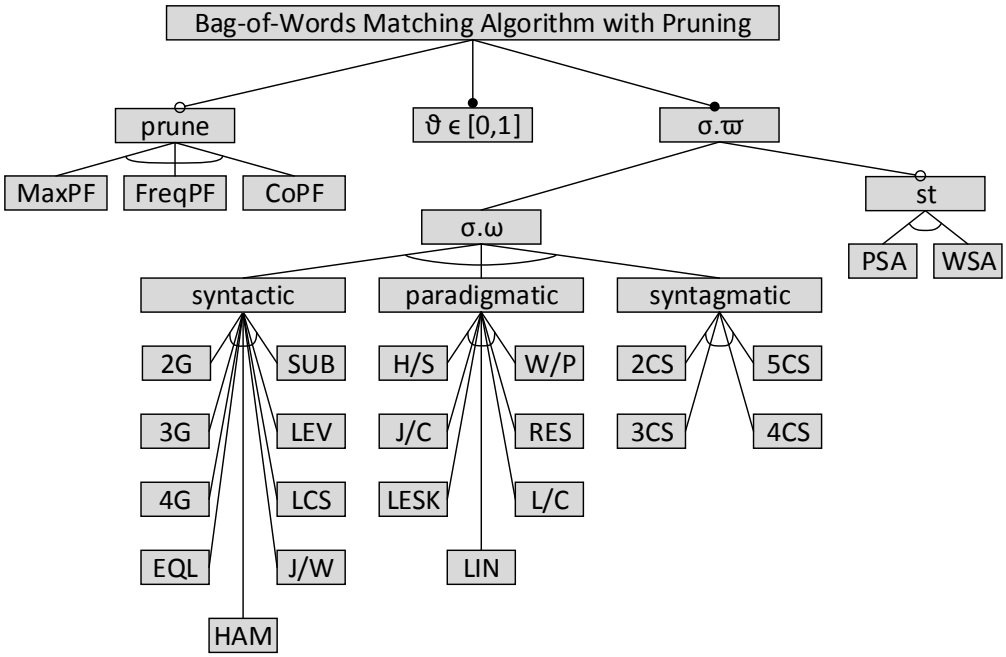

FIGURE 4.6: The feature model for the bag-of-words matching algorithm with pruning

SUB are chosen as the best performing word similarity functions in combination with the bag-of-words matching algorithm. Table 4.9 summarizes the effectiveness.

With regard to BR applying MaxPF or FreqPF yields a higher micro f-measure for both word similarity functions, whereas CoPF only increases the effectiveness for SUB. The maximum micro f-measure of .474 is the result of combining HAM with MaxPF. This however is only a marginal improvement, as HAM yields a micro f-measure of .461 without pruning. In this regard, MaxPF also leads to the maximum f-measure of .466 for SUB. This value constitutes a stronger improvement (.466 vs. .414).

This slightly positive effect is not confirmed by the results on UA. Similar to BR, MaxPF yields higher values for both word similarities than FreqPF and CoPF. Yet, the micro f-measure for MaxPF in combination with HAM is .401 and with SUB .414. These values are slightly lower than those yielded without pruning (HAM: .405; SUB: .417).

The evaluation results show that similar to the stemming functions, pruning does not significantly improve the effectiveness. Instead it can even decrease the overall effectiveness in terms of the micro f-measure. Another finding is that MaxPF is to be preferred over FreqPF and CoPF. The reason is that MaxPF yields the highest micro f-measures on both datasets for both word similarity functions.

TABLE 4.9: Effectiveness of the bag-of-words matching algorithm with pruning

		BR				*UA*			
st	$\sigma.w$	ϑ	pr_μ	re_μ	F_μ	ϑ	pr_μ	re_μ	F_μ
MaxPF	HAM	.598	.546	.420	**.474**	.764	.484	.343	.401
	SUB	.641	.478	**.455**	.466	.748	.429	**.401**	**.414**
FreqPF	HAM	.571	.518	.430	.470	.792	.632	.275	.383
	SUB	.643	.508	.394	.444	.785	.554	.299	.389
CoPF	HAM	.583	**.564**	.384	.457	.783	.564	.298	.390
	SUB	.654	.539	.378	.445	.764	**.648**	.267	.379

4.5 The Bag-of-Words Matching Technique

In this section the Bag-of-Words Technique (BOT) is finally introduced. It consists of two parts. First, there is the *bag-of-words matching algorithm with pruning and filtering* which is based on the algorithms from the previous sections. Second, it comprises a set of features which provide configuration options. In this regard, the most promising options are considered, whereas the options for which poor results were obtained in the previous analyses are discarded.

The bag-of-words matching algorithm with pruning and filtering is an extended version of the bag-of-words matching algorithm with pruning. Its basic structure is shown in Algorithm 4.4. From an abstract point of view, the algorithm can be divided into a sequence of two steps. The first step (lines 2 to 11) filters activity pairs. It is optional and can be deactivated by setting the *filter* variable to "false" (line 2). The second step (lines 12 to 25) iterates over the remaining activity pairs and applies the bag-of-words similarity in combination with the pruning function to classify the activities.

In more detail, the filtering step searches the set of all activity pairs for equally labeled pairs. Whenever such a pair is found (line 5), it is considered as a correspondence and added to the alignment (line 6). This is based on the finding that equal labels are a precise correspondence indicator. As shown in Section 4.1, 85.5% of the activity pairs with equal labels actually correspond on BR and 78.2% on UA. Thus, assuming equally labeled

Algorithm 4.4: Bag-of-words matching algorithm with pruning and filtering

Input: $P = (N, E, \lambda, \tau, A)$, $P' = (N', E', \lambda', \tau', A')$

Output: $\mathscr{A}$

```
1  𝒜 = ∅; A_= = ∅;
2  if filter then
3      foreach a ∈ A do
4          foreach a' ∈ A' do
5              if norm(λ(a)) = norm(λ'(a')) then
6                  𝒜 = 𝒜 ∪ {(a, a')};
7                  A_= = A_= ∪ {a, a'};
8              end
9          end
10     end
11 end
12 foreach a ∈ A \ A_= do
13     label = norm(λ(a));
14     ϖ = tok(label);
15     foreach a' ∈ A' \ A_= do
16         label' = norm(λ'(a'));
17         ϖ' = tok(label');
18         ϖ_p = prune(ϖ, ϖ');
19         ϖ'_p = prune(ϖ', ϖ);
20         similarity = σ.ϖ(ϖ_p, ϖ'_p);
21         if similarity ≥ ϑ then
22             𝒜 = 𝒜 ∪ {(a, a')};
23         end
24     end
25 end
```

activities to correspond results in only a small amount of false positives. But, the filtering goes further and also uses equally labeled activity pairs as an exclusion criteria. That is, activities that have an equally labeled counterpart in the other process are considered totally dissimilar from the remaining activities in the other process. Accordingly, activities that occur in an equally labeled activity pair are stored (line 7) and all activity pairs

that contain one of these activities are not considered in the second step (lines 12 and 15). This is based on the assumption that equal labels are a typical characteristic for elementary correspondences which by definition comprise activities that do not correspond to any other activity (cf. Section 3.1). No analyses carried out so far provides evidence towards this assumption. Thus, in addition to the precision values, the impact of removing activity pairs from the set of possible correspondences is analyzed here with regard to the development datasets. On BR 46 correspondences are excluded from the classification in the second step when filtering is activated. From a relative perspective, this conforms to 1.53% of the excluded activity pairs and to 7.88% of all correspondences. On UA there are only two correspondences excluded being equivalent to .05% of all excluded pairs and .38% of all correspondences. These overall low values show that only a small number of correspondences are missed, if equally labeled activity pairs are considered to be elementary correspondences. Thus, evidence towards the assumption is given and filtering might be used to reduce the search space for the second step. That is, a huge share of truly non-corresponding activity pairs is already correctly classified in the filtering step. Consequently, the filtering is also a strategy to reduce the amount of false positives.

The second step relies on the bag-of-words model to compute similarity scores for activities instead of applying string similarities to the whole label. This decision is made because the bag-of-words model allows for a more fine-grained and accurate calculation of similarity scores. Evidence in this regard is given by the evaluation results of the bag-of-words matching algorithm which achieves higher micro f-measures than the basic label matching algorithm (cf. Section 4.2). Here, additional support for the decision is presented in terms of Receiver Operating Characteristic (ROC) and Precision Recall (PR) curves. These curves are well-known in the field of information retrieval [30] and provide means to inspect the development of the effectiveness with regard to different configurations of matching algorithms.

ROC curves show the development of the true positive and the false positive rate. While the true positive rate is the micro level recall, the false positive rate measures how many of the non-corresponding activities were falsely suggested as correspondences. For each possible configuration the true positive and false positive rate are determined

and plotted as a curve. Here, the x-axis represents the false positive and the y-axis the true positive rate. Thus, the curve shows to which degree a change in the technique's configuration that increases the number of correctly detected correspondences is linked to an increase in the number of falsely suggested correspondences. Effective matching techniques are typically characterized by large true positive rates for all possible values of the false negative rate. Hence, the larger the area under the curve, the better is the matcher suited for activity pair classification.

However, the ROC curve is known to present an optimistic view on the effectiveness of matchers in case there is a huge difference in the number of corresponding and non-corresponding activity pairs [268]. As this is usually the case for business process model collections as outlined by the descriptive statistics for the datasets in Section 3.4, PR curves are also investigated here. A PR curve outlines the tendency to which the precision of a matching technique decreases, when it is configured to yield a certain recall. Therefore, the micro recall and the micro precision values are calculated for all configurations of a matching technique. Then, the micro level recall is sampled in equal steps on the interval of $[0, 1]$. Here, the step size is .01 resulting in 101-point PR curves. For each of the sampled recall values the highest precision value that was yielded together with a recall value equal to or higher than the sampled value is determined. Then, the curve plots the sampled recall values on the x-axis and the respective precision values on the y-axis. Similar to the ROC curve, effective matching techniques are characterized by large precision values for all possible values of the recall. Thus, the larger the area under the curve, the better is the technique suited for matching.

Figures 4.7 and 4.8 contrast the ROC and PR curves for the basic label and the bag-of-words matching algorithms on both development datasets. Here, both algorithms were configured with HAM on BR and SUB on UA as these are the word similarities that yielded the highest micro f-measure for the bag-of-words matching algorithm on one of the datasets. Moreover, stemming is neglected and the threshold is the only parameter that is varied for both algorithms. As the figure reveals, all curves for the bag-of-words matching algorithm cover a larger area than those for the basic label matching algorithm. Consequently, the bag-of-words algorithm is better suited as an increase in the recall is

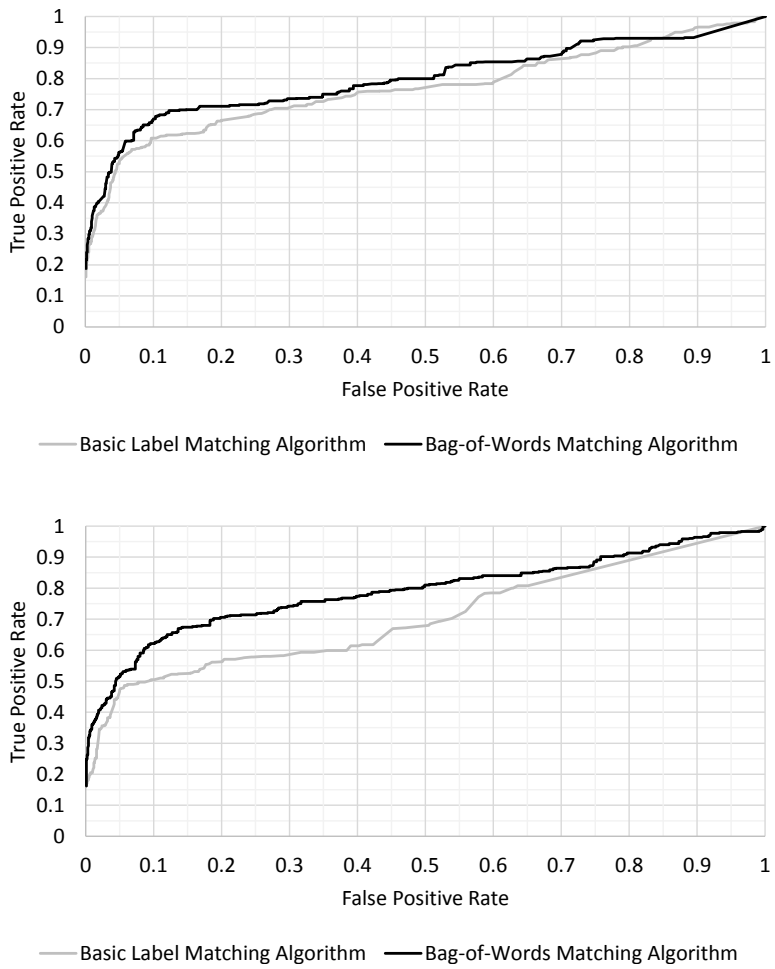

FIGURE 4.7: ROC curves on BR (top) and UA (bottom)

connected with a smaller decrease in the precision and thus also with a smaller increase in the false positive rate. In other words, when the bag-of-words matching algorithm is configured to yield a certain micro recall it is likely to propose less false positives than a configuration of the basic label matching algorithm that achieves the same micro recall.

BOT's features are shown in Figure 4.9. Besides the filter feature which can be selected or not, BOT's features are oriented towards the ones introduced in the previous sections. However, BOT does not comprise all of these features. Instead, only those features that showed positive effects on the effectiveness are considered. In the following, the selection of features is briefly discussed.

First, the threshold ϑ is used to cut off activity pairs that are considered dissimilar. Here, all values in the interval $[0, 1]$ can be chosen for the threshold.

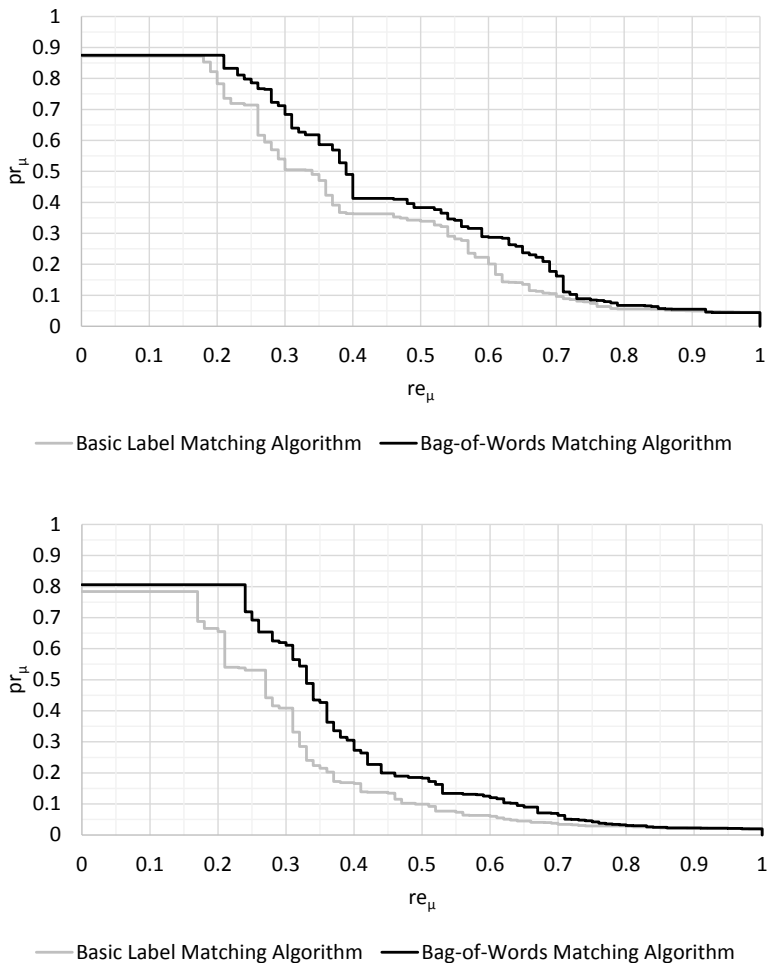

FIGURE 4.8: PR curves on BR (top) and UA (bottom)

Additionally, a variety of syntactic, paradigmatic, and syntagmatic word similarity functions was introduced. In total there are 20 different functions. However, their effectiveness varies and some of them yield low micro f-measures. Thus, the number of possible functions is reduced to twelve. First, HAM, LCS, LEV, and SUB are proposed as the syntactic similarities because they yield higher micro f-measures on the development datasets than the other functions from this group. Similarly, J/C, LESK, LIN, and W/P are the representatives of the paradigmatic similarity functions. Finally, all four syntagmatic similarity functions are considered because their effectiveness is quite similar and none of the functions clearly outperformed the others in the evaluations.

As shown in Section 4.2 the stemming functions do not have a significant impact on the effectiveness. Moreover, there is also no significant difference between PSA and WSA. Consequently, only PSA is considered as an option which can be enabled or disabled. The

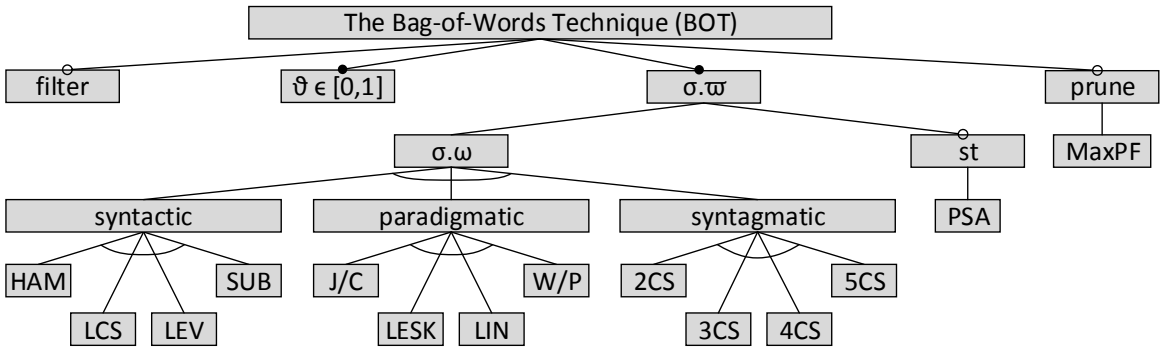

FIGURE 4.9: The feature model for the Bag-of-Words Technique

reason for neglecting WSA is that it is limited to English as it is based on WordNet. In contrast, PSA is available for different languages[28].

Finally, similar to stemming, pruning is optional as it might lead to an increase or decrease in the effectiveness. Moreover, only MaxPF is considered, because CoPF and FreqPF performed worse than MaxPF.

4.6 Evaluation and Analysis

To conclude the discussion of label-based process model matching and the verification of Sub-hypothesis H2, this section evaluates and analyzes BOT. First, BOT is evaluated on the development datasets and a default configuration of BOT is derived. Such a configuration enables the direct application of BOT without the need to manually configure it. Next, the effectiveness of this default configuration is examined with regard to the evaluation datasets. These results provide insights into BOT's general effectiveness as well as into its limitations. Furthermore, the use of the default configuration is contrasted to a semi-manual configuration approach. In this approach experts provide alignments for a subset of the model collection. From these alignments the best performing configuration is derived and used to match the remaining model pairs in the collection. Finally, a challenge analysis is presented. This analysis explicates problems regarding the identification of correspondences based on BOT. Thus, it also provides guidance for future work on the label-based matching of process models.

[28] http://snowball.tartarus.org, accessed: 13/01/2017

4.6.1 Effectiveness on the Development Datasets

In Section 4.5 the configuration space of BOT was cut by removing features which had little influence on the effectiveness of the discussed matching algorithms. However, without considering the possibility to add further features in future work, the configuration space of BOT is still large. That is, filtering, pruning, and stemming can be enabled or disabled. Additionally, one of twelve word similarities needs to be chosen. Accordingly, there are $(2 \times 2 \times 2 \times 12 =)$ 96 options to determine how BOT calculates similarity scores for activity pairs in a model collection. Moreover, the threshold parameter needs to be set to a specific value in order to split the activity pairs into corresponding and non-corresponding pairs based on the calculated similarity values. If, for example, the interval of possible threshold values $[0, 1]$ is sampled in steps of .05, there are 21 different threshold values and consequently $(96 \times 21 =)$ 2016 configurations of BOT.

With that in mind, the maximum effectiveness achieved by any BOT configuration on the development datasets is determined. This results provides further insights into BOT's effectiveness. Moreover, it serves as a baseline for the selection of a default configuration that can be directly applied by experts without additional configuration efforts. To this end, three configurations are considered: the maximum effectiveness of BOT_{BR} and BOT_{UA} as the best performing configurations on each dataset and BOT_{ALL} which was optimized on the union of both datasets. Table 4.10 summarizes the features and the effectiveness of the three configurations.

While BOT_{BR} and BOT_{UA} yield the highest values on the according datasets, they also yield the lowest effectiveness on the other dataset. In particular, the effectiveness of BOT_{BR} on UA is drastically lower than this of BOT_{UA}. In contrast, BOT_{ALL} ranks second on

TABLE 4.10: Effectiveness of the optimized BOT configurations on BR and UA

	Options					*BR*			*UA*		
Matcher	*filter*	$\sigma.w$	st	$prune$	ϑ	pr_μ	re_μ	F_μ	pr_μ	re_μ	F_μ
BOT_{BR}	true	2CS	PSA	-	.859	.652	**.452**	**.534**	.095	.460	.157
BOT_{UA}	true	J/C	PSA	MaxPF	.577	.611	.301	.404	.406	**.486**	**.442**
BOT_{ALL}	true	HAM	-	-	.550	**.657**	.344	.452	**.429**	.380	.403

both datasets and achieves the highest average micro f-measure (BOT_{ALL}: .428; BOT_{BR}: .346; BOT_{UA}: .423). Due to the better average performance BOT_{ALL} is proposed as BOT's default configuration.

The results also outline two problems related to universal label-based matching techniques. First, the effectiveness of a specific configuration usually varies across datasets. Here, adapting BOT to the characteristics of one dataset and then applying the respective configuration to other datasets typically results in a poor performance with regard to the maximum effectiveness. This could be observed for BOT_{BR} and BOT_{UA} as well as for the configurations of the matching algorithms throughout this chapter. Moreover, the effectiveness of BOT_{ALL} does not vary that strongly, but is still outperformed by the configuration with the maximum effectiveness. The reason is that the domain characteristics of model collections vary and are reflected differently by the configurations. This problem has also been recognized in the area of schema and ontology matching [171, 269, 270].

Second, the low effectiveness shows that the domain characteristics are not represented sufficiently by BOT and its universal features. That this is a general problem of label-based matching techniques is on the one hand substantiated by the consideration of state-of-the-art techniques from natural language processing, ontology matching, and information retrieval. In this chapter a broad variety of such approaches has been discussed and analyzed. However, a high effectiveness could not be achieved. On the other hand, the comparison to the best performing techniques from the process model matching contests in 2013 and 2015 [19, 20] reveals that other techniques also struggle with the assessment of the label similarity. Table 4.11 contrasts the results of the best techniques from the contests and the results of the three BOT configurations. As the publication from the first contest [19] only reported the macro level effectiveness of the techniques, the macro and the micro effectiveness are outlined in the table.

The results reveal that BOT's maximum effectiveness outperforms the state of the art. That is, BOT_{BR} and BOT_{UA} yield higher micro and macro f-measures than the techniques from the contests. Moreover, the default configuration (BOT_{ALL}) yields results comparable to that of the state of the art. To this end, its micro f-measure is virtually identical to that of pPalm-DS on BR (.452 vs. .459). The macro level effectiveness is worse than that

TABLE 4.11: Effectiveness of BOT and the contest matchers [19, 20] on BR and UA

Dataset	*Matcher*	pr_M	re_M	F_M	pr_M	re_M	F_M
BR	BOT_{BR}	.652	**.452**	**.534**	.633	**.467**	**.511**
	BOT_{ALL}	**.657**	.344	.452	.615	.329	.382
	RMM/NSCM	-	-	-	.68	.33	.45
	pPalm-DS	.502	.422	.459	.499	.429	.426
UA	BOT_{UA}	**.406**	**.486**	**.442**	.443	**.511**	**.453**
	BOT_{ALL}	.380	.403	.428	**.455**	.386	.382
	RMM/NSCM	-	-	-	.37	.39	.38

of pPalm-DS and RMM/NSCM (.382 < .426, .382 < .45). Yet, the significance of this observation is limited, as the macro f-measure tends to draw a distorted picture of the effectiveness (cf. Section 3.1). On UA RMM/NSCM and BOT_{ALL} also achieve virtually identical macro f-measures (.382 vs. .38). These results show that in comparison to the state of the art BOT can be considered as a high performing matching technique. Moreover, as both techniques from the contests also solely exploit labels (cf. Section 3.3.3), the results further substantiate that label-based matching techniques suffer from a generally low effectiveness.

4.6.2 Effectiveness on the Evaluation Datasets

To examine the general validity of the findings, the evaluation datasets are used to assess the effectiveness of BOT. Besides the default configuration BOT_{ALL}, the other two optimized configurations from the development datasets (BOT_{BR}, BOT_{UA}) are considered here as well. Additionally, the top performing BOT configurations on each of the two evaluation datasets (BOT_{SR}, BOT_{AW}) are determined and serve as a baseline. Furthermore, as the SR dataset was used in the second process model matching contest [20], the results of the BOT configurations are compared to AML-PM, the best performing technique on this dataset. Table 4.12 presents the respective results.

TABLE 4.12: Effectiveness of BOT and the contest matcher [20] on SR and AW

	SR			AW		
Matcher	pr_μ	re_μ	F_μ	pr_μ	re_μ	F_μ
BOT_{BR}	.606	.590	.598	.519	.285	.368
BOT_{UA}	.750	.581	.655	.510	.333	.403
BOT_{SR}	**.887**	.568	**.692**	.947	.240	.383
BOT_{AW}	.227	**.608**	.330	.616	**.552**	**.582**
BOT_{ALL}	.774	.572	.658	**.959**	.251	.397
AML-PM	.786	.595	.677	-	-	-

On SR the maximum effectiveness is $F_\mu = .692$ yielded by BOT_{SR}. This high effectiveness value in comparison to the development datasets can be traced back to the increased label homogeneity. For example, 47% of the correspondences on SR have the same labels, whereas on the development datasets only 16% are equally labeled (cf. Table 4.1). Moreover, the effectiveness of the configurations trained on the other datasets falls into the interval of $[.330, .658]$ and BOT_{ALL} yields the highest micro f-measure among those configurations. Finally, the f-measure of BOT_{SR} and BOT_{ALL} is similar to that of AML-PM.

On AW the maximum micro f-measure of .582 yielded by BOT_{AW} is only moderate, but still higher than that on the development datasets. Moreover, the configurations that were optimized on the other datasets perform poorly in comparison to BOT_{AW}. That is because these optimized configurations rely on filtering and thus suggest equally labeled activities as elementary correspondences. However, this appears to be too restrictive for AW where equally labeled activities tend to be part of complex correspondences.

In summary, the findings provide further evidence towards Sub-hypothesis H2. First, the analysis on the evaluation datasets confirms that label-based matching techniques cannot be assumed to yield a high effectiveness on all datasets, as they do not sufficiently reflect the domain characteristics and require a high labeling homogeneity to yield a high effectiveness. Moreover, the results revealed that the effectiveness of matcher configurations that are optimized on some datasets varies with regard to the maximum effectiveness when they are applied to new datasets. This problem of a limited portability

of matcher configurations has also been recognized in the field of schema and ontology matching [171, 269, 270]. Finally, the comparison to the state of the art demonstrated that BOT together with its default configuration BOT_{ALL} is a high performing matching technique. Thus, its results can be considered to be representative of the state of the art substantiating the general validity of the findings.

4.6.3 Semi-manual Configuration

As shown in the preceding evaluations, on each dataset the quality of the default configuration is lower than the maximum. Thus, experts might be interested in configuring BOT in order to maximize its utility. This selection of a configuration (i) requires some ground truth that can be used to estimate the effectiveness and (ii) needs to be repeated whenever the context, i.e., the model collection, changes. With that in mind, the following semi-manual configuration approach is applied to investigate the manual effort needed to improve the effectiveness of the default configuration. First, a part of the model collection is manually matched by the experts. Then, the best-performing configuration on these alignments is automatically determined and used to match the remaining model pairs.

In the experiment, the experts' opinion is simulated by selecting gold standard alignments. That is, on each dataset the 36 model pairs are randomly partitioned into $s = 36/k$ distinct sets of size $k \in \{1, 2, 3, 4, 6, 9\}$. For each k 36 sets are determined by generating $36/s$ partitions. Then, for each of the sets the BOT configuration is optimized, i.e., the configuration with the highest micro f-measure is determined. After that, this configuration is applied to the model pairs that were not used in the optimization. Finally, per k the average f-measure $\overline{F_\mu}$ on the evaluation model pairs is computed as an estimation of the effectiveness that can be achieved by training BOT. Further, the experts' effort is estimated in terms of the average of the number of correspondences $\overline{|\{c\}|}$ and activity pairs $\overline{|\{ap\}|}$ in the training sets: the user needs to correctly identify $\overline{|\{c\}|}$ correspondences from a pool of $\overline{|\{ap\}|}$ candidates. Table 4.13 contrasts the results of the semi-manual configuration approach to the effectiveness of the default configuration (BOT_{ALL}) and the maximum (BOT_{MAX}).

TABLE 4.13: Results of the semi-manual configuration approach

	BR			UA			SR			AW		
k	$\overline{F_\mu}$	$\overline{\lvert\{c\}\rvert}$	$\overline{\lvert\{ap\}\rvert}$	$\overline{F_\mu}$	$\overline{\lvert\{c\}\rvert}$	$\overline{\lvert\{ap\}\rvert}$	$\overline{F_\mu}$	$\overline{\lvert\{c\}\rvert}$	$\overline{\lvert\{ap\}\rvert}$	$\overline{F_\mu}$	$\overline{\lvert\{c\}\rvert}$	$\overline{\lvert\{ap\}\rvert}$
1	.42	16	371	.36	15	746	.46	6	126	.45	10	52
2	.45	32	742	.39	30	1492	.50	12	253	.47	21	104
3	.46	49	1113	.40	44	2238	.60	19	380	.50	31	156
4	.47	65	1484	.40	59	2983	.59	25	507	.52	42	207
6	.48	97	2226	.41	89	4476	.63	37	760	.52	63	311
9	.50	146	3340	.42	133	6713	.63	56	1140	.55	94	467
BOT_{ALL}	.45	-	-	.40	-	-	.66	-	-	.40	-	-
BOT_{MAX}	.53	-	-	.44	-	-	.69	-	-	.58	-	-

The table reveals that even when experts manually align nine model pairs to optimize the BOT configuration they do not reach the maximum effectiveness. However, on BR they need to align two model pairs, on UA three, and on AW only one in order to yield an effectiveness that is higher than that of the default configuration. Only on SR they do not reach the default configuration's effectiveness, but after three model pairs have been matched the effectiveness levels off and it is close to that of the default configuration. These observations suggest that the provision of alignments for (3 out of 36 $\widehat{=}$) 8% of the model pairs will enable experts to yield a configuration that is at least close to and often better than the effectiveness of the default configuration. Yet, given the rather small differences in the f-measure, the potentially huge effort, e.g., for $k = 3$ experts need to identify 40 out of 2238 activity pairs on UA, should be considered by experts before opting for a semi-manual configuration approach.

4.6.4 Challenge Analysis

The overall low and varying effectiveness on all four datasets raises the question why label-based matching techniques struggle with the identification of correspondences. In order to better understand the problems, an analysis of challenges is presented in the following. This analysis also gives further evidence towards Sub-hypothesis H2 and provides guidance for the development of enhanced label-based matching techniques.

TABLE 4.14: Overview of the false positive challenges

Challenge	*BR*	*UA*	*SR*	*AW*	Σ
Equal Labels	16	32	1	0	49
Shared Words	121	182	15	129	447
Stop Word Removal	0	10	0	0	10
No commonalities	4	2	0	0	6
Σ	141	226	16	129	

In particular, it focuses on BOT's misclassifications, i.e., the false positives and the false negatives. The analysis builds upon a representative sample of such misclassifications. Hence, for each dataset it considers the misclassifications of the best performing BOT configuration. Focusing on these misclassifications is a limiting factor because this way the analysis ignores the similarity assessment and the degree to which an activity pair is misclassified. That is, a similarity score close to the threshold can be considered to be less problematic than values with a larger distance. Nevertheless, as all misclassifications are regarded, the analysis is considered to provide a representative overview of the challenges.

The first part of the analysis focuses on the false positives. That means it investigates reasons for the identification of correspondences that do not exist. To this end, all false positives were derived from the datasets and manually classified with respect to the reason of the misclassification. In this regard, the guidelines for qualitative analysis [45, 46] (cf. Section 1.3) were applied. The result of the analysis is a set of four challenges which are discussed in the following. Additionally, Table 4.14 summarizes the frequencies of the challenges in the datasets.

Equal Labels. The analysis revealed that there are false positives with equal labels. In contrast to the label equality similarity function EQL labels were also considered to be equal, if they consisted of the same words regardless of the specific word form used in the label. Consequently, the labels "wait for response" and "waiting for response" are viewed as equal labels. This observation confirms that matching techniques which build upon label equality need to accept exceptions. These exceptions can be due to implicit roles or

to different contexts and positions. However, in comparison to the next challenge this challenge rarely occurs.

Shared Words. The most frequent problem which in total comprises 447 of the 512 false positives ($\widehat{=}$ 87%) refers to situations where some words occur in both labels, but there are also words that occur in only one of the labels. Examples include "create birth certificate" vs. "send birth certificate" and "accept application" vs. "reject application". Here, the words that occur in both labels dominate the determination of the bag-of-words similarity and thus a high similarity score is yielded. This effect is increased, if pruning is enabled because this way more emphasis is put on words with high similarity scores. Additionally, low threshold values, e.g., the threshold of BOT_{UA} is .577, amplify the impact of this problem. A strategy to mitigate the problem is to only consider high similarity values as a correspondence indicator. However, as discussed in the context of the false negative challenges (see below), this strategy might lead to a low recall because the similarity of many truly correspondences is not assessed properly. Accordingly, many true correspondences are ruled out when the threshold is set to a high value.

Stop Word Removal. A challenge that was only observed on UA refers to the removal of stop words and in particular to the removal of "not" from the bag-of-words, e.g., "mark student as not qualified" is transformed into the bag-of-words {"mark", "student", "qualified"}. Here, discarding "not" changes the meaning and thus the label might be matched to labels which actually constitute antonyms like "mark student as qualified". Yet, this problem was only observed ten times.

No commonalities. Finally, the least frequently occurring challenge is that some activities were matched although their labels have nothing in common. That is, labels like "archive documents" and "return to migrantshelter" were proposed as correspondences although their meanings are not related in any sense and the labels do not share any words. This clearly is attributed to a wrong assessment of the senses. However, only six false positives fall into this category making the problem negligible.

TABLE 4.15: Overview of the false negative challenges

Challenge	*BR*	*UA*	*SR*	*AW*	Σ
Holonymy	68	204	73	0	345
Same Holonym	78	39	2	120	239
Generic Activity	88	2	9	12	111
Synonymy	74	87	7	37	205
Case Differentiation	0	0	5	2	7
Filtering	12	6	0	0	18
Σ	320	338	96	171	

The second part of the analysis dealt with the false negatives. Here, reasons why BOT did not propose activity pairs that actually correspond were investigated. Similar to the analysis of the false positives qualitative methods were applied to categorize the challenges. Table 4.15 introduces the six identified challenges and their number of occurrence within the datasets.

Holonymy. The first challenge is the most frequently occurring challenge with regard to the false negatives. It comprises all activity pairs where one of the activities comprises the other activity as it is more generic, e.g., "publishing the letters" vs. "send letter of rejection" and "send letter of acceptance". Hence, this challenge is related to the existence of 1:n-correspondences.

Same Holonym. Similar to the first challenge BOT also often struggles with the identification of m:n-correspondences. Here, two activities might not represent the same functionality, but are part of the same abstract activity. An example is given by the labels "create and add cv" vs. "fill in online form of application" that represent sub-steps of the more abstract activity "apply online".

Synonymy. In contrast to the first two challenges the third challenge refers to elementary correspondences. Some correspondences exist between activities that represent the same functionality, but their labels differ as they also comprise conditions or rely on

different terminology, e.g., “send documents by post” vs. “send application”. Like the other challenges this problem occurs frequently.

Generic Activity. This challenge is linked to the other challenges, but was observed less frequently. In some cases one of the labels indicates a generic task like “adjust” and “selection” which does not provide specific information on the functionality and can thus occur in many different contexts.

Case Differentiation. Some labels represent the same functionality in different contexts. For example, the labels “transfer to ps of time recorded” and “forwarding of time sheet data to cs” represent activities where similar business objects are forwarded to a business unit. Yet, this challenge rarely occurs.

Filtering. Finally, there are those activity pairs that were excluded in the filtering step because at least one of the activities had an equally labeled counterpart in the other process model. As already outlined in Section 4.5 this might lead to the exclusion of correspondences. In the analysis only activity pairs with almost identical labels were assigned to this category. The remaining activity pairs were assigned to the other challenges as BOT would not have identified them anyway.

The analysis of the false negatives reveals that a huge share of the correspondences is not discovered as the sense relations between the labels are not assessed correctly. Accordingly, the similarity scores yielded for the respective activity pairs are low and the pairs are misclassified as non-corresponding. Here, holonymy and synonymy relations as well as generic activities constitute the major challenges in the assessment of the similarity of activities. The low similarity values resulting from this misjudgement of the sense relations pose a problem because lowering the threshold value in order to relax the degree to which activities are considered similar results in many false positives. In this regard, the analysis revealed that the major problem concerning the false positives is that the respective labels often share words that dominate the similarity assessment and lead to high scores. In such situations the influence of the words that are responsible for the different meaning diminishes. In summary, the analysis verifies that universal label-based

matching techniques are likely to yield an insufficient effectiveness. The reason is that such techniques rely on universal knowledge which does not necessarily reflect the domain characteristics of the model collections.

The generalizability of this finding is limited by the number of the considered knowledge sources and similarity measures that were considered in this chapter. Yet, relying on other corpora and measures with a general character like those discussed in Section 4.3 is unlikely to improve the effectiveness. On the one hand, the state-of-the-art matchers from the matching contests [19, 20] incorporated other knowledge sources, but yielded an effectiveness close to or lower than that of BOT. On the other hand, the statement is substantiated by the knowledge acquisition bottleneck [271] which is a known problem in the context of measuring the semantic relatedness between words [225]. According to this problem, knowledge sources must be suited to the specific domain characteristics and the domain vocabulary in order to yield reliable results. Consequently, different model collections require different knowledge sources. However, the creation of such sources is expensive and time-consuming [272]. The knowledge acquisition problem has been recognized as a central challenge in the field of schema and ontology matching [171]. Accordingly, the use of domain specific knowledge sources was discussed. To this end, Aleksovski [273], Madhavan et al. [274], and Saha et al. [275] consider the use of corpora that comprise schemas and alignments. Additionally, improving schema and ontology matching by incorporating domain specific ontologies was amongst others examined in [276–279]. In a similar vein, Brockmans et al. [160] require experts to provide domain ontologies for business process model matching.

4.7 Summary

This chapter discussed the matching of process models by solely exploiting the labels of the activities. It started by introducing various options for the design of label-based matching techniques and by analyzing them with regard to the development datasets. First, it was shown that considering labels as strings and applying syntactic similarity measures to assess the similarity of the strings does not generally guarantee a high effectiveness. Next,

a more fine-grain assessment of the label similarity was examined. Here, labels were split into bag-of-words, the words were normalized through stemming, and a label similarity score was determined based on the comparison of the words. Although, the effectiveness was improved, it was still fairly low. Thus, the incorporation of word similarities that measure the sense relation of the words was studied. In contrast to the motivation that such approaches are necessary to assess the similarity of activities the considered similarities did not result in a significant increase in the effectiveness. Finally, techniques to address different levels of specificity were discussed, i.e., labels might be more abstract or definite than others. Here, pruning was introduced in order to cut large bag-of-words to the size of smaller bag-of-words. In this regard, relying on the maximum similarity to select words from the larger label yielded the best results. However, the impact was marginal.

Based on the results, the Bag-of-Words Technique (BOT) was introduced. It filters activities based on label equality, harmonizes labels, breaks these labels down into sets of harmonized words, reduces differences in the label specificity, and compares the words to determine a similarity score which can be used to classify activities as corresponding or not. BOT also comprises different features that implement these steps and that can be used to configure BOT.

While all the analysis results from the examination of the different design options already provided evidence towards Sub-hypothesis H2, the evidence was refined by analyzing BOT with regard to all datasets. First, a default configuration that permits the direct application of BOT was derived from the development results. Here, it was shown that the three considered configurations yield a varying and rather low effectiveness on the development datasets. This observation confirms that label-based matching techniques are characterized by a varing and generally poor effectiveness as postulated by Sub-hypothesis H2. Moreover, a comparison to the state of the art in terms of the results from the matching contests [19, 20] revealed that BOT's maximum effectiveness outperforms the state-of-the-art matchers and that the default configuration performs similarly to these matchers.

With regard to the evaluation datasets, the maximum effectiveness of BOT is higher, due to a higher labeling homogeneity. Further, the results confirmed that the effectiveness of different configurations varies across datasets. Additionally, the results also demonstrated that BOT is high performing with respect to the state of the art. Overall, the examination of BOT on these datasets provided evidence towards the general validity of Sub-hypothesis H2.

When applying the proposed default configuration, experts need to accept that it yields an effectiveness that is lower than the maximum effectiveness any configuration of BOT could achieve. To overcome this limitation, a semi-manual configuration approach was examined. That is, a subset of the gold standard alignments was used to optimize BOT's configuration. Then, the remaining model pairs were matched automatically by the optimized configuration. In this context, it was shown that a substantial amount of correspondences needs to be manually identified, in order to yield a configuration with an f-measure that is at least close to the default configuration. This result does not only justify the use of the default configuration, but also motivates the remaining sub-hypotheses which address the optimization of BOT's configuration.

Lastly, to conclude the discussion of the sub-hypothesis, challenges related to the identification of correspondences were analyzed. In this regard, it was revealed that false negatives reside on a poor assessment of the sense relations between the activities' labels. Moreover, the analysis unveiled that false positives are typically characterized by sets of shared words. Consequently, these findings demonstrated that knowledge sources are needed that reflect domain characteristics of model collections in order to improve the effectiveness of label-based techniques. However, in line with the literature it was argued that these knowledge sources are typically not available and expensive to create.

5

Analyzing Structure and Behavior

H3: The maximization of the effectiveness of label-based matching techniques is enabled by the analysis of control flow information.

In addition to the functional perspective which provides textual descriptions of activities, business process models also capture the behavioral perspective of business processes [68, 69]. That is, they define the control flow, i.e., structural and behavioral dependencies between activities, including sequential, parallel, and alternative execution patterns. Accordingly, many existing matching techniques consider the control flow. In this regard, the existence of certain control flow characteristics is usually assumed and a respective design that exploits these characteristics is proposed. Yet, evidence for the basic assumptions is typically not provided and the effects of the according design decisions are rarely studied (cf. Section 3.3). From this observation the question arises: *how can the consideration*

of control flow information improve the matching process? Through the verification of sub-hypothesis H3 this chapter aims to answer this question. In particular, the behavioral perspective is viewed from three different angles. First, it is examined whether control flow information is suited to enhance the pairwise classification of activities. The goal is to identify similarity scores based on control flow properties that help to better separate non-corresponding from corresponding activity pairs. Second, structural patterns for the identification of corresponding activity clusters are investigated. As defined in Section 3.1 such activity clusters constitute sets of activities within a process model that are part of complex correspondences. Respectively, the idea is to extend the classification of activity pairs by detecting activity clusters within process models and considering them as candidates for complex correspondences. Third, the order relation between correspondences is studied. This concept refers to the degree to which the order of activities in one process model resembles the order of their corresponding counterparts in another model. Here, the goal is to estimate the effectiveness of alignments by assessing their consistency, so that matching techniques can automatically optimize the proposed alignments by improving the consistency. Finally, the results are used to develop the Order Preserving Bag-of-Words Technique (OPBOT): a self-optimizing matching technique that searches the space of configurations of BOT in order to identify configurations that yield a high effectiveness and to combine the results of the best performing configurations.

The remainder of this chapter is organized as follows. Section 5.1 examines the pairwise classification, Section 5.2 structural patterns for activity clusters, and Section 5.3 the order relation. Next, OPBOT is introduced in Section 5.4. Then, it is evaluated and analyzed in Section 5.5. Finally, Section 5.6 discusses the findings in order to confirm the sub-hypothesis H3.

5.1 Multi-Dimensional Classification of Activity Pairs

The design of the label-based matching algorithms in Chapter 4 reposes on the idea to view business process model matching as a classification problem. That is, for a given pair of process models these algorithms iterate over the set of all activity pairs. For each of the

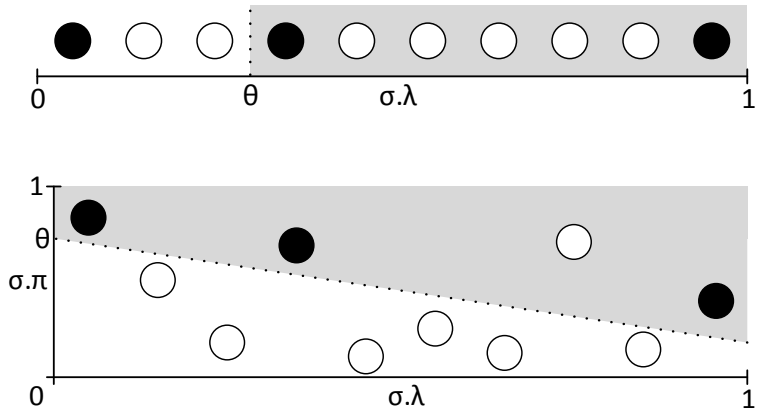

FIGURE 5.1: One-dimensional (top) vs. two-dimensional (bottom) classification of activity pairs

pairs they compute a single similarity score and classify the pair as corresponding, if the score is high enough, and otherwise as non-corresponding. As shown in Chapter 4 and on the top of Figure 5.1 the effectiveness of these one-dimensional label-based classifiers is limited. In the figure, the classifier yields two true positives, one false negative, and five false positives. Accordingly, the recall is .667, the precision .286, and the f-measure .4.

To improve the effectiveness of those one-dimensional, label-based classifiers, this section pursues the idea to consider multiple similarity dimensions. In particular, the goal is to add similarity scores based on the behavioral perspective to the matching techniques. The right side of Figure 5.1 illustrates the effect that this extension strives to achieve. Here, adding another similarity score leads to a distribution of activity pairs within a space of similarity scores. That is, the activity pairs are now distributed in a two-dimensional space spanned by a label similarity score $\sigma.\lambda$ and a similarity score based on the behavioral perspective $\sigma.\pi$, instead of being ordered according to a label similarity score $\sigma.\lambda$. The figure shows the ideal case where adding $\sigma.\pi$ allows for a definition of a threshold function that yields a better separation of corresponding and non-corresponding activity pairs. As the classifier proposes all truly existing correspondences and only one false positive, the effectiveness is improved: the recall is 1, the precision .75 and the f-measure .857.

In order to examine the extension of label-based matching algorithms, a series of similarity scores is introduced. These scores are based on a diverse range of activity properties derived from the behavioral perspective. Each property is represented by a particular *property function* that returns a numeric value for a given activity. There

are two versions for each property function. The first version returns a natural number for the activity, e.g., the number of activities on the path to the start node. It is also referred to as the *absolute property function* Π. However, process models are usually of a varying complexity and contain a different number of activities or control flow constraints, e.g., the number of activities in the process models of the BR dataset varies from 9 to 25 activities (cf. Table 3.3). Hence, relying on absolute values may lead to a distorted similarity assessment as illustrated by the following example: consider two activities a, a' from two process models P, P' and their distances to the start node. While a is preceded by one activity in P, a' is preceded by three activities in P'. Thus, the absolute property values for the activities are $\Pi(a) = 1$ and $\Pi(a) = 3$. With regard to these values, both activities are different. However, the assessment differs when the context of both activities is taken into account, i.e., the respective process models. In the example, both models are sequences where a is succeeded by one activity and a' by three activities. In this case both activities have the same relative distance to the start node as they are located in the middle of the respective process model. Consequently, they should be considered as equal with regard to the start node distance. Based on these considerations the second version of the property function is the *relative property function* π which returns a value on the interval $[0..1]$. It is based on the normalization of the absolute value achieved by dividing the value with the maximum value found for any activity in the same model.

Definition 5.1 (Property) *Let $P = (N, E, \lambda, \tau, A)$ be a process model and $a \in A$ be an activity. Then, an absolute property function Π is a function*

$$\Pi : A \rightarrow \mathbb{N}$$

that returns a natural number for the activity with regard to a certain property. The respective relative property function π is then defined as

$$\pi(a) = \begin{cases} 0 & \text{if } \max\limits_{a^* \in A} \Pi(a^*) = 0 \\ \frac{\Pi(a)}{\max\limits_{a^* \in A} \Pi(a^*)} & \text{else} \end{cases}$$

To consider property functions for multi-dimensional activity pair classification, *property similarity functions* $\sigma.\pi$ are defined to measure the similarity of activities with regard

to a certain property. For two activities the absolute difference between their relative property values is determined and then subtracted from one. Due to relying on the relative property functions the respective similarity functions are also bound to the interval $[0,1]$.

Definition 5.2 (Property similarity) *Let $P = (N, E, \lambda, \tau, A)$, $P' = (N', E', \lambda', \tau', A')$ be two process models and $a \in A$, $a' \in A'$ be two activities. Given a relative property function π, the property similarity function $\sigma.\pi$ is defined as:*

$$\sigma.\pi(a, a') = 1 - |\pi(a) - \pi(a')|$$

In the following, specific properties in terms of absolute property functions are introduced. These functions are grouped into three property categories: *path*, *fragment*, and *execution semantics* properties. Moreover, for each property a subscript x is defined which is used to refer to the absolute property, Π_x, the relative property π_x and the property similarity $\sigma.\pi_x$ functions. Beside the formal definitions, examples based on the university admission process models from Section 3.1 are provided and matching techniques from related work are pointed out that incorporate similar properties. After the properties were introduced, they are assessed with regard to their suitability for activity pair classification.

5.1.1 Path Properties

The first group of properties considers the process models as directed graphs consisting of nodes that represent the process elements including activities, events, gateways etc., and edges which depict the dependencies between these elements. To this end, the execution semantics that the models capture are ignored, e.g., and-, xor-, and or-gateways are simply considered as model elements and the differences in their meanings are neglected. In particular, the focus is on *paths* which depict connections between nodes in a graph.

Following the common understanding from graph theory [280], a path in a directed graph (N, E) is a sub-graph which contains a sequence of nodes $\{n_i\}_{i=1}^{k \in \mathbb{N}}$ with $n_i \in N$ where each node that is part of the path only occurs once in the path, i.e., $\forall 1 \leq i, j \leq k : (n_i \neq n_j \Leftrightarrow i \neq j) \wedge (n_i = n_j \Leftrightarrow i = j)$. Moreover, for each node in the sequence there must be a directed edge in the graph that connects the node to its successor, i.e., $\forall 1 \leq i \leq k-1 : (n_i, n_{i+1}) \in E$.

Definition 5.3 (Path) *Let $P = (N, E, \lambda, \tau, A)$ be a process model. Then, a path is defined as a subgraph $P_{\rightarrow} = (N_{\rightarrow}, E_{\rightarrow})$ such that*

- $N_{\rightarrow} = \{n_i\}_{i=1}^{k \in \mathbb{N}}$ *with* $n_i \in N$ *and* $\forall 1 \leq i, j \leq k : (n_i \neq n_j \Leftrightarrow i \neq j) \wedge (n_i = n_j \Leftrightarrow i = j)$ *is a sequence of distinct nodes; and*
- $E_{\rightarrow} = \{(n_i, n_{i+1})\}_{i=1}^{k-1}$ *with* $(n_i, n_{i+1}) \in E$ *is the sequence of edges connecting the nodes.*

Furthermore, $n_1 \rightarrow n_k$ *explicitly denotes that the path* $P_{\rightarrow}$ *leads from* n_1 *to* n_k*. Finally, the set of all distinct paths leading from* n_1 *to* n_k *is referred to as* $n_1 \overset{*}{\rightarrow} n_k$*.*

Examples of paths can be found in Figure 5.2. This figure shows the university admission process models from Section 3.1 where the labels were omitted and replaced by the ids of the nodes. As the labels are irrelevant for the definition of the control flow properties, this was done to keep the example concise. First, consider the nodes β_1 and β_3 in process B. These nodes are connected by a path that consists of the activities β_1, β_2, and β_3 as well as of the edges e_2 and e_3. On the contrary, there is no path in process B that leads from β_3 to β_1. Second, the nodes s_α and α_3 from process A are connected by two paths. There is the path that contains s_α, α_1, and α_3 as well as e_1, e_2, and e_4. Moreover, the nodes are also connected by the path that comprises s_α, α_2, and α_3 as well as e_1, e_3, and e_5. This example shows that paths do not necessarily reflect the observable behavior as α_1 and α_2 are part of a parallel block and are always carried out before α_3. However, they are part of different paths connecting s_α to α_3.

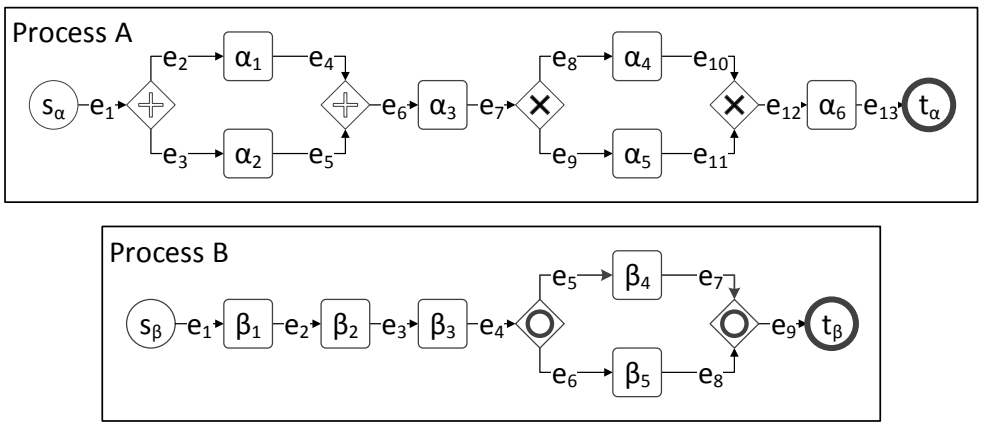

FIGURE 5.2: Graph structure of the university admission models

There are different algorithms to determine paths within graphs. One of the most popular algorithms is Dijkstra's algorithm [281] for finding the shortest paths from a start node to all nodes in a graph. The A* (A star) algorithm [187, 188] is an extension of Dijkstra's algorithm that generally achieves a better performance. Additionally, the set of all paths between two nodes can typically be determined by a depth first search [282].

Based on this understanding the first subset of properties in this category are the *path position properties*. In alignment with the related work the assumption is that corresponding activities are located at similar positions in a process model. For example, the matching techniques by Nejati et al. [161] and Baumann et al. [201] consider the position of activities. Furthermore, the ICoP framework [162] also comprises several components which exploit the position of activities. Here, two properties are defined. The absolute *start distance* $\Pi_{\rightarrow a}$ of an activity was used to explain the difference between the relative and absolute property function at the beginning of this section. It is the minimum number of activities that can be found on any path leading from any of the start nodes of the process to the activity. Similarly, the absolute *end distance* $\Pi_{a\rightarrow}$ of an activity is the minimum number of activities that can be found on any path leading from the activity to any of the end nodes of the process model.

Definition 5.4 (Path position properties) *Let $P = (N, E, \lambda, \tau, A)$ be a process model. Further, let $N_s = \{n_s | n_s \in N \wedge \forall n \in N : (n, n_s) \notin E\}$ be the set of start nodes and $N_e = \{n_e | n_e \in N \wedge \forall n \in N : (n_e, n) \notin E\}$ be the set of end nodes. Given an activity $a \in A$, the absolute start distance $\Pi_{\rightarrow a}$ and end distance $\Pi_{a\rightarrow}$ properties are defined as:*

$$\Pi_{\rightarrow a}(a) := \min_{n_s \in N_s} \min_{n_s \rightarrow a \in n_s \overset{*}{\rightarrow} a} |N_{\rightarrow} \cap A| - 1$$

$$\Pi_{a\rightarrow}(a) := \min_{n_e \in N_e} \min_{a \rightarrow n_e \in a \overset{*}{\rightarrow} n_e} |N_{\rightarrow} \cap A| - 1$$

To illustrate the graph position properties Table 5.1 summarizes the respective property and similarity values for the activity pairs (α_3, β_2) and (α_6, β_1) from the example in Figure 5.2. The activities α_3 and β_2 are located at similar positions in their process models. That is, both of them have an absolute start distance of 1 and an absolute end distance of 2. As the maximum absolute start and end distance is 3 in both process models the

TABLE 5.1: Path position properties for the university admission example

	$\Pi_{\rightarrow a}$	$\pi_{\rightarrow a}$	$\sigma.\pi_{\rightarrow a}$	$\Pi_{a\rightarrow}$	$\pi_{a\rightarrow}$	$\sigma.\pi_{a\rightarrow}$
α_3	1	$.\overline{3}$	1	2	$.\overline{6}$	1
β_2	1	$.\overline{3}$		2	$.\overline{6}$	
α_6	3	1	0	0	0	0
β_1	0	0		3	1	

relative distances are also equal and the property similarity scores are 1. In contrast, the activities α_6 and β_1 are located at opposite ends of their process models. While β_1 is the activity closest to the start node, α_6 is the activity closest to the end node. Consequently, the respective similarity scores are 0.

The second category of properties is based on the assumption that corresponding activities are embedded in similar neighborhoods. To this end, different variants of the *path neighborhood* are introduced. For a given activity a the absolute *upstream neighborhood* property $\Pi_{\bullet a}$ returns the number of activities for which there is at least one path leading to a that does not contain any other activity. In contrast, the absolute *downstream neighborhood* property $\Pi_{a\bullet}$ for a given activity a is the number of activities for which there exists at least one path that leads from a to them and that does not contain any other activity. Finally, the absolute *neighborhood* property $\Pi_{\bullet a\bullet}$ combines the upstream and the downstream neighborhood. Similar to these properties, the matching technique from [161] as well as the Triple-S approach from the first matching contest [19] incorporate notions of graph neighborhoods.

Definition 5.5 (Path neighborhood properties) *Let $P = (N, E, \lambda, \tau, A)$ be a process model and $a \in A$ be an activity. Then, the upstream neighborhood $\Pi_{\bullet a}$, the downstream neighborhood $\Pi_{a\bullet}$, and the neighborhood $\Pi_{\bullet a\bullet}$ properties are defined as:*

$$\Pi_{\bullet a}(a) := |\{a' | a' \in A \wedge \exists a' \rightarrow a : N_{\rightarrow} \cap A = \emptyset\}|$$

$$\Pi_{a\bullet}(a) := |\{a' | a' \in A \wedge \exists a \rightarrow a' : N_{\rightarrow} \cap A = \emptyset\}|$$

$$\Pi_{\bullet a\bullet}(a) := |\{a' | a' \in A \wedge (\exists a \rightarrow a' : N_{\rightarrow} \cap A = \emptyset \vee \exists a' \rightarrow a : N_{\rightarrow} \cap A = \emptyset)\}|$$

TABLE 5.2: Path neighborhood properties for the university admission example

	$\Pi_{\bullet a}$	$\pi_{\bullet a}$	$\sigma.\pi_{\bullet a}$	$\Pi_{a\bullet}$	$\pi_{a\bullet}$	$\sigma.\pi_{a\bullet}$	$\Pi_{\bullet a\bullet}$	$\pi_{\bullet a\bullet}$	$\sigma.\pi_{\bullet a\bullet}$
α_3	2	1	1	2	1	.5	4	1	$.\overline{3}$
β_2	1	1		1	.5		1	$.\overline{3}$	
α_6	2	1	0	0	0	.5	2	1	$.\overline{6}$
β_1	0	0		1	.5		2	$.\overline{6}$	

Table 5.2 shows the corresponding values for the activity pairs (α_3, β_2) and (α_6, β_1) from the running example. Here, α_3 and β_2 are equal with regard to the upstream neighborhood, but differ with regard to the other two properties. Similarly, α_6 and β_1 are totally dissimilar with regard to the upstream neighborhood, but share similarities with regard to the other two properties.

5.1.2 Fragment Properties

Similar to the path category the fragment properties neglect the dynamic aspects of the behavioral perspective. In contrast, the fragment properties do not directly rely on the graphs. Instead, they are defined with regard to a nested hierarchy of *fragments* derived from process models. In this regard, fragments are connected sub-graphs that have a *single-entry* and a *single-exit* node [283]. That is, all paths leading from a node in the fragment to a node outside the fragment contain the exist node. Likewise, all paths connecting a node outside the fragment to a node in the fragment comprise the entry node. The entry and the exit node of a fragment are also referred to as the fragment's *boundary nodes*. Moreover, fragments might be decomposed into further fragments and the whole process model is typically perceived as the root fragment.

Many matching techniques incorporate the idea to decompose process models into hierarchies of fragments. In this regard, the basic assumption is that fragments are sub-processes and their activities refer to the same purpose. Thus, it is believed that activities within a fragment are likely to correspond to activities in a different process model that are also part of the same fragment. For instance, the ICoP framework [162] contains

various components that rely on a fragment hierarchy, e.g., the tree depth ratio booster is used to filter correspondences by comparing the depth of the two activities in the respective fragment hierarchies. Here, correspondences are favored where the activities have a similar depth in the hierarchy over those with dissimilar depth values. Weidlich et al. [198] represent process models as text documents where each passage represents an activity and the context of passages, i.e., the preceding and succeeding passages, is considered in the similarity computation. Hereby, the sequential ordering of the passages is derived from a fragment hierarchy. Other matching techniques that rely on fragment hierarchies include [16, 193, 197].

The first step to define fragment properties is to determine how fragments are derived from a process model. For this task there are various approaches available. Tarjan and Valdes [284] introduced an approach to decompose sequential programs into sub-program hierarchies based on their work on graph connectivity [285]. Similarly, program structure trees are presented in [286, 287]. Moreover, Ouyang et al. [288, 289] developed a parsing technique to translate BPMN models into block structures.

In this thesis, a process model is decomposed into a Refined Process Structure Tree (RPST) [283, 290] which is also used by many of the aforementioned matching techniques [16, 162, 198]. In contrast to the other approaches that detect fragments RPSTs have the advantage that they are unique and more fine-grain [283]. In other words, the computation of an RPST for a process model is deterministic, i.e., there is only one RPST for each model. Moreover, the resulting hierarchy contains more fragments than hierarchies determined by other approaches. Finally, the fragments in an RPST are maximal. That means that no other node from the model can be added to the fragment so that it is a connected sub-graph and still has one exit and one entry node. A simplified algorithm to compute RPSTs based on other decomposition techniques [291, 292] is presented in [293].

To outline the decomposition of process models based on the RPST, Figure 5.3 presents the RPSTs for the process models from the example. The edges of a process model are considered as the most fine-grain fragments and are referred to as *trivial* fragments. As a consequence each edge in a process model corresponds to a leaf node in the according

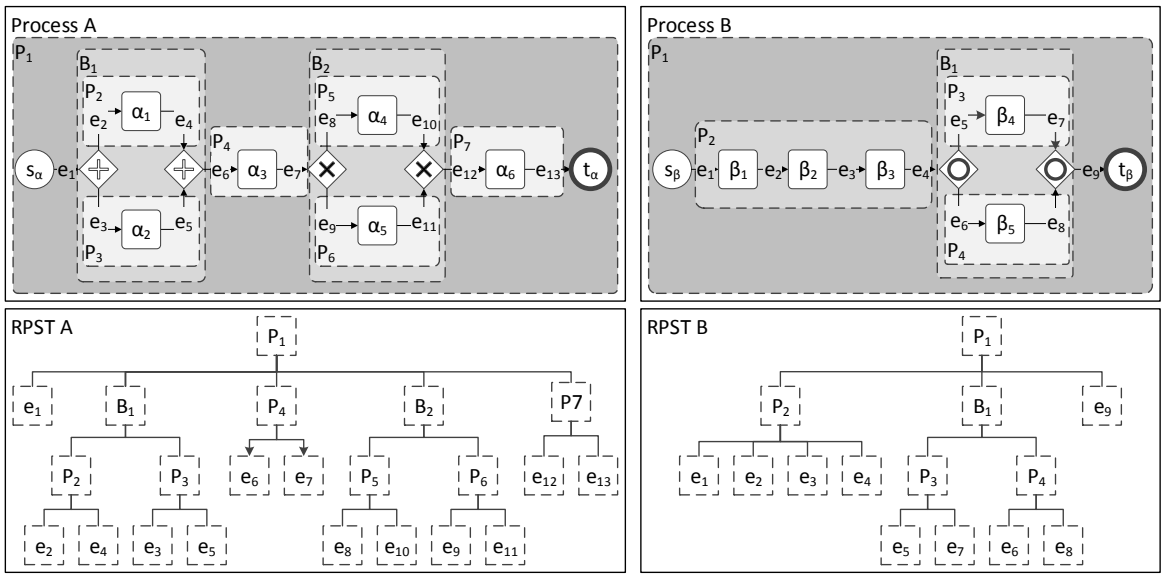

FIGURE 5.3: RPSTs for the university admission models

RPST. In the hierarchy more complex fragments are composed of sub-fragments. There are three different types of such complex fragments. A *polygon* (marked with P in Figure 5.3) is a path in the model that connects the entry to the exit node and all nodes in the fragment are part of this path. In the example, both process models constitute polygons. There are also further polygons, e.g., process B contains a polygon that comprises the entry node, the activities β_1, β_2, β_3, and the or-split. A *bond* (marked with B in Figure 5.3) is composed of multiple other sub-fragments. Here, it is required that in each sub-fragment the entry node corresponds to either the entry or the exit node of the bond, and that in each sub-fragment the exit node also corresponds to either the entry or the exit node of the bond. An example of a bond is the parallel block in process A. Here, all sub-fragments connect the parallel split to the parallel join. Complex fragments that are neither a polygon nor a bond are referred to as *rigids*. The example models contain no rigids.

A prerequisite for the computation of the RPST is that process models contain exactly one start and exactly one end node. However, in practice process models might contain more than one start and one end node. Consequently, this requirement might appear to limit the application of the RPST and thus of the fragment properties. But, models with multiple start and/or end nodes can be transformed into models with a single start and a single end node without changing the original structure of the model [294]. In case there are multiple start nodes, a new start node is introduced and for all initial start nodes an

edge is introduced that connects the new start node to the initial start node. Analogously, multiple end nodes are handled by defining a new end node and an edge for each of the original end nodes that leads from the original end node to the new end node.

Another requirement is introduced to simplify the definition of the fragment properties. It is expected that for each activity there is at most one incoming and at most one outgoing edge, i.e., $\forall a \in A, n \in N : (|\{n|(n,a) \in E\}| \leq |\{n|(a,n)| \in E\} \leq 1)$. Again this requirement can be ensured without impacting the fragment structure [294]. Therefore, for each activity with multiple incoming edges a new node is introduced and all incoming edges are replaced by edges that end in the new node rather than in the activity. The new node is then linked to the activity. Activities with multiple outgoing edges are handled analogously and so are activities with multiple incoming and outgoing edges. The result of this transformation is that there are at most two trivial fragments for each activity, one representing the incoming and the other the outgoing edge. In case there are two trivial fragments for an activity, the paths connecting the root to the ancestors of the trivial fragments in the RPST are identical for both trivial fragments. Accordingly, there is exactly one path of complex fragments for each activity. In the remainder of the thesis, the introduced transformations are implicitly applied to a process model when its RPST is computed.

Based on these considerations the RPST is formally defined as a set of complex fragments where each fragment contains a set of activities and is located at a certain depth. The depth of a fragment is the number of fragments on the path from the root to the fragment inclusive of the fragment itself and exclusive of the root. By definition the depth of the root is 0.

Definition 5.6 (Refined process structure tree) *Let $P = (N, E, \lambda, \tau, A)$ be a process model. The refined process structure tree R of this process is a 3-tuple*

$$(F, depth, act)$$

such that

- *F is the set of complex fragments;*

- $depth : F \rightarrow \mathbb{N}$ *is a function that returns the depth of a fragment; and*
- $act : F \rightarrow \mathscr{P}(A)$ *is a function that returns the set of activities in a fragment.*

Given the definition of the RPST, two fragment property functions are defined. The first function is the absolute *RPST depth* $\Pi_{\downarrow R}$. Similar to the graph position properties, it refers to the position of an activity in the RPST. The depth for an activity is equal to the depth of the lowest complex fragment in the RPST that contains the activity. The other function is the absolute *RPST neighborhood* $\Pi_{\bullet R \bullet}$ which determines the neighborhood of an activity in the RPST. Here, the neighborhood is defined as the activities in the lowest complex fragment that contains the activity exclusive of the activity. The RPST neighborhood function returns the number of activities in this set.

Definition 5.7 (Fragment properties) *Let* $P = (N, E, \lambda, \tau, A)$ *be a process model and* $R = (F, depth, act)$ *be the respective RPST. Further, let* $a \in A$ *be an activity and* $frag_a \in F$ *be the fragment with the largest depths that contains* a*, i.e.,* $\neg\exists frag \in F : frag \neq frag_a \wedge a \in act(frag) \wedge depth(frag) \geq depth(frag_a)$*. Then, the absolute RPST depth* $\Pi_{\downarrow R}$ *and the absolute RPST neighborhood* $\Pi_{\bullet R \bullet}$ *properties are defined as:*

$$\Pi_{\downarrow R}(a) = depth(frag_a)$$

$$\Pi_{\bullet R \bullet}(a) = |act(frag_a)| - 1$$

Table 5.3 shows the respective values for the two activity pairs from the university admission example. With regard to the RPST depth both activity pairs are characterized by equality. That is because the absolute depth of all four activities is 1 and the maximum depth for both processes is 2. On the contrary, the activity pairs are totally dissimilar with regard to the RPST neighborhood. The main reason is that each of the activities in the first model has no neighbors in the RPST. Thus, all activities take an absolute and a relative value of 0 for this property, while β_1 and β_2 take the largest absolute values in process B.

5.1.3 Execution Semantics Properties

In contrast to the other property groups the last category comprises properties that rely on the execution semantics of processes. They define in which order activities can be

TABLE 5.3: Fragment properties for the university admission example

	$\Pi_{\downarrow R}$	$\pi_{\downarrow R}$	$\sigma.\pi_{\downarrow R}$	$\Pi_{\bullet R \bullet}$	$\pi_{\bullet R \bullet}$	$\sigma.\pi_{\bullet R \bullet}$
α_3	1	.5	1	0	0	0
β_2	1	.5		2	1	
α_6	1	.5	1	0	0	0
β_1	1	.5		2	1	

executed and modeling languages typically provide a set of elements to capture them. The most common elements in this regard provide means to model the parallel, exclusive, and inclusive execution of activities. The parallel execution indicates that activities can be carried out simultaneously. The exclusive execution is used when there are different alternative execution paths and only one of them can be executed. Similarly, the inclusive execution refers to situations where either one, all, or a subset of the alternatives needs to be chosen. Respective elements in process model languages comprise the parallel, exclusive, and inclusive gateways in BPMN as well as the and-, xor-, and or-connectors in EPC. An overview of constructs and patterns to capture the execution semantics of processes is given in [295, 296].

The examination of the execution semantics typically relies on *traces* where a trace captures a possible order in which the activities of a process can be executed. That is, a trace is a sequence of activities $\theta = \{a_i\}_{i \in \mathbb{N}}$ and the order of the activities in the sequence is determined by the order in which the execution of the activities is started [5, 85pp.]. Thus, for any two activities a_k, a_l that occur in the trace with $k < l$ holds that the execution of a_k was started before the execution of a_l. To illustrate the concept of a trace Table 5.4 presents the set of all possible traces for the processes from the running example.

For Process A there are four different traces that can be derived from the process model. As α_1 and α_2 are part of a parallel block, they are part of every trace, but their order might differ depending on which activity is started first. Consequently, each trace starts with either $\alpha_1 \mapsto \alpha_2$ or $\alpha_2 \mapsto \alpha_1$. Once these activities were executed, α_3 is carried out. Thus, each trace starts with $\alpha_1 \mapsto \alpha_2 \mapsto \alpha_3$ or $\alpha_2 \mapsto \alpha_1 \mapsto \alpha_3$. Next, α_4 and α_5 are executed alternatively. Consequently, there are four sub-traces containing one of

TABLE 5.4: Possible execution traces of the university admission process models

Process A	*Process B*
$\alpha_1 \mapsto \alpha_2 \mapsto \alpha_3 \mapsto \alpha_4 \mapsto \alpha_6$	$\beta_1 \mapsto \beta_2 \mapsto \beta_3 \mapsto \beta_4$
$\alpha_2 \mapsto \alpha_1 \mapsto \alpha_3 \mapsto \alpha_4 \mapsto \alpha_6$	$\beta_1 \mapsto \beta_2 \mapsto \beta_3 \mapsto \beta_5$
$\alpha_1 \mapsto \alpha_2 \mapsto \alpha_3 \mapsto \alpha_5 \mapsto \alpha_6$	$\beta_1 \mapsto \beta_2 \mapsto \beta_3 \mapsto \beta_4 \mapsto \beta_5$
$\alpha_2 \mapsto \alpha_1 \mapsto \alpha_3 \mapsto \alpha_5 \mapsto \alpha_6$	$\beta_1 \mapsto \beta_2 \mapsto \beta_3 \mapsto \beta_5 \mapsto \beta_4$

the two start traces followed by either α_4 or α_5. Finally, each trace ends with α_6 as it is always carried out after α_4 or α_5. Similarly, there are also four traces for Process B. Here, each trace starts with $\beta_1 \mapsto \beta_2 \mapsto \beta_3$. This sequence of activities is completed by a combination of β_4 and β_5. As these two activities are part of an inclusive block, there are four sub-traces for this block. On the one hand, β_4 and β_5 can be executed in parallel leading to the sub-traces $\beta_4 \mapsto \beta_5$ and $\beta_5 \mapsto \beta_4$. On the other hand, they can be executed alternatively resulting in two sub-traces that contain just one of them. Note that the exclusive execution of β_4 might violate the intention of the modeler. The reason is that β_4 represents the activity 'register applicant' and β_5 stands for 'publish notification'. In this context, β_5 should always be executed as all applicants need to be notified about the university's decision. In contrast, β_4 will only be carried out in cases where the applicant is accepted. To capture the exclusive execution that β_5 always needs to be executed, the model needs to be annotated with additional rules. However, such annotations are not considered in this thesis.

A strategy to incorporate traces is to use logs which contain traces that were observed during the execution of processes. Logs were suggested for the determination of the similarity of process models [297, 298]. However, relying on observed behavior implies that experts need to provide logs or that logs are available. As this limits the applicability of the approaches, logs are not considered here. It is also possible to derive traces from models through model simulation. Yet, the set of possible traces can be very large and it might even be impossible to determine all traces [299, 300]. Hence, the execution semantics properties in this thesis rely on an abstract representation of the execution semantics in terms of the *behavioral profile* [163, 301, 302]. The behavioral profile of

a process model can be computed without determining the set of all possible traces [303, 304]. In this thesis, the implementation provided by the jBPT library[29] is utilized. The matching technique by Leopold et al. [194] considers constraints that alignments must satisfy and that are derived from behavioral profiles. Other related uses of the behavioral profile include consistency checking [163] and similarity search [305].

A prerequisite for the computation of the behavioral profiles is that they are sound [162]. This is a general assumption for approaches that determine the execution semantics of a process model. The reason is that unsound process models can contain deadlocks, livelocks, dead tasks, or might not properly terminate [92] and thus their execution semantics cannot be reliably assessed. Consequently, the execution semantics properties only yield reliable results, if the matched process models fulfill the soundness criterion.

In essence, a behavioral profile captures relations between activities from the same process models and provides information whether activities occur in sequence, in parallel, or alternatively. These relations are defined with regard to the *weak order relation* $\succ_P$. Two activities a_1, a_2 are in a weak order relation $a_1 \succ_P a_2$, if there exists a trace $\theta = \{a_i\}_{i=1}^{m \in \mathbb{N}}$ in which a_1 occurs before a_2.

Definition 5.8 (Weak order relation) *Let $P = (N, E, \lambda, \tau, A)$ be a process model and Θ_P the set of all traces. The weak order relation $\succ_P \subseteq A \times A$ contains all activity pairs (a_x, a_y) for which there is a trace $\theta \in \Theta_P$ with $\theta = \{a_i\}_{i=1}^{m \in \mathbb{N}}$ such that $a_j = a_x$, $a_k = a_y$ and $i < k \leq m$.*

Based on the weak order relation between activities from a process model, the behavioral profile comprises four relations that define in which order the activities occur. The *strict order* $\rightsquigarrow_P$ and the *inverse strict order* relation $\rightsquigarrow_P^{-1}$ hold between activities that occur in sequence. That is, if there is at least one trace in which a_1 occurs before a_2, but no trace in which a_2 occurs before a_1, a_1 is in strict order with a_2, i.e., $a_1 \rightsquigarrow_P a_2$. Moreover, in this case a_2 is in inverse strict order with a_1, i.e., $a_2 \rightsquigarrow_P^{-1} a_1$. If there is at least one trace in which a_1 appears before a_2 and there is also at least one trace in which a_2 appears before a_1, then these two activities are in *interleaving order* $a_1 \parallel_P a_2$ indicating that they can be executed in parallel. Finally, if there is no trace that contains both activities, they are in

[29] https://www.openhub.net/p/jbpt, accessed: 13/01/2017

exclusive order $a_1 +_P a_2$, meaning that they are carried out alternatively. By definition each activity is in exclusive order to itself, i.e., $\forall a \in A : a +_P a$. Note that Weidlich et al. [306] introduced the causal behavioral profiles which distinguishes two types of strict order relations. The first relation comprises all activities in a strict order relation for which it holds that whenever the first activity is part of a trace, the second is too. The second relation contains all remaining activities. However, the application of the execution semantics properties is restricted to sound models. For this reason the focus is on a basic evaluation of the control flow properties and the extension of the analysis to cover more fine-grained properties is subject to future work.

Definition 5.9 (Behavioral profile) *Let* $P = (N, E, \lambda, \tau, A)$ *be a process model and* $\succ_P$ *the respective weak order relation. Then, the behavioral profile* $\mathscr{B}_P$ *is a 4-tuple*

$$(\rightsquigarrow_P, \rightsquigarrow_P^{-1}, +_P, \|_P)$$

such that

- $\rightsquigarrow_P \subseteq A \times A$ *with* $\forall (a_x, a_y) \in \rightsquigarrow_P : (a_x, a_y) \in \succ_P \wedge (a_y, a_x) \notin \succ_P$ *is the strict order relation;*
- $\rightsquigarrow_P^{-1} \subseteq A \times A$ *with* $\forall (a_x, a_y) \in \rightsquigarrow_P : (a_x, a_y) \notin \succ_P \wedge (a_y, a_x) \in \succ_P$ *is the inverse strict order relation;*
- $+_P \subseteq A \times A$ *with* $\forall (a_x, a_y) \notin \rightsquigarrow_P : (a_x, a_y) \in \succ_P \wedge (a_y, a_x) \notin \succ_P$ *is the exclusive order relation; and*
- $\|_P \subseteq A \times A$ *with* $\forall (a_x, a_y) \in \rightsquigarrow_P : (a_x, a_y) \in \succ_P \wedge (a_y, a_x) \in \succ_P$ *is the interleaving order relation.*

Table 5.5 outlines the behavioral profiles for the process models from the example. In accordance with the set of possible traces from Table 5.4 the behavioral profile of process A shows that the activities α_4 and α_5 constitute alternatives. Moreover, the activities α_1 and α_2 are carried out in parallel. Similarly, β_4 and β_5 are also considered as simultaneously executed activities, because they are part of an inclusive block and thus they can be

TABLE 5.5: Behavioral profiles for the university admission process models

Process A							*Process B*					
	α_1	α_2	α_3	α_4	α_5	α_6		β_1	β_2	β_3	β_4	β_5
α_1	$+$	$\parallel$	$\rightsquigarrow$	$\rightsquigarrow$	$\rightsquigarrow$	$\rightsquigarrow$	β_1	$+$	$\rightsquigarrow$	$\rightsquigarrow$	$\rightsquigarrow$	$\rightsquigarrow$
α_2	$\parallel$	$+$	$\rightsquigarrow$	$\rightsquigarrow$	$\rightsquigarrow$	$\rightsquigarrow$	β_2	$\rightsquigarrow^{-1}$	$+$	$\rightsquigarrow$	$\rightsquigarrow$	$\rightsquigarrow$
α_3	$\rightsquigarrow^{-1}$	$\rightsquigarrow^{-1}$	$+$	$\rightsquigarrow$	$\rightsquigarrow$	$\rightsquigarrow$	β_3	$\rightsquigarrow^{-1}$	$\rightsquigarrow^{-1}$	$+$	$\rightsquigarrow$	$\rightsquigarrow$
α_4	$\rightsquigarrow^{-1}$	$\rightsquigarrow^{-1}$	$\rightsquigarrow^{-1}$	$+$	$+$	$\rightsquigarrow$	β_4	$\rightsquigarrow^{-1}$	$\rightsquigarrow^{-1}$	$\rightsquigarrow^{-1}$	$+$	$\parallel$
α_5	$\rightsquigarrow^{-1}$	$\rightsquigarrow^{-1}$	$\rightsquigarrow^{-1}$	$+$	$+$	$\rightsquigarrow$	β_5	$\rightsquigarrow^{-1}$	$\rightsquigarrow^{-1}$	$\rightsquigarrow^{-1}$	$\parallel$	$+$
α_6	$\rightsquigarrow^{-1}$	$\rightsquigarrow^{-1}$	$\rightsquigarrow^{-1}$	$\rightsquigarrow^{-1}$	$\rightsquigarrow^{-1}$	$+$						

executed in parallel. All remaining activities are in a strict order or inverse strict order relation, respectively.

For each of the four relations in the behavioral profile there is one *behavior property*. The absolute *strict order property* $\Pi_{\rightsquigarrow}$ and the absolute *inverse strict order property* $\Pi_{\rightsquigarrow^{-1}}$ are counterparts of the graph position properties. For an activity a they return the number of activities that are executed before or after a, but not in parallel. Similarly, the absolute *exclusive order property* Π_{+} and the absolute *interleaving order property* $\Pi_{\parallel}$ represent counterparts of the graph neighborhood properties. Thus, they return the number of parallel or alternatively executed activities.

Definition 5.10 (Behavior properties) *Let* $P = (N, E, \lambda, \tau, A)$ *be a process model and* $\mathscr{B}_P = (\rightsquigarrow_P, \rightsquigarrow_P^{-1}, +_P, \parallel_P)$ *its behavioral profile. For a given activity* a*, the absolute strict order* $\Pi_{\rightsquigarrow}$*, inverse strict order* $\Pi_{\rightsquigarrow^{-1}}$*, exclusive order* Π_{+} *and interleaving order* $\Pi_{\parallel}$ *properties are defined as:*

$$\Pi_{\rightsquigarrow}(a) = |\{a_x | (a, a_x) \in \rightsquigarrow_P\}|$$

$$\Pi_{\rightsquigarrow^{-1}}(a) = |\{a_x | (a, a_x) \in \rightsquigarrow_P^{-1}\}|$$

$$\Pi_{+}(a) = |\{a_x | (a, a_x) \in +_P \wedge a \neq a_x\}|$$

$$\Pi_{\parallel}(a) = |\{a_x | (a, a_x) \in \parallel_P\}|$$

TABLE 5.6: Execution semantics properties for the university admission example

	$\Pi_{\rightsquigarrow}$	$\pi_{\rightsquigarrow}$	$\sigma.\pi_{\rightsquigarrow}$	$\Pi_{\rightsquigarrow^{-1}}$	$\pi_{\rightsquigarrow^{-1}}$	$\sigma.\pi_{\rightsquigarrow^{-1}}$	Π_{+}	π_{+}	$\sigma.\pi_{+}$	$\Pi_{\parallel}$	$\pi_{\parallel}$	$\sigma.\pi_{\parallel}$
α_3	3	.75	1	2	.4	$.9\overline{3}$	0	0	1	0	0	1
β_2	3	.75		1	$.\overline{3}$		0	0		0	0	
α_6	0	0	0	5	1	0	0	0	1	0	0	1
β_1	4	1		0	0		0	0		0	0	

Lastly, Table 5.6 presents the property values for the activities from the example. Like the graph position properties, the strict and inverse strict order property similarities show that α_3 and β_2 are located at similar positions, whereas α_6 and β_1 are totally dissimilar. As none of the four activities is carried out in parallel with or alternatively to any other activity the exclusive and the interleaving similarities indicate that they are equal with regard to these properties.

5.1.4 Suitability Analysis

In order to extend the one-dimensional, label-based classification of activity pairs, the final step is to evaluate which of the property similarity functions are suited to enhance the classification. This is the case, if a similarity function reliably separates non-corresponding from corresponding activity pairs. That is, it needs to assign corresponding and non-corresponding activity pairs to different ranges on the interval $[0, 1]$. For this reason, the correlation between the similarity values yielded by the functions and the classification of activity pairs as corresponding or non-corresponding is empirically examined next.

First, each of the similarity functions was applied to all activity pairs from the two development datasets. Here, the set of all corresponding and the set of all non-corresponding activity pairs for both datasets are considered as representative samples for both classes. At this point, it should be noted that five of the process models in the UA dataset are not sound. As soundness is a necessary prerequisite for computing the execution semantics properties, these properties can only be determined for six out of the 36 model pairs on UA. Due to this restriction these properties are only examined on BR.

TABLE 5.7: p-values of the Kolmogorov–Smirnov test for BR and UA

Dataset	$\sigma.\pi_{\rightarrow a}$	$\sigma.\pi_{a\rightarrow}$	$\sigma.\pi_{\bullet a}$	$\sigma.\pi_{a\bullet}$	$\sigma.\pi_{\bullet a\bullet}$	$\sigma.\pi_{\downarrow R}$	$\sigma.\pi_{\bullet R\bullet}$
BR	**.001**	**.004**	.994	.581	.699	.016	**.002**
UA	**.000**	**.000**	.281	.155	.367	.967	.155

Dataset	$\sigma.\pi_{\rightsquigarrow}$	$\sigma.\pi_{\rightsquigarrow^{-1}}$	$\sigma.\pi_{+}$	$\sigma.\pi_{\parallel}$
BR	**.000**	**.000**	.111	.994
UA	-	-	-	-

To assess whether the similarity functions separate non-corresponding from corresponding activity pairs the respective value distributions within these sets are compared for each of the functions. As shown in Table 3.4 only 4.4% of the activity pairs on BR are correspondences and 2% on UA. Accordingly, there are roughly 22 times more non-corresponding activity pairs than correspondences on BR and even almost 50 times more on UA. As the huge imbalance of non-corresponding and corresponding activity pairs would distort the analysis 100 activity pairs were randomly selected per class and dataset. Next, for each dataset and similarity function a two-sided Kolmogorov-Smirnov test [43] at a significance level of 0.01 was conducted. The neutral hypothesis of this test is that the examined data samples come from the same distribution. It is rejected, if the p-value yielded by the test is lower than the significance level. With regard to the classification of activity pairs, the test can hence be used to analyze whether the similarity functions assign different values to non-corresponding and corresponding activity pairs. That is, the rejection of the neutral hypothesis for a certain similarity function is considered as an indicator for the suitability of the similarity. Conversely, a similarity function is not suited for activity pair classification, if the neutral hypothesis is accepted. Table 5.7 summarizes the p-values yielded for each similarity function and dataset. Bold values highlight p-values that are below the significance level.

As the table reveals there are only two similarity functions ($\sigma.\pi_{\rightarrow a}$, and $\sigma.\pi_{a\rightarrow}$) for which the null hypothesis is rejected on both datasets. Moreover, the null hypothesis is only rejected on BR for $\sigma.\pi_{\bullet R\bullet}$, $\sigma.\pi_{\rightsquigarrow}$, and $\sigma.\pi_{\rightsquigarrow^{-1}}$. From this analysis, these five similarities are considered as candidates for the extension of the label-based classification.

While the analysis gave evidence that only five of the similarity functions yield different value distributions for corresponding and non-corresponding activity pairs, it does not allow to judge how well the sets of non-corresponding and corresponding activity pairs can be separated with regard to these functions. Hence, in order to further substantiate the analysis, the *information gain* is computed for each of the functions per dataset. The information gain is a well-established measure from statistics [44] and can be used to examine the goodness of the separation achieved by the similarity functions. For all of the similarity functions the ratio of all corresponding and non-corresponding activity pairs in a dataset serves as a reference point. This ratio is encoded in terms of the *Shannon entropy* [307, 308]. It can be seen as the worst case scenario where all activity pairs are classified equally and thus no separation is achieved. For each similarity function it is then investigated how well it can improve this classification. Therefore, a threshold ϑ is introduced that splits the interval of $[0,1]$ into the two intervals $[0,\vartheta)$ and $[\vartheta,1]$. Each of the activity pairs is assigned to one of the intervals according to the value yielded by the similarity function. For both intervals the Shannon entropy is calculated again and based on these values the information gain provides information to which extent the initial ratio was improved. In the worst case the ratio of corresponding and non-corresponding activity pairs is equal to the initial ratio. Here, the information gain takes a value of 0 which indicates that the similarity function in combination with the threshold value does not separate corresponding from non-corresponding activity pairs. On the contrary, the higher the information gain the better the separation. Note that the information gain depends on the initial entropy value and is thus not bound to a specific interval. Instead, it is a relative measure which allows to compare different classifications.

The information gain for the similarity functions is determined for each dataset independently. Additionally, all different similarity scores yielded by a function were considered as threshold values. For each of these threshold values the information gain is computed and the highest score is selected. Table 5.8 summarizes the information gains for each similarity function. Moreover, it introduces the information gain of the bag-of-words similarity $\sigma.\varpi$ as a baseline. In this regard, HAM is used as a word similarity and stemming and pruning are not applied.

TABLE 5.8: Information gain for the selected attributes on BR and UA

Dataset	$\sigma.\varpi$	$\sigma.\pi_{\rightarrow a}$	$\sigma.\pi_{a\rightarrow}$	$\sigma.\pi_{\bullet R\bullet}$	$\sigma.\pi_{\rightsquigarrow}$	$\sigma.\pi^{-1}_{\rightsquigarrow}$
BR	**.041**	.010	0.004	.000	.011	.008
UA	**.018**	.006	0.006	.000	-	-

Table 5.8 shows that $\sigma.\varpi$ yields the highest and $\sigma.\pi_{\bullet R\bullet}$ the lowest information gain, while the remaining similarity functions rank in between. In comparison to the bag-of-words similarity, all the property similarity functions yield low information gains. On BR $\sigma.\pi_{\rightsquigarrow}$ is the property similarity with the highest information gain but it only achieves 25% of the bag-of-words similarity. Likewise, $\sigma.\pi_{\rightarrow a}$ and $\sigma.\pi_{a\rightarrow}$ only achieve 33% on BR. Considering the low effectiveness values achieved by the bag-of-words similarity (cf. Chapter 4) and the relatively low information gain of the property similarities, this analysis shows that none of the property functions is suited to improve the one dimensional, label-based classification. To convey a better intuition for this result Fig. 5.4 visualizes the distribution of the similarity values for three similarity functions in terms of box plots. Here, $\sigma.\varpi$ represents the highest, $\sigma.\pi_{\rightarrow a}$ a medium, and $\sigma.\pi_{\bullet R\bullet}$ the lowest information gain. The figure clearly confirms the analysis results as the distributions for the property similarities do not differ as strongly as the distributions for $\sigma.\varpi$.

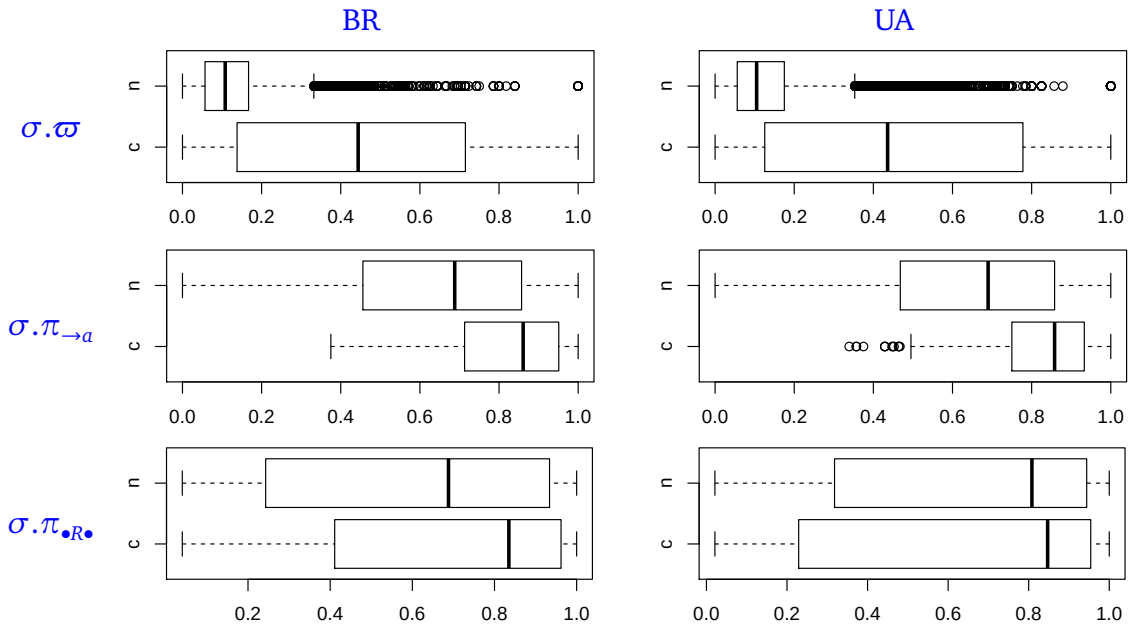

FIGURE 5.4: Box plots for corresponding (c) and non-corresponding (n) activity pairs

5.2 Patterns for Activity Cluster Detection

The previous section examined the extension of the pairwise classification of activities. The basic strategy in this regard is to classify each activity pair separately. However, as outlined in Sections 3.1 and 3.4 model collections might contain complex correspondences. That is, an activity or a set of activities corresponds to a set of activities in another process model. In fact, it cannot be ruled out that the matching techniques introduced so far, detect such complex correspondences, but they do not explicitly address complex correspondences. Moreover, the challenge analysis in Section 4.6 revealed that the label-based matching techniques struggle with identifying such correspondences.

To enhance the detection of complex correspondences, some matching techniques from prior research exploit structural relations between activities from the same process model to identify candidates for such complex correspondences. In this regard, Branco et al. [16] rely on the assumption that such candidates can be inferred from RPST fragments. Yet, their evaluation indicates that the RPST is unsuited to detect sets of activities in a process model that are part of complex correspondences. Another approach that relies on the same assumption is presented by Ling et al. [204]. Similarly, Dijkman et al. [116] propose a post-processing step in which elementary correspondences are extended by subsequently adding neighbors of activities to these elementary correspondences. The evaluation in [116] shows that the post-processing step has a positive, but marginal effect on the effectiveness.

This section takes on the ideas from prior research and examines the nature of sets of activities that are part of complex correspondences. In alignment with the definitions in Section 3.1 such sets of activities are referred to as corresponding activity clusters in the following. In particular, the investigation in this section aims to derive structural patterns of corresponding activity clusters. Similar to the state of the art the idea is to reuse the identified patterns to derive candidates for complex correspondences before, during, or after the matching process.

To reveal such structural patterns a categorizing qualitative analysis [45] was carried out based on the two development datasets. More precisely, all corresponding activity

clusters contained in these two datasets constituted the analysis unit. Based on the assumption that RPST fragments are an indicator for corresponding activity clusters [16, 204], the initial set of structural patterns comprised the three non-trivial RPST fragment types: bond, polygon, and rigid (cf. Section 5.1.2). In a first iteration occurrences of these patterns were marked. Afterwards, all activity clusters that were not marked in the first iteration, were iteratively analyzed. That is, additional structural patterns were derived and their occurrences were marked until every activity cluster was assigned to a pattern. The final, consolidated pattern catalog contains eight patterns that are introduced in the following. In this regard, the patterns are illustrated based on BPMN models. Note that the development datasets actually contain Petri Net models. However, to keep the examples concise, BPMN was used here.

Polygon. The first category is the *polygon* pattern and refers to eponymous RPST fragments. As introduced in Subsection 5.1.2, a polygon is a maximal sequence of nodes in the process model. However, there is one limitation. That is, the polygon pattern refers to polygon fragments that only contain trivial fragments. An example is shown on the left side of Figure 5.5.

Sequence. Compared to the polygon pattern the *sequence* pattern is more general. Like a polygon a sequence is a connected sub-graph of the process model in which there is only one path leading from the first activity in the sequence to the last activity in the sequence. All other nodes in the sequence lie on this path and do not have any other edges that connect them to nodes outside the sequence. But, in contrast to a polygon, a sequence is not maximal. That is, there is at least one other activity in the process model that needs to be added in order to transform the sequence into a polygon. The difference between polygons and sequences is shown in Figure 5.5. The sequence on the right side in the figure consists of the activities d and e. Adding activity c to this sequence, results in the

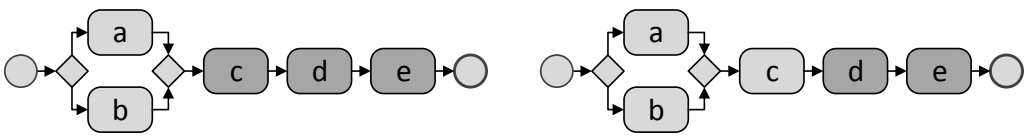

FIGURE 5.5: Examples of the polygon (left) and the sequence pattern (right)

polygon shown on the left side. Unlike the sequence, the polygon cannot be extended by another activity in a way that the result is still a sequence or a polygon. Although sequences subsume polygons, they are treated separately here.

Path. The *path* pattern corresponds to the definition of path introduced in Section 5.1.1. It is similar to the sequence pattern insofar that it also represents sub-graphs where all activities are on a path from the start to the end node. Yet, contrary to sequences the nodes on the path have edges that connect them to nodes that are not part of the path. As a result, the behavioral characteristics of such a sub-graph differ from those of a sequence or a polygon. In more detail, if the first activity of a sequence (polygon) occurs in a trace of a process model, it is followed by all activities that are part of the sequence (polygon) in the same order as they occur in the sequence (polygon). This does not hold for the path pattern. First, not all activities in a sub-graph adhering to the path pattern need to occur in the same traces. The left side of Figure 5.6 shows the sub-graph $\{a, b\}$ where the traces that contain a only partly overlap with the traces that contain b. The reason is the alternative block that leads to two traces for the entire process model: $a \mapsto b \mapsto d \mapsto e$ and $a \mapsto c \mapsto d \mapsto e$. While a occurs in both traces, b is only part of the first. Second, if all activities of the sub-graph occur in the same traces, other activities might occur in between them. This is shown on the right side of Figure 5.6 where there is the corresponding activity cluster $\{a, b, c\}$ that adheres to the path pattern. Here, the entire process model also has two traces: $a \mapsto b \mapsto c \mapsto d \mapsto e$ and $a \mapsto c \mapsto b \mapsto d \mapsto e$. As a consequence of the parallel block, c occurs either between a and b or between b and d in the traces.

Bond. The *bond* pattern describes sub-graphs that correspond to bond-fragments in the RPST of the process model. As outlined in Subsection 5.1.2 a bond is a set of at least two other RPST fragments that share the same boundary nodes. The upper left corner of Figure 5.7 shows a respective example. The corresponding activity cluster $\{b, c, d\}$

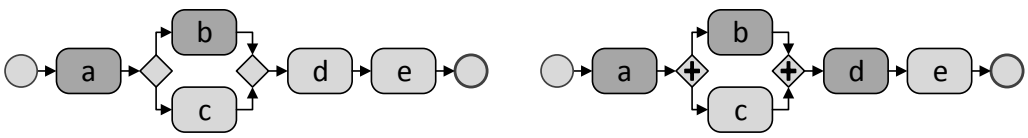

FIGURE 5.6: Examples of the path pattern

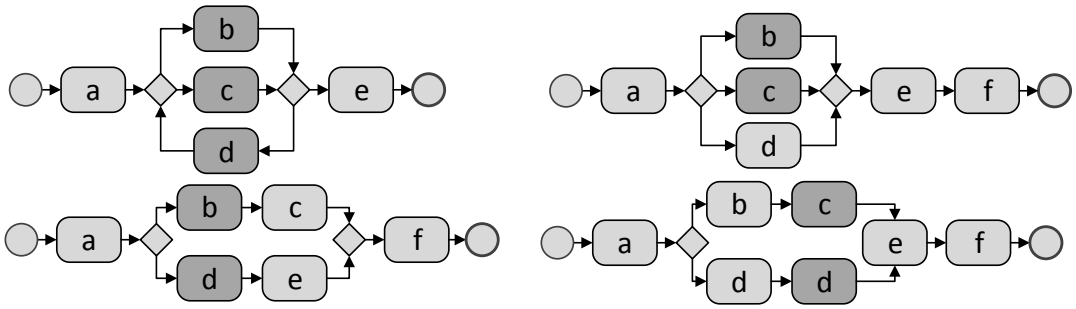

FIGURE 5.7: Examples of the bond (upper left corner) and the partial bond pattern (rest)

comprises all activities that lie between the exclusive gateways. Furthermore, each activity is on a different path that connects these gateways.

Partial bond. The *partial bond* pattern describes connected sub-graphs that consist of nodes which are all part of the same bond fragment. In contrast to the bond pattern, not all activities of the bond are part of the sub-graph, but the activities have to occur in more than one sub-fragment of the bond and there must be a connected sub-graph containing all the activities from the cluster and maybe nodes of other types, but no other activities. A first example of a partial bond is presented in the upper right corner of Figure 5.7. The activities b, c, and d occur in a bond, but only b and b are part of the corresponding activity cluster. The other two examples in Figure 5.7 constitute sub-graphs where only parts of the sub-fragments of the bonds occur in the partial bond. In the the lower left corner activities b and d are part of a partial bond and are also part of different sub-fragments of the bond. Here, both activities are connected to the split. Likewise, the example in the lower right corner outlines a case where activities c and e form a partial bond. As outlined in the context of the model transformation rules for the RPST calculation (Section 5.1.2), activity e implicitly comprises a parallel gateway. Thus, this gateway together with activities b and d is considered to be part of a connected sub-graph.

Fragment sequence. The *fragment sequence* pattern characterizes sub-graphs of a process model where all activities of at least two non-trivial RPST fragments are part of the sub-graph. Furthermore, the fragments must be arranged in sequence so that they either share a boundary node or are connected by an edge or a trivial RPST fragment, respectively. Two examples of this pattern are shown in Figure 5.8. The sub-graph on the left side

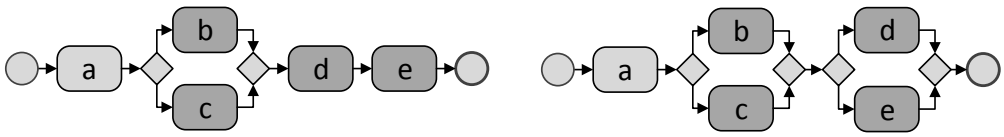

FIGURE 5.8: Examples of the fragment sequence pattern

comprises a bond fragment that the activities b and c belong to and a polygon containing the activities d and e. The exclusive join gateway connects both fragments. Similarly, the two bond fragments on the right side constitute a fragment sequence. Both bond fragments are connected by a trivial fragment that connects the exclusive join gateway of the first bond with the exclusive split gateway of the second bond.

Arbitrarily connected sub-graph. The *arbitrarily connected sub-graph* comprises all connected sub-graphs that do not adhere to one of the other patterns. Figure 5.9 depicts two examples of this category which are extensions of the partial bond in Figure 5.7. On the left side the partial bond that comprises the activities b and d is extended by adding activity a. This way, it does not adhere to the partial bond pattern anymore. Instead, it is now classified as an arbitrarily connected sub-graph. Accordingly, on the right side activity e is added to the partial bond consisting of the activities c and d. As a result an arbitrarily connected sub-graph is yielded.

Disconnected sub-graph. The last identified pattern is the *disconnected sub-graph* pattern. A sub-graph characterized by this pattern comprises activities that cannot be connected without adding other activities from the process model to the sub-graph. Such a sub-graph is shown in Figure 5.10. This sub-graph comprises the activities a, d, and e. While there is an edge that connects d and e, the two gateways and at least activity b or c must be added in order to also connect a to these activities.

Rigid. Initially, the *rigid* pattern as the last representative of the non-trivial RPST fragments was part of the pattern catalog. As there were no occurrences within the development

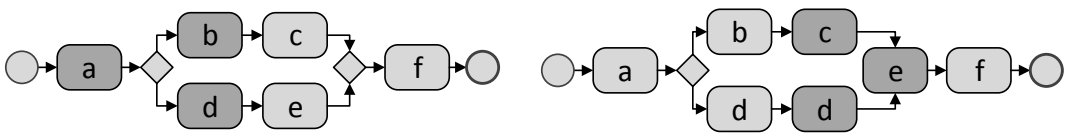

FIGURE 5.9: Examples of the arbitrarily connected sub-graph pattern

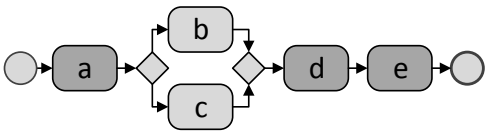

FIGURE 5.10: Example of the disconnected sub-graph pattern

datasets, it was removed from the catalog. Nevertheless, from a theoretical point of view rigids are subsumed by the arbitrarily connected sub-graph pattern.

The pattern catalog covers a broad variety of structural relations between activities in corresponding activity clusters. Some patterns can be considered as strict, whereas others are rather inaccurate. In this regard, the bond and the polygon patterns are strict patterns. That is because they require clusters to correspond to an RPST fragment and thus rely on a clear criterion. On the contrary, the disconnected sub-graph pattern is the most inaccurate pattern as there are no structural relations that can be used to detect respective activity clusters. All remaining patterns can be classified as connected sub-graphs that do not represent an RPST fragment and thus they rank in between the other patterns. However, their strictness differs. For example, the fragment sequence can be considered as rather strict as it comprises clusters that can be inferred from the RPST. In contrast, arbitrarily connected sub-graphs instead are very inaccurate as they only require nodes to span a connected sub-graph.

In order to utilize a pattern for the detection of candidates of complex correspondences, it is desirable that many sets of activities that adhere to the pattern actually constitute corresponding activity clusters. In this context, the strictness of a pattern is an important criterion because it can be assumed that the more precise a pattern is, the smaller the number of activity sets that adhere to the pattern. To illustrate this assumption, the numbers of non-trivial RPST fragments, connected sub-graphs, and sub-graphs are determined (including connected and disconnected sub-graphs) in the development datasets. In the BR dataset there are 211 non-trivial RPST fragments and in the UA dataset 229. On the contrary, BR contains 125,321 distinct connected sub-graphs and UA 5,535,807,993. Finally, there are even 52,969,801 sub-graphs in the BR dataset and 281,760,613,146,367 in the UA dataset. This explosion of the amount of potential candidates substantiates that patterns need to be strict in order to limit the number of potential candidates.

TABLE 5.9: Absolute (abs), relative (rel), and cumulative (cul) frequencies of the patterns

	BR			*UA*		
Pattern	*abs*	*rel*	*cul*	*abs*	*rel*	*cul*
Polygon	2	.035	.035	0	.000	.000
Bond	6	.105	.140	1	.019	.019
Fragment Sequence	6	.105	.245	0	.000	.019
Sequence	5	.088	.333	9	.170	.189
Path	9	.158	.491	1	.019	.208
Partial Bond	7	.123	.614	13	.245	.453
Arbitrarily Connected Sub-Graph	15	.263	.877	11	.208	.661
Disconnected Sub-Graph	7	.123	1.00	18	.340	1.00
Σ	57	1.00		53	1.00	

With that in mind, the extent to which corresponding activity clusters rely on strict patterns is examined next. Here, Table 5.9 presents the absolute and the relative frequencies of the patterns within the datasets. The relative frequency can be interpreted as a recall value, i.e., how many of the truly existing activity clusters are retrieved, if all activity sets that adhere to such a pattern are selected. Furthermore, the patterns are arranged in descending order with respect to their strictness and the cumulative frequencies are shown.

According to the table, only a small portion of the corresponding activity clusters corresponds to RPST fragments in the development datasets. In total, 14% of all corresponding activity clusters adhere to one of these two patterns on BR and on UA only 1.9%. The fact that each of the datasets contains more than 200 RPST fragments, shows that applying these patterns does not only result in a small recall, but also in a small precision. Thus, the results confirm the evaluation results from [16] that the RPST is an unreliable means for the detection of corresponding activity clusters.

Relaxing the strictness of the patterns by considering the fragment sequence, the sequence and the path pattern does not lead to large improvements in the cumulative frequency. That is, on BR it is only lifted to 49.1% and to 20.8% on UA. In contrast to

these rather strict patterns, a large portion of corresponding activity patterns adhere to inaccurate patterns. That is 50.9% of the corresponding activity clusters are characterized by rather inaccurate patterns on BR and 79.2% on UA. In this regard, on UA a surprisingly large amount of 34% of all corresponding activity clusters does not even constitute a connected sub-graph. These results outline that in order to detect all corresponding activity clusters an enormous amount of potential candidates, i.e., all sub-graphs, need to be considered. Thus, the analysis indicates that structural relations cannot reliably be exploited to detect complex correspondences.

5.3 Alignment Consistency

The analysis in this section reposes on the assumption that the relative positions of activities in a process model are similar to the relative positions of their corresponding counterparts in a different model. Accordingly, an alignment where the relative positions of the correspondences resemble each other is referred to as a *consistent* alignment. The alignment for the university admission processes from the running example is consistent (cf. Figure 3.1). Here, α_1 and α_2 correspond to β_1, α_3 to β_2, as well as α_4 and α_5 to β_3, β_4, and β_5. In the first process α_1 and α_2 are in a parallel block which is followed by α_3 which is succeeded by an alternative block that contains α_4 and α_5. The correspondences of these activities in the second process show the same ordering. That is, β_1 is the first activity in the process which is followed by β_2 which is connected to the cluster containing β_3, β_4, and β_5.

In this manner, some matching techniques from related work consider relations between corresponding activities. Leopold et al. [194] optimize alignments based on constraints. Some of these constraints impose the requirement that correspondences have to have similar control flow relations in both processes. For example, if an activity a_1 is in strict order with a_2 in a process and a'_1 is in strict order with a'_2 in another process, matching a_1 to a'_1 and a_2 to a'_2 satisfies the constraint. Similar approaches are proposed in [162, 201]. Yet, in [194] such constraint slightly improve the effectiveness, whereas they have a negative impact in [162].

These ideas are related to the work by Smirnov et al. [309] who derive action patterns from process models in a model collection. Such patterns capture control flow constraints between generic actions, e.g., that making a decision typically requires an assessment to be carried out beforehand. Once those patterns are derived, they can be reused when new models are created to verify that the new model satisfies common practices. Furthermore, the area of *process similarity search* comprises approaches that investigate the consistency of models. For instance, in [146] the graph edit distance for alignment construction [116] is adapted to process similarity search and compared to a measure that analyzes possible execution traces. Another approach that relies on traces is the trace index similarity [310]. In contrast, the workflow similarity in [311] is based on the number of corresponding nodes and edges, like the edit distance [116]. The measure in [312] considers the depth of activities in process trees. An overview of similarity measures is provided in [147]. Similar to many of these approaches, the relative position of activities is investigated in the following. Yet, in contrast to the similarity measures, the process model matching and thus non-corresponding nodes are disregarded, as the goal is to examine the consistency of alignments rather than that of process models.

In particular, the *order relation score* is introduced. It measures the consistency of an alignment by checking whether the ordering of activities in one process model is similar to that of their corresponding counterparts in the other model. There are different variants of this score which rely on one of the three position property functions, $\pi_{\rightarrow a}$, $\pi_{a\rightarrow}$, and $\pi_{\downarrow R}$ from Section 5.1. Note that the execution semantics properties $\pi_{\rightsquigarrow}$ and $\pi_{\rightsquigarrow^{-1}}$ are neglected here, as they require models to be sound, which is a limiting factor. Moreover, in the analysis in Section 5.1 they performed similar to the other three properties.

For each of these three position functions π_x a respective order relation score δ_x is defined. Basically, the order relation score δ_x is defined with regard to a set of alignments. To calculate the score, an alignment score γ_x is computed for each alignment in the set. That is, for each pair of distinct correspondences from an alignment (c_1, c_2) with $c_1, c_2 \in \mathscr{A} \wedge c_1 \neq c_2 \wedge c_1 = (a_1, a_1') \wedge c_2 = (a_2, a_2')$, it is checked whether the order of the activities a_1, a_2 from the first process is equal to the order of the activities a_1', a_2' from the second process. Therefore, a test is carried out which examines if $\pi_x(a_1) - \pi_x(a_2)$

and $\pi_x(a_1') - \pi_x(a_2')$ have the same sign, or at least one is 0: then the correspondence pair yields 1, otherwise 0. Next, the sum of all values is averaged over the number of correspondence pairs to yield γ_x. In other words, the alignment score is the percentage of all pairs of distinct correspondences from an alignment for which the respective activities have the same relative order in a process model. Finally, to compute the order relation score δ_x, the scores yielded for the alignments are averaged.

Definition 5.11 (Order relation score) *Given a set of alignments $\mathscr{A}^*$ and a position property $\pi : A \to [0,1]$ the order relation score δ is defined as:*

$$\delta(\mathscr{A}^*) := \frac{1}{|\mathscr{A}^*|} \sum_{\mathscr{A} \in \mathscr{A}^*} \gamma(\mathscr{A})$$

with

$$\gamma(\mathscr{A}) := \frac{\sum_{c_1 \in \mathscr{A}} \sum_{c_2 \in \mathscr{A} \setminus \{c_1\}} \gamma_c(c_1, c_2)}{|\mathscr{A}| \cdot (|\mathscr{A}| - 1)}$$

where $c_1 = (a_1, a_1')$, $c_2 = (a_2, a_2')$ and

$$\gamma_c(c_1, c_2) := \begin{cases} 1 & [\pi(a_1) - \pi(a_2)] \cdot [\pi(a_1') - \pi(a_2')] \geq 0 \\ 0 & \textit{else} \end{cases}$$

To test if the assumption holds, the order relation score is determined for each development dataset and each position property function. In this regard, all gold standard alignments are considered to compute the scores. As Table 5.10 shows, high values are yielded for all scores on both datasets with $\delta_{\to a}$ resulting in the highest values. This indicates, that for a high percentage of correspondence pairs, the respective activities have the same ordering in their process models. However, there are also exceptions which partly arise from m:n-correspondences: consider a process model pair for which the alignment contains one complex correspondence consisting of a_1, a_2 from the first and of a_1', a_2' from the second process. In this case, the complex correspondence is represented by four elementary correspondences (a_1, a_1'), (a_1, a_2'), (a_2, a_1'), and (a_2, a_2'). Respectively, there are six pairs of distinct correspondences. Further, let the position of a_1 (a_1') be smaller than that of a_2 (a_2'). In this case, the order relation does not hold for the pair $((a_1, a_2'), (a_2, a_1'))$ as the activities occur in reverse order. Thus, the score is not 1, but

TABLE 5.10: Order relation scores of the gold standards on BR and UA

Dataset	$\delta_{\rightarrow a}$	$\delta_{a\rightarrow}$	$\delta_{\downarrow R}$
BR	**.92**	.81	.85
UA	**.93**	.89	.81

$.8\overline{3}$. Nevertheless, the high values suggest that the effect of m:n-correspondences and the exceptions is rather small.

The analysis is further refined in order to examine whether a high order relation score is a distinctive characteristic of the true alignments or if it is an arbitrary characteristic that holds for any (or at least many other) alignments. Therefore, a diverse range of alignments was simulated. These alignments can be interpreted as results of different matchers. In particular, 1,000 sets of alignments were randomly generated for both development datasets. Each set of alignments comprises one alignment per model pair in the model collection. To simulate the full range of matcher results the generation was controlled such that the micro f-measures of the sets were equally distributed over the interval $[0, 1]$. Then, the collection order relation score was computed for each set of alignments and position property function. Finally, the correlation between all pairs of scores and the micro f-measure was examined in order to assess whether the order relation scores systematically differ for sets of alignments with a different quality. To this end, *Spearman's rank correlation coefficient* (ρ) [313] was applied and the results are presented in Table 5.11.

The coefficients show a strongly positive correlation between all variables on both datasets. The findings are significant for all variable pairs as all p-values are much smaller

TABLE 5.11: Correlation coefficients on BR and UA

	BR				*UA*			
	F_μ	$\delta_{\rightarrow a}$	$\delta_{a\rightarrow}$	$\delta_{\downarrow R}$	F_μ	$\delta_{\rightarrow a}$	$\delta_{a\rightarrow}$	$\delta_{\downarrow R}$
F_μ	-	**.97**	.95	.95	-	**.97**	.97	.88
$\delta_{\rightarrow a}$	**.97**	-	.96	.96	**.97**	-	.98	.91
$\delta_{a\rightarrow}$	.95	.96	-	.94	.97	.98	-	.89
$\delta_{\downarrow R}$	.95	.96	.94	-	.88	.91	.89	-

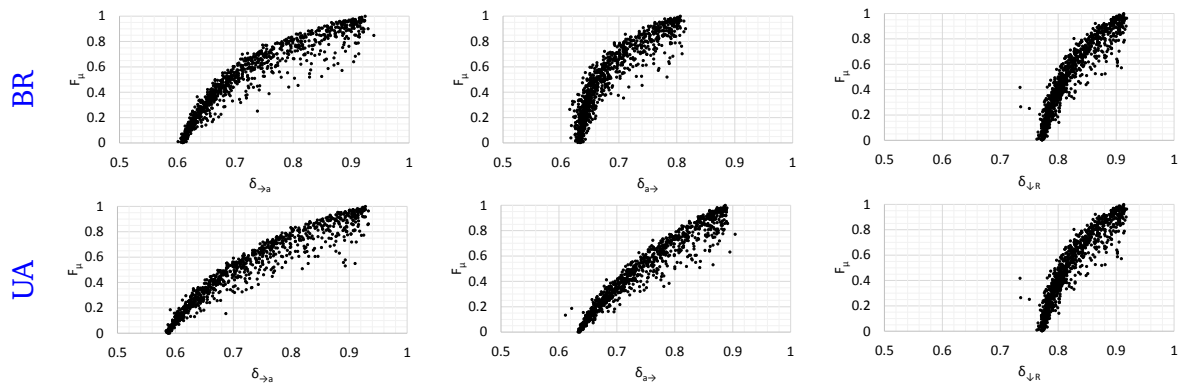

FIGURE 5.11: Scatter plots for the order relation scores vs the micro f-measure

than .001. Thus, it can be concluded that the alignment scores are connected to the micro f-measure. That is, high scores indicate a high micro f-measure. Additionally, alignments of a low micro f-measure typically yield low order relation scores. This relation also holds in the reverse direction. However, the results also reveal that the scores are strongly correlated among themselves. Thus, it is only meaningful to consider one of the scores. In this regard, the start distance-based score $\delta_{\to a}$ yields the highest score for the gold standards and also shows the strongest correlation to the micro f-measure F_μ. Hence, it is proposed as a means to investigate how well alignments proposed by a matcher preserve the order between the corresponding activities. This decision is also supported by the scatterplots in Figure 5.11. These diagrams show that the range of score values is the largest for $\delta_{\to a}$ and the smallest for $\delta_{a\to}$. Consequently, $\delta_{\to a}$ has the highest discriminative power to separate sets of alignments with low f-measures from those with high f-measures. In the remainder of this thesis, the term order relation score is used to refer to the start distance-based variant.

5.4 The Order Preserving Bag-of-Words-Technique

The analyses of the behavioral perspective revealed that sets of alignments which are close to the objective truth, i.e., for which a high micro f-measure is yielded, are likely to take higher order relation score values than those sets which differ greatly from the objective truth. Consequently, the order relation score can be used to solve the configuration problem. That is by considering it as an approximation for the effectiveness of matcher

results, it can be used to estimate the matcher quality *without* knowing the true alignments and thus making human intervention obsolete while maximizing the effectiveness. In other words the goal is to select features of BOT so that the resulting configuration outperforms the default configuration, comes close to the maximum effectiveness, and does not require the manual provision of training alignments.

As already pointed out in Chapter 4, the problem of *matcher configuration* has been recognized as a central challenge in schema and ontology matching [171, 269, 270]. Accordingly, several approaches have been proposed to deal with the configuration problem, see [171, 269] for an overview. Basically, such approaches can be divided into two classes. The first class comprises approaches that rely on human intervention. For example, Peukert et al. [314] developed a software tool that assists users in manually assembling and refining schema matchers. In the context of process model matching, the manual provision of domain ontologies was proposed in [160]. The other class of approaches addresses the automated configuration. Here, eTuner [315] assesses the quality of different matchers based on a set of schema pairs which it automatically derives from a given schema. Additionally, ontology matchers are viewed as individual agents that negotiate in order to reach an agreement on alignments [316]. Complementary to these works, the Order Preserving Bag-of-Words Technique (OPBOT) which is introduced in the following addresses the configuration of process model matchers and uses process specific control flow information to achieve this.

In this context, the prediction framework for process model matching by Weidlich et al. [197] (cf. Section 3.3.3) is closely related to OPBOT. The rationale of this framework is to use a set of alignments identified by experts to train a prediction model. This model correlates process model properties and process similarity measures to the effectiveness of matchers. Once the model has been trained, it can be used to select the most promising matcher for a given model pair. However, in contrast to OPBOT which is an applicable matching technique and which utilizes the empirically verified order relation score, the framework constitutes a generic architecture for which a set of prediction means has been proposed, but no evidence was given towards their applicability. Moreover, the framework does not contain any specific matchers.

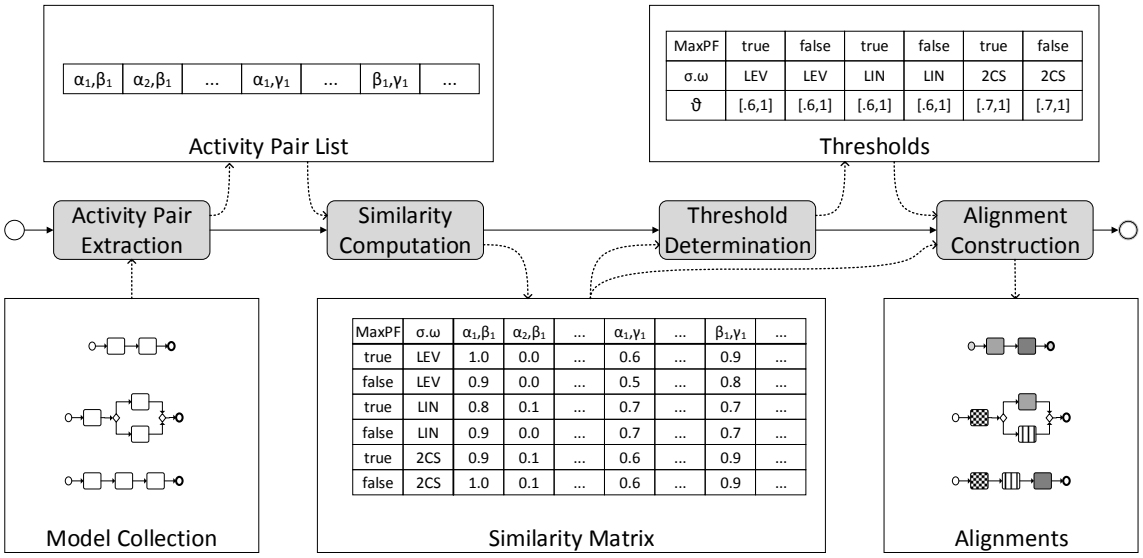

FIGURE 5.12: The OPBOT match workflow

As explained, the idea behind OPBOT is to automate the search for a BOT configuration that yields an effectiveness close to that of the optimal BOT configuration. A straightforward search strategy is to simply exploit the strong correlation between the order relation score $\delta_{\rightarrow a}$ and the micro f-measure F_{μ}. That is, all possible configurations are considered and for each configuration $\delta_{\rightarrow a}$ is computed. Then, the configuration with the highest score $\delta_{\rightarrow a}$ is proposed. Besides being computationally expensive, this strategy is prone to select outliers as the order relation score is an approximation of the effectiveness. This can be illustrated with regard to the scatter plots in Figure 5.11. These plots show that for a certain order relation score value different micro f-measures might be observed and that the range of possible micro f-measures overlaps for different score values. For instance, on BR for $\delta_{\rightarrow a} = .7$ the micro f-measures span the interval [.44, .64], and for $\delta_{\rightarrow a} = .75$ the interval [.48, .8]. Consequently, the configuration with the higher score $\delta_{\rightarrow a}$ might actually have a lower effectiveness, e.g., $F_{\mu} = .64$ at $\delta_{\rightarrow a} = .7$ vs. $F_{\mu} = .48$ at $\delta_{\rightarrow a} = .75$. Thus, OPBOT's search strategy must minimize the chance of selecting outliers while still maximizing the chance of detecting the optimal configuration.

With that in mind, different search strategies were developed and evaluated on the development datasets. The final strategy is depicted in Figure 5.12. On an abstract level, it is based on ideas from the general match workflow from schema matching [140] (cf. Section 3.1). That is it simultaneously executes different BOT configurations and

combines their results. Moreover, it processes the entire model collection at once. The reason is that looking for configurations per model pair is prone to select poorly performing configurations due to the limited extent of data. On the contrary, performing the search based on all model pairs in a model collection has the advantage that the influence of outliers is diminished. In the following, each step of OPBOT's workflow is described.

Activity pair extraction: At the beginning the entire model collection is processed to extract all activity pairs from the model collection. In this regard, all models are loaded and all activity labels are normalized. Then, the list of all activity pairs is constructed.

Similarity calculation: As explained above, considering the whole space of configurations entails the risk of favoring outliers. Thus, this space is reduced in this step by selecting promising BOT features. Therefore, the optimal BOT configurations on the development datasets are considered (cf. Table 4.10). In both configurations filtering is enabled and PSA is selected to stem words. Accordingly, the other values for these features are neglected. The optimal configurations differ with regard to the word similarity, the use of pruning and the threshold. For the pruning feature both options, i.e., the application of MaxPF and the deactivation of pruning, are considered. With regard to the word similarity function, the range of possible functions was limited by only regarding one syntactical, one paradigmatic, and one syntagmatic similarity function. From all corresponding word similarity combinations, the one was chosen for which OPBOT yielded the best results on the development datasets. As a result, LEV was selected as a syntactical, LIN as a paradigmatic, and 2CS as a syntagmatic word similarity. Lastly, the range of possible threshold values was also reduced to minimize the risk of detecting outliers. Here, the evaluation results from Chapter 4 were considered to select an interval for which it can be assumed that it contains the optimal threshold for these three word similarities. Hence, the range of threshold values was limited to the interval of $[.6, 1]$ for LEV and LIN. For 2CS it was limited to the interval of $[.7, 1]$. Figure 5.13 summarizes the reduced space of possible BOT configurations.

To prepare the search through the reduced space of possible configurations, a similarity matrix is determined in this step. For each of the activity pairs in the model collection there

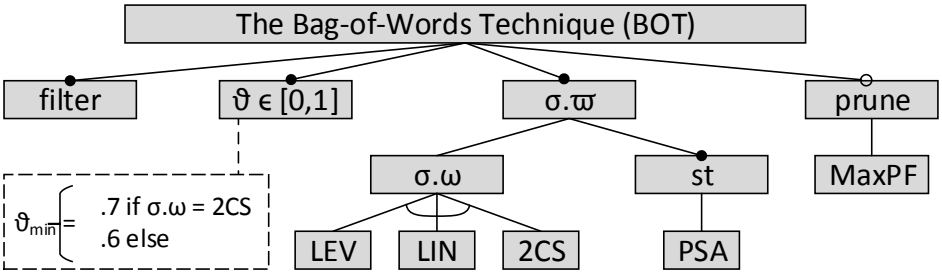

FIGURE 5.13: The feature model for the reduced space of BOT configurations

is exactly one column in the matrix. Additionally, there is one row per BOT configuration. In total there are six configurations as each word similarity is combined with each of the two pruning options. As the determination of similarity scores is independent of the threshold value, no specific threshold value is selected for the configurations in this step. To fill the matrix each model pair is processed by each of the BOT configurations. In this regard, the similarity values for the activity pairs are set to the score yielded by the bag-of-words similarity. Moreover, all activity pairs with equal labels that were identified in the filtering step have a similarity score of 1. The similarity value is set to 0 for activity pairs that were removed during filtering.

Threshold determination: Based on the similarity matrix from the previous step, the alignments are constructed for each BOT configuration. That is, the thresholds are optimized by computing the collection order relation score $\delta_{\rightarrow a}$. Here, the set of distinct similarity scores that are larger than or equal to the according minimal threshold ($\vartheta_{min} = .6$ for LEV and LIN; $\vartheta_{min} = 0.7$ for 2CS) are considered as possible threshold values for each row or configuration, respectively. Then, for each threshold value the order relation score $\delta_{\rightarrow a}$ is computed by considering all activity pairs with a similarity value larger than or equal to the threshold value as correspondences. For each configuration the value with the highest score is selected as the threshold, as it is predicted to yield the best effectiveness. Afterwards, the six configurations are ranked in descending order with regard to their order relation score $\delta_{\rightarrow a}$. Finally, per configuration the similarity score for activity pairs is set to 0, if the score is smaller than the optimized threshold.

Alignment construction: The last step, is to create a set of alignments that contains one alignment for each process model pair from the collection. This is accomplished

by combining the results of the top two configurations from the previous step. Again, considering two configurations is done to minimize the risk of favoring outliers. In more detail, for each activity pair the maximum similarity score yielded by one of the two top-ranked configurations is determined. Then, all pairs with a maximum similarity score different from 0 are proposed as correspondences and added to the alignments.

5.5 Evaluation and Analysis

This section presents the evaluation results of OPBOT in order to conclude the verification of Sub-hypothesis H3. Here, the effectiveness of OPBOT is assessed with regard to the development and the evaluation datasets. Following, the general validity of the order relation score $\delta_{\rightarrow a}$ is examined by investigating its correlation to the f-measure on the evaluation datasets and its portability to matcher selection is assessed.

5.5.1 Effectiveness on the Development Datasets

To investigate the effectiveness of OPBOT, the primary focus is on its relative performance. In other words, the question is *how does it perform in comparison to BOT*? Here, the comparison to the configuration with the maximum effectiveness (BOT_{MAX}) allows to assess how well the optimization implemented through OPBOT's search strategy works. Moreover, contrasting OPBOT to the default configuration (BOT_{ALL}) outlines the improvement that is gained by automatically configuring BOT rather than optimizing it on a few model collections. Finally, the effectiveness achieved by the semi-manual configuration approach is considered to investigate, if and to which degree OPBOT unburdens experts from providing training data. First, the focus is on the effectiveness of OPBOT, the maximum and the default BOT configurations, as well as the best performing matchers from the contests [19, 20]. Table 5.12 presents their results.

OPBOT improves the results of the default configuration on both datasets. With regard to the micro f-measure, OPBOT achieves (.520 vs. .452 $\widehat{=}$) 115% of the effectiveness of BOT_{ALL} on BR and (.442 vs. .403 $\widehat{=}$) 110% on UA. Moreover, OPBOT is close to the optimum on BR as it achieves (.520 vs. .534 $\widehat{=}$) 97% of the effectiveness of BOT_{MAX}

TABLE 5.12: Effectiveness of OPBOT, BOT, and the contest matchers [19, 20] on BR and UA

Dataset	*Matcher*	pr_μ	re_μ	F_μ	pr_M	re_M	F_M
BR	OPBOT	.613	.452	.520	.583	.469	.499
	BOT_{MAX}	.652	**.452**	**.534**	.633	**.467**	**.511**
	BOT_{ALL}	**.657**	.344	.452	.615	.329	.382
	RMM/NSCM	-	-	-	.68	.33	.45
	pPalm-DS	.502	.422	.459	.499	.429	.426
UA	OPBOT	**.598**	.350	**.442**	**.578**	.357	.412
	BOT_{MAX}	.406	**.486**	**.442**	.443	**.511**	**.453**
	BOT_{ALL}	.380	.403	.428	.455	.386	.382
	RMM/NSCM	-	-	-	.37	.39	.38

and even (.442 vs. .442 $\widehat{=}$) 100% on UA. These evaluation results show that the use of information from the behavioral perspective supports the optimization of the effectiveness of label-based matching techniques.

The comparison to the best approaches from the matching contests [19, 20] provides further evidence towards the improvements gained by OPBOT. As Table 5.12 reveals, OPBOT achieves higher micro and macro f-measures than the best techniques from the contests on both datasets.

Finally, on the development datasets OPBOT indeed makes the provision of training alignments obsolete. On both datasets the semi-manual configuration achieved the highest average micro f-measure using when nine process models were manually aligned and used for training (cf. Table 4.13). For this training dataset size the average micro f-measure of $\overline{F_\mu} = .50$ is smaller than that of OPBOT ($F_\mu = .52$) on BR. Likewise, on UA OPBOT yields a higher effectiveness than the semi-manual configuration approach ($F_\mu = .44 > \overline{F_\mu} = .42$). Thus, in favor of the Sub-hypothesis H3 these results show that control flow information can be used to maximize the effectiveness of label-based matching techniques.

5.5.2 Effectiveness on the Evaluation Datasets

The evaluation results on the development datasets confirm that the strong correlation between $\delta_{\rightarrow a}$ and F_μ can be exploited to automatically configure BOT. Yet, as the order relation score $\delta_{\rightarrow a}$ was derived from the analysis of these datasets, the general validity of the evaluation results is limited. This leads to the question whether the order relation score $\delta_{\rightarrow a}$ and OPBOT can be applied successfully on other model collections. Thus, to substantiate the findings, OPBOT is assessed on the evaluation datasets next. The respective results are summarized in Table 5.13 where OPBOT is contrasted to the maximum and the default BOT configurations as well as the best performing matcher from the process model matching contest in 2015 [20].

On AW OPBOT performs better than the default configuration BOT_{ALL} as it achieves a relative performance of (.463 vs. .397 $\widehat{=}$) 117%. With regard to the optimal configuration BOT_{MAX} its relative performance is only (.463 vs. .582 $\widehat{=}$) 80%. This low value is attributed to the reduction of the possible BOT configurations. That is, on AW the best results for BOT are yielded for configurations where the filtering is disabled. However, OPBOT only considers configurations where filtering is enabled. In comparison to the best configuration with filtering which achieves a micro f-measure of .481 OPBOT's relative performance is (.463 vs. .481 $\widehat{=}$) 96% and hence is clearly improved. OPBOT's relative performance in comparison to the optimal configuration BOT_{MAX} is (.625 vs. .692 $\widehat{=}$) 90% on SR. But, its effectiveness is lower than that of the default configuration BOT_{ALL} because it amounts to (.625 vs. .658 $\widehat{=}$) 95%. Moreover, OPBOT also performs slightly worse than AML-PM in terms of the micro f-measure (.625 vs. .677).

TABLE 5.13: Effectiveness of OPBOT, BOT, and the contest matcher [20] on SR and AW

	SR			AW		
	pr_M	re_μ	F_μ	pr_μ	re_μ	F_μ
OPBOT	.599	**.653**	.625	.730	.339	.463
BOT_{ALL}	.774	.572	.658	**.959**	.251	.397
BOT_{MAX}	**.887**	.568	**.692**	.616	**.552**	**.582**
AML-PM	.786	.595	.677	-	-	-

To conclude the evaluation of OPBOT, it is compared to the semi-manual configuration approach. On SR this approach yields the highest average micro f-measure for a training dataset size of six and nine. Despite the lower relative performance of OPBOT with regard to the default and maximum BOT configuration, it still makes the provision of training data obsolote, as its micro f-measure is virtually identical to that of the semi-manual configuration ($\overline{F_\mu} = .63$ vs. $F_\mu = .625$). On AW OPBOT's effectiveness is exceeded by the semi-manual configuration, if at least two model pairs are provided for training ($F_\mu = .46 < \overline{F_\mu} = .47$). Again, the reason is that OPBOT neglects the option to turn off filtering. In case that the semi-manual training of BOT also discards this option, the maximum average micro f-measure of $\overline{F_\mu} = .44$ is yielded for a training set size of nine model pairs. This value is below the effectiveness of OPBOT.

Overall, the analysis results confirmed that OPBOT's search strategy is able to detect high performing configurations within the restricted configuration space. Thus, the results verify that the use of control flow information in process model matchers has a positive impact on the effectiveness. But, they also show that OPBOT's effectiveness is in general limited by the effectiveness of the restricted configuration space.

5.5.3 General Validity of the Order Relation Score

In the previous evaluation OPBOT was treated as a black box. Consequently, the evidence towards the general validity of the order relation score $\delta_{\rightarrow a}$ is limited. Thus, the analysis from Section 5.3 is repeated on AW and SR, i.e., the order relation scores for the gold standard alignments and the correlation to the micro f-measure are investigated again. On AW the order relation score for the gold standard is a bit lower than on the development datasets ($\delta_{\rightarrow a} = .86$), but the correlation between $\delta_{\rightarrow a}$ and F_μ is still very strong ($\rho = .97$ with $p \ll .01$). With regard to SR's gold standard the order relation score is much lower ($\delta_{\rightarrow a} = .77$) and the correlation is only moderate ($\rho = .54$ with $p \ll .01$). Unlike the other datasets where all process models refer to the same higher level process, SR contains model pairs where correspondences exist but appear in different contexts, and other pairs without any correspondences. The latter strongly impacts the order relation score as

the alignment score is 0 for all model pairs without correspondences. To investigate the magnitude of this effect, all six model pairs without correspondences in the gold standard were removed from the dataset. Based on the remaining model pairs the order relation score for the gold standard and the correlation were determined again. The result is that both scores are strongly improved ($\delta_{\rightarrow a} = .93$ and $\rho = .81$ with $p \ll .01$). Thus, the strong correlation between $\delta_{\rightarrow a}$ and F_μ is confirmed and further evidence towards Sub-hypothesis H3 is given. Yet, the successful exploitation of control flow information and in particular of $\delta_{\rightarrow a}$ seems to be limited to situations where the models share similarities.

5.5.4 Portability to Matcher Selection

Last, the applicability of the order relation score in the context of matcher selection is examined. That is, the order relation score is used to estimate and compare the effectiveness of different matching techniques rather than of different configurations of the same technique. To this end, the results of the twelve matchers that participated in the second contest [20] were considered. These results are publicly available[30] for BR and SR. Based on the results an order relation score was computed per matching technique and dataset. Then, for both datasets the matching techniques were ranked in descending order with regard to the determined score. In order to assess the goodness of this ranking, the top k matchers in this ranking were compared to the top k matchers with respect to the micro f-measure F_μ. However, to avoid distortion, the model pairs without correspondences on SR were excluded in this analysis. On BR the top performing matcher ($k = 1$) also yields the highest micro f-measure. In contrast, the best performing matcher on SR is not the best ranked matcher with regard to the order relation score. Moreover, on both datasets the top three matchers in the order relation score ranking ($k = 3$) comprise two of the three best performing matchers and the top five ($k = 5$) three of the five best performing matching techniques. Although the best performing matching technique was not ranked first on SR, the top ranked matcher with regard to the order relation score still achieves a relative performance of 89% compared to the best performing matcher on this

[30] https://ai.wu.ac.at/emisa2015/contest.php, accessed: 13/01/2017

dataset. The maximum relative performance among the top three was 98% and among the top five 100%. Overall, the results confirm that the order relation score can be applied in the context of matcher selection and thus Sub-hypothesis H3 is further substantiated. Additionally, the fact that the matchers were developed by other researchers strengthens the validity of the findings. With the comfirmation of the portability of the score to matcher selection the verification of Sub-hypothesis H3 concludes.

5.6 Summary

This chapter dealt with sub-hypothesis H3 and investigated, if the incorporation of behavioral information captured in the process models improves the effectiveness of label-based matching techniques. To this end, three different approaches were pursued.

The first approach was to define similarity functions that in contrast to the label similarities from Chapter 4 are based on activity properties that refer to the behavioral perspective. In this regard, a set of property functions was introduced. This set comprised functions that consider paths in the models, those that repose on the decomposition of process models into hierarchies of fragments, and those that evaluate the execution semantics captured in the process models. Afterwards, the respective similarity functions were evaluated with regard to their suitability to enhance the effectiveness of the label-based classification. The empirical analysis of these similarities on the development datasets falsified that the incorporation of these similarities improves the label-based matching. That is, by showing that these similarities do not separate corresponding from non-corresponding activity pairs on the development datasets, it was verified that the similarities are not universally applicable for process model matching.

Subsequently, a second approach was examined. Here, the problem of identifying complex correspondences was tackled. In particular, the idea was to identify sets of activities within a process model where the activities refer to the same purpose and are thus likely to form a corresponding activity cluster that is part of complex correspondences. Therefore, a qualitative analysis was carried out to investigate whether structural dependencies can be exploited to identify such activity clusters. In this regard, it was

shown that the structural relations within such activity clusters are too diverse, in order to be reliably exploited. While only a small share of corresponding activity clusters within the development datasets can be identified based on strict criteria, the majority of these clusters is characterized by rather versatile structural patterns. Thus, in order to detect all corresponding activity clusters, a very large set of non-corresponding activity clusters needs to be retrieved as well. Again, the analysis falsified the assumption that structural relations can generally be applied to derive candidates for complex correspondences.

In contrast to these two analyses, the investigation of the third approach revealed a way to exploit the behavioral perspective for matching. In more detail, the third approach is based on a measure for the consistency of the structural relations that exist between the corresponding activities within an alignment. In other words, it is based on the assumption that structural relations between activities from a process model also hold between their corresponding counterparts in another model. To investigate this concept, the order relation score was introduced. For each alignment it measures the ratio of pairs of distinct correspondences from the alignment where the relative position of the activities from the first model is similar to that of the activities within the second model. With respect to a set of alignments it is the average of all alignment scores. Moreover, there are three different variants of the score which measure the position of an activity with regard to the start node, the end node, and the RPST. All three variants take high values for the gold standard alignments from the development datasets. This observation was considered as evidence towards the assumption. To refine the evidence, it was investigated, if high order relation scores are a typical characteristic of the objective ground truth, i.e., the gold standard alignments. The corresponding analysis revealed that the degree to which sets of alignments differ from the set of gold standard alignments in terms of the micro f-measure has a strong positive correlation to the order relation score. However, the results also indicated a strong correlation between the three variants of the order relation score. Hence, only one of the variants should be considered. As the start distance order relation score $\delta_{\rightarrow a}$ yields the highest value for the gold standard alignments and has the strongest correlation to the micro f-measure, it was suggested as a means to improve label-based matching.

The order relation score was then used to design the Order Preserving Bag-of-Words Technique. The basic idea of OPBOT is to search the configuration space of BOT for a configuration that is estimated to have a high effectiveness. In particular, OPBOT applies six different BOT configurations to all activity pairs in a process model collection. These configurations differ with regard to the word similarity function and the pruning option. For each configuration it optimizes the threshold based on the similarity scores yielded by the configuration. That is, it iterates over a set of candidate values and proposes the value for which the highest order relation score is yielded as the optimal threshold for the configuration. Once the thresholds of the configurations were optimized, OPBOT combines the results of the two configurations with the highest order relation score and constructs a set of alignments that contains one alignment per model pair in the collection. By selecting the configurations which best reflect the characteristics of the model collection OPBOT achieves a domain adaptation of BOT. However, in contrast to the default configuration of BOT its application is limited to situations where an entire model collection can be analyzed.

The evaluation on all datasets verifies both: OPBOT's and the order relation score's validity. That is, it was shown that OPBOT achieves a high relative performance on the datasets that is close to the performance of the best BOT configuration. Additionally, it outperforms the default BOT configuration in most cases and also makes the manual provision of training data obsolete. Especially the results on the evaluation datasets substantiate the applicability of the score and OPBOT, because these datasets were not used for the development of both concepts. Although the results provide evidence that confirms Sub-hypothesis H3, the analysis revealed two limitations. First, the reduction of the search space can be too strict and can limit the effectiveness of OPBOT as better performing configurations are excluded from the search. However, the reduction is essential to balance the chance of detecting the best configuration with the chance to detect outliers. Second, the applicability of the order relation score $\pi_{\rightarrow a}$ is limited to alignments between model pairs that share some similarities, because the score is distorted if there are no similarities between the models. Nevertheless, the additional analysis of the score on the evaluation datasets confirmed the general validity of the score for

models that share similarities. In this regard, further evidence was given through the examination of the score in the context of matcher selection. Here, the score was used to rank the matchers from the second matching contest in 2015 [20] and it was shown that the top-ranked matchers yield a high performance in comparison to the best performing matcher from the contest. In summary, these evaluation and analysis results substantiated that control flow information is suited to improve the effectiveness of label-based matching techniques as postulated by Sub-hypothesis H3.

6

Learning From Expert Feedback

H4: The effectiveness of matching techniques is improved by the utilization of expert feedback.

From an abstract point of view, process model matching techniques constitute classifiers that evaluate information related to an activity pair in order to decide whether the activity pair corresponds or not. A first strategy to design such classifiers is referred to as *rote learning* [317]. This means that knowledge required for making decisions is statically implemented in the classifier. Many process model matchers including BOT rely on this strategy. That is, they comprise a set of predefined rules that are evaluated during runtime, e.g., to compute a similarity score based on the labels. Then, the outcome is used to classify the activity pair as corresponding or not. Yet, evidence from the analyses in the previous chapters as well as from related work suggests that such universal classifiers

yield a low effectiveness. With regard to BOT, this can be traced back to the assessment of word similarity that relies on universal similarity measures which do not necessarily reflect the domain characteristics of model collections (cf. Chapter 4). Another strategy is to design matchers in a way that they learn from observation which is also called *unsupervised learning* [317]. Such approaches evaluate the data they need to process and then automatically derive knowledge from it. In this regard, two strategies were examined in the previous chapters. First, some word similarities are based on word co-occurrences in the model collections or exploit semantic relations in a dictionary. But, these similarities yielded varying results (cf. Chapter 4). Second, OPBOT falls into this category, as it analyzes control flow information to learn which BOT configurations yield the best results. Although its effectiveness is generally higher and more stable than the effectiveness of the default BOT configuration, its effectiveness is still bound by the word similarities (cf. Chapter 5).

With that in mind, this chapter focuses on the confirmation of Sub-hypothesis H4 and examines the idea of *supervised learning* which characterizes approaches that derive knowledge from additional data provided by teachers or supervisors, respectively [318]. In particular, this chapter relies on the *interaction* with experts to collect feedback on automatically determined alignments. Algorithms that rely on interaction are generally seen as more powerful than rule-based algorithms [319, 320]. The advantage is that in addition to the use of universal rules and information from other knowledge sources matchers can also learn from experts who manually perform the task the matcher was designed for. This way the matcher has a baseline which it can use to adjust the decision making process in a way that it emulates the decision making process of the experts. Based on these considerations *feedback collection* is viewed as the manual process of correcting an automatically determined alignment. In particular, this chapter examines strategies to analyze such feedback to improve the effectiveness of BOT and OPBOT, respectively. To this end, the Adaptive Bag-of-Words Technique (ADBOT) is introduced and evaluated in order to give evidence towards Sub-hypothesis H4.

In the following, the specific approach to feedback collection pursued in this thesis is outlined in Section 6.1. Afterwards, two strategies to learn from the feedback are

presented. First, the adaptation of word similarities is explored in Section 6.2. Second, the transitivity of alignments as a means to automatically infer alignments from already known alignments is studied in Section 6.3. Based on the according analyses results, Section 6.4 introduces ADBOT. Subsequently, ADBOT is evaluated and contrasted to BOT, OPBOT, and the state-of-the-art matchers in Section 6.5. This section also deals with strategies to minimize the workload for experts while maximizing the effectiveness. Finally, Section 6.6 summarizes the findings to verify Sub-hypothesis H4.

6.1 The Process of Feedback Collection

In the context of process model matching Weidlich et al. [197] propose to derive a prediction model for the quality of matchers from a set of manually provided alignments. Yet, their work does not go beyond the introduction of a generic framework (cf. Section 3.3.3). In addition to this idea, there is a body of works in the field of schema and ontology matching that deals with the integration of experts into the matching process. Basically, these works can be assigned to one of three aspects: *the user interface*, *the process of feedback collection*, and *the analysis of feedback*. Regarding the design of user interfaces, guidelines to support experts in understanding and creating alignments were investigated in [321, 322]. Specific tools that assist users in creating alignments and applying matchers include amongst others COMA++ [323], PROMPT [324], and AMC [314], an overview is provided in [139]. The process of feedback collection was investigated by McCann et al. [325] who discuss different types of feedback ranging from the verification of attribute classifications over the analysis of domain constraints to the verification of correspondences. Additionally, [326] propose a generic model for representing feedback and examine a couple of challenges related to feedback collection including the identification of inconsistencies, the validation of feedback, and clustering of users. Jeffery et al. [327] introduce an approach to order correspondences for the validation by experts and to control the amount of feedback that is collected. Lastly, approaches that analyze feedback to improve the effectiveness include [328] where partial alignments, potentially provided by experts, are analyzed to reduce the search space. That is, additional correspondences

might only be identified in the contexts of correspondences from the partial alignment, or corresponding schema fragments are first derived from the partial alignment and then refined. Furthermore, Duan et al. [329] present an ontology matching technique that iteratively completes an alignment between two ontologies. Therefore, it uses correspondences provided by users in each iteration to adjust the weights of an aggregated similarity score. Moreover, Agreementmaker [330] incorporates the capability to tune algorithms based on gold standard alignments that are provided by experts.

This chapter builds upon these ideas and focuses on the analysis of feedback to improve the effectiveness of process model matching techniques. Hence, this chapter also abstracts from the particular user interface that is employed to collect feedback. In this regard, a basic assumption is that feedback is provided by experts when needed and that this feedback represents the objective ground truth. However, the strategies that are examined here depend on the specific process that is employed to collect feedback. The reason is that this process defines the type of feedback that is collected and thus determines what kind of additional information can be exploited.

In this regard, the author of this thesis in cooperation with other researchers introduced a framework for the design of feedback collection tasks in the context of process model matching in [25]. This framework is the result of a discussion on how to relate, combine, and slice aspects with regard to feedback collection. It provides guidance to systematically study feedback collection for process model matching and on an abstract level comprises three aspect groups that need to be considered when collecting feedback. While the *question* and the *answer* group comprise aspects that determine the information that is collected, the *answer quality* group focuses on measures to ensure high quality feedback. As this chapter abstracts from quality aspects, the latter group is ignored in the following. A detailed overview of the aspects in the question and the answer group is provided in Table 6.1.

Question group. This group defines which tasks an expert needs to carry out in order to provide feedback. It also comprises options to provide experts with additional information.

TABLE 6.1: Conceptual overview of design options for feedback collection tasks

Groups	*Aspects*	*Options*		
Question	Task description	Correspondence identification	Activity cluster identification	Activity annotation
	Representation	Whole process	Process fragment	Activity label
	Documentation	Additional		None
Answer	Modality	Fixed	Free	Combination
	Range	Binary	Numeric	Semantic
	Direction	Unidirectional		Bidirectional

The *task description* is the essential aspect in this group and defines what kind of feedback should be collected. Here, experts might be asked to *identify correspondences* or to *identify activity clusters*. While the former option can be used to yield (a sub-set of) the objective ground truth, the latter addresses the grouping of activities within a process model to derive candidates for complex correspondences. Alternatively, experts could be asked to *annotate activities* within a process model in order to yield a richer description of the activities. In this regard, experts might be provided with a set of harmonized labels, semantic annotations, or reference processes that serve as a basis for the annotation.

In contrast, the *representation* is related to the design of the user interface. It is used to control the complexity of the task and defines the context that is provided to the expert. Here, the *whole process models*, *fragments* of the models, or only the *activity labels* might be presented to the user.

Similar to the representation, the *documentation* provides the opportunity to support experts with *additional* information, such as a short explanation, process handbooks, or glossaries. In case such documentation is not available or is believed to unnecessarily increase the complexity of the task, *no* documentation might be provided.

Answer group. Whereas the question group refers to the task presentation, the answer group comprises options to specify the information collected from the experts.

The *modality* addresses the degree of freedom an expert has when providing answers. That is, a user might be restricted to provide answers from a *fixed* set of options. In

contrast, no options might be defined beforehand and the experts are able to provide *free* text answers. Additionally, it is possible to *combine* both variants.

Next, it might be of interest to collect detailed information regarding the relation between two activities or an activity and its annotation. The basic variant is to ask experts for a *binary* decision where a relation is confirmed to hold or not. In addition, a *numeric* degree might be used to characterize the extent to which a relation holds and provides an option to collect more fine-grained information. Lastly, the relations might be described *semantically*, e.g., by providing classes of relations like "unrelated", "subsumes", or "equal".

Furthermore, the *direction* of a relation might be restricted. In this regard, the relation might be expected to be *unidirectional* or *bidirectional*.

This framework and in particular the question and answer groups describe an extensive number of scenarios for feedback collection. Without considering specific ways to implement the options, the two groups already span a space of ($3^4 \times 2^2 =$) 324 different scenarios. As this number goes beyond the scope of this thesis, one specific scenario is chosen and examined in the following. In particular, the view by Bellahsene and Duchateau [269] is adopted here. Based on [28] they define the interaction between a matcher and a user in the context of schema matching as the confirmation or rejection of a relation between two elements [269]. This means that an automatically determined alignment is presented to experts and they are asked to confirm or to oppose classifications made by the matching technique. In this context, experts might find correspondences to hold (true positives) or to be falsely suggested (false positives). Similar, experts might identify additional correspondences that were overlooked by the matcher (false negatives) or confirm that activities do not correspond (true negatives).

With regard to the framework, this way of feedback collection can be characterized as follows. The task for experts is the correspondence identification. Although the user interface is not considered here, it is assumed that the experts are presented with the whole process models. The alignment proposed by the matching technique can be seen as additional information that is provided to the experts who are asked to make a binary decision, i.e., does an activity pair correspond or not. Accordingly, the modality is fixed and the identified correspondence relations are bidirectional.

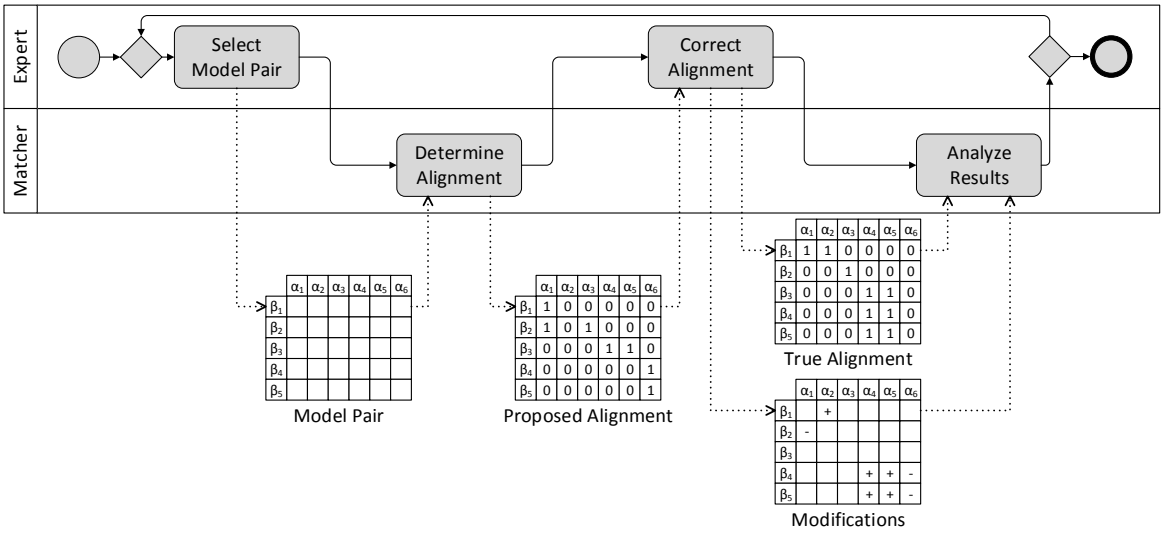

FIGURE 6.1: The process of feedback collection

The respective process of feedback collection is summarized in Figure 6.1. It is based on ideas from the semiautomatic schema matching process [139]. The rationale of the process is to iteratively match process model pairs from the model collection and to utilize knowledge gained during previous iterations. First, the expert selects a process model pair that should be matched. Then, the matcher automatically matches the model pair and proposes an alignment. In the next step the alignment is presented to the expert who subsequently corrects the proposed alignment. As explained above, the expert therefore adds missing correspondences and removes activity pairs that are not corresponding. The results of this step can then be analyzed by the matcher to adapt its matching process in order to achieve a better effectiveness in the subsequent iterations. To this end, the matcher might investigate the true alignment identified by the expert, or the modifications the expert carried out to transform the proposed alignment into the true alignment. Once an iteration is completed, a new iteration is triggered. The process stops when all process model pairs are aligned. However, this might be impractical in situations where the alignments between the process models are only an intermediate result that further analysis builds upon. In such situations, the feedback might be used to increase the effectiveness and might be collected until the effectiveness of the matcher reaches a certain level. Once this level is reached, all remaining model pairs are matched automatically without asking experts to manually correct the alignments.

In the following sections, the focus is on strategies to adjust the matching process by analyzing the results of the feedback collection. That is, strategies are examined that adapt word similarities and transitively infer correspondences from known alignments.

6.2 Word Similarity Adaptation

BOT and OPBOT primarily classify activity pairs as corresponding or not by applying the bag-of-words similarity $\sigma.\varpi$. At heart, it works by extracting the sets of individual words from the labels of the two activities that are compared. Subsequently, it computes a similarity score based on a word similarity function $\sigma.w$ for each pair of words that consists of a word from the first and a word from the second label. Once all pairs of words were compared, the maximum similarity score from the previous step is determined for each word in each label. The similarity score for the activity pair is then equal to the average of these maximum scores. If this score is higher than or equal to a threshold, the activities are considered similar and are suggested as correspondences. All activity pairs with a score lower than the threshold are classified as non-corresponding.

While the quality of the bag-of-words similarity is also impacted by the stemming and the pruning function, it effectively depends on the word similarity $\sigma.w$. On the one hand, it was shown in Chapter 4 that the effect of stemming and pruning is marginal concerning the effectiveness. On the other hand, in line with [225, 271, 272] it was argued that universal word similarity measures do not necessarily represent the domain characteristics of the model collections. However, developing measures that reflect the characteristics of a certain model collection is expensive and time consuming. Moreover, it needs to be repeated for each model collection. With that in mind, the following strategy aims to achieve such a domain adaptation for the word similarities. Note that in agreement with the argumentation, stemming and pruning are *not* considered and thus both features are *disabled* in all BOT configurations in the following.

Instead of requiring experts to design a word similarity measure, the strategy uses expert feedback to adjust the universal similarity measures. The rationale of the approach is to learn from misclassifications which were identified during the feedback collection,

i.e., the false positives and false negatives, and which were caused by the bag-of-words similarity. That is, it is assumed that in each iteration, the same BOT configuration is applied to identify the alignment. This alignment is then corrected by the experts and their modifications are subsequently used to adapt the word similarity measure that is part of the BOT configuration. As a consequence, later iterations benefit from the adjustments carried out in earlier iterations. With regard to the bag-of-words similarity, a false positive occurs, if the similarity score of an activity pair is higher than or equal to the threshold, but it actually does not correspond and was thus removed from the proposed alignment by the expert. Conversely, a false negative is an activity pair for which the similarity score is lower than the threshold, but it constitutes a correspondence and was added to the alignment by the expert. Accordingly, these misclassifications can be traced back to the word similarity $\sigma.w$. That is, for a false positive the average of the maximum word similarity scores was too high and for a false negative too low. In order to improve the assessment of the word similarities accordingly, the *word similarity adaptation algorithm* in Algorithm 6.1 is applied in each iteration to analyze the feedback.

The input of this algorithm is a binary relation $\mathscr{A}_{mc} \subseteq A \times A$ that represents the modifications carried out by the experts to transform the proposed alignment into the true alignment. That is, it contains the false positives and the false negatives.

Then, the algorithm consists of two coarse-grained steps. First, the word pairs (WP) for which the similarity score needs to be adapted and the specific correction values ($correct$) are determined (lines 3 to 15). The set of word pairs WP as well as the matrix $correct$ that stores the correction values are set up at the beginning of the algorithm (lines 1 and 2). Initially, the correction values are set to 0 for each possible word pair. Second, based on the results from the first step, the word similarity $\sigma.w$ is updated (lines 16 to 25). The reason for the separation of these two steps is that the determination of the word pairs for which the similarity needs to be adjusted depends on the old similarity values. As each activity pair is processed separately in this step, updating the similarity for a word pair distorts the detection of word pairs and correction values for other pairs.

The identification of word pairs based on the modifications works as follows. The algorithm iterates over each of the false positives and negatives (lines 3 to 15). In this

Algorithm 6.1: Word similarity adaptation algorithm

```
   Input: 𝒜_mc
1  WP = ∅;
2  correct = initialize();
3  foreach (a, a') ∈ 𝒜_mc do
4      if notCausedByFiltering(a, a') then
5          ϖ = tok(norm(λ(a)));
6          ϖ' = tok(norm(λ'(a')));
7          similarity = σ.ϖ(ϖ, ϖ');
8          foreach (w, w') ∈ maxWordPairs(ϖ, ϖ') do
9              if w ≠ w' then
10                 WP = WP ∪ (w, w');
11                 correct(w, w') = correct(w, w') + ϑ − similarity;
12             end
13         end
14     end
15 end
16 foreach (w, w') ∈ WP do
17     σ.w(w, w') = σ.w(w, w') + correct(w, w');
18     if σ.w(w, w') > 1 then
19         σ.w(w, w') = 1;
20     end
21     if σ.w(w, w') < 0 then
22         σ.w(w, w') = 0;
23     end
24     σ.w(w, w') = σ.w(w', w)
25 end
```

regard, it first examines whether the misclassification can be traced back to the bag-of-words similarity or not by applying the $notCausedByFiltering$ function (line 4). As BOT applies a filtering step in which correspondences are determined based on label equality and in which the set of potential correspondences is reduced accordingly, not all activity pairs are classified based on the bag-of-words similarity. Respectively, there can be false positives with equal labels, but which do not correspond according to the expert.

Moreover, there can be false negatives where one of the activities has an equally labeled counterpart in the other process model and thus the activity pair was considered to not correspond, but it was identified by the expert as a correspondence. If one of these two conditions applies the $notCausedByFiltering$ function returns $false$ and the activity pair is not processed, as the misclassification is not caused by the bag-of-words similarity.

For all other activity pairs, the algorithm determines the bag-of-words similarity (lines 5 to 7). Here, the word similarity $\sigma.w$ which was applied by the BOT configuration to previously propose the alignment is utilized. Next, the function $maxWordPairs$ is applied (line 8) to determine the set of word pairs that need to be adapted with regard to the current activity pair. For each word w in the union of the two bag-of-words it yields a word pair (w, w') consisting of the word w and the word w' from the other bag-of-words that yielded the maximum word similarity score for w. The respective set of word pairs thus comprises all word pairs that contributed to the misclassification of the activity pair. Note that the word similarity function is a symmetric function where for any word pair (w, w') it holds that $\sigma.w(w, w') = \sigma.w(w', w)$. Thus, in order to ensure a consistent management of word pairs at this point, each word pair (w, w') is arranged alphabetically, i.e., w occurs before w' in a dictionary. Then, the algorithm iterates over the determined set of word pairs (line 8 to 13). If the word pair consists of two different words (line 9) the word pair (w, w') is added to WP (line 10). Moreover, the correction value $corr(w, w')$ for this word pair is updated (line 11). Therefore, the algorithm subtracts the overall similarity score for the activity pair from the threshold and adds the respective difference to the stored correction value. The difference between the threshold and the overall similarity score characterizes the degree to which the similarity of the activities was misjudged. In case of a false positive the difference is negative and decreasing the word similarity by this difference for each of the determined word pairs will result in a similarity score that better reflects the similarity assessment of the expert. Analogously, the difference is positive for a false negative and increasing the word similarities respectively will lead to a higher overall similarity score. Note that if a word pair contributes to several misclassifications, its correction value is the sum of all differences yielded for the respective activity pairs.

Once the first step is finished, the word similarity values are updated by iterating over all word pairs in WP (lines 16 to 25). For each word pair the correction value is added to the word similarity score (line 17). Here, a word pair that predominantly contributed to false positives will have a negative correction value as its similarity was generally overestimated. Thus, its new word similarity score will be lower. In contrast, a word pair that predominantly contributed to false negatives will have a positive correction value. This indicates that its similarity was underestimated and its similarity score needs to be higher. In order to ensure that the word similarity $\sigma.w$ is bound to the interval $[0,1]$, the new word similarity value is modified, if the update leads to a value outside this interval (lines 18 to 23). That is, if the value is larger than 1, it is set to 1. Additionally, if it is smaller than 0, it is set to 0. Moreover, $\sigma.w$ is a symmetric function that yields the same similarity score for two words independent of the ordering of the words. Thus, the new similarity value for the word pair (w, w') is also assigned to the pair (w', w) (line 24).

To investigate the effect of the word similarity adaptation, the following experiment is carried out based on the development datasets. In the experiment different configurations of BOT are used to determine alignments for process models. Here, in all configurations stemming and pruning are *disabled*. In order to achieve a broad evaluation of the adaptation algorithm, the three word similarity measures that are part of OPBOT (LEV, LIN, and 2CS) as well as five different threshold values (.5, .6, .7, .8, and .9) are applied. Consequently, 15 different BOT configurations are used.

For each of the two datasets all process model pairs are matched following the process for feedback collection from Figure 6.1. In this regard, the gold standards are used to simulate the expert feedback and the process is executed for each of the BOT configurations separately. That is, in each iteration a model pair from the dataset is matched by the respective BOT configuration. The proposed alignment is then stored to compute the effectiveness once all process model pairs have been matched. Moreover, it is compared to the gold standard for the model pair and all misclassifications are determined. The falsely classified activity pairs are then passed to the word similarity adaptation algorithm to adjust the word similarity applied by the respective BOT configuration. In this regard, the threshold that was set for the BOT configuration will be used to determine the correction

value. Once the algorithm is done, the next iteration of the process is carried out. That is, the next model pair is processed by the BOT configuration using the updated word similarity. This way the word similarity is adjusted stepwise until the whole model collection is matched. Note that by storing the alignment in each iteration, it is ensured that the alignments which are used to compute the effectiveness at the end only depend on the adaptation that was achieved before the model pair was processed. All adaptations in later iterations do not impact the assessment of the effectiveness for the alignments.

A factor that influences the word similarity adaptation is the order in which the process model pairs are matched. That is because the order of model pairs determines the order of the similarity adapations. Thus, to examine the degree to which the ordering influences the adaptation, 100 orders were randomly generated for each of the model collections. Each of these random orders was processed by each of the 15 BOT configurations. Consequently, a total of 1,500 separate runs was carried out for each dataset.

As a first indicator for the effect of the word adaptation Table 6.2 shows the maximum micro f-measure that was observed for each dataset. That is, the table reports the best result that was observed for any of the 1,500 runs. Additionally, the best micro f-measure yielded by any of the 15 BOT configurations without the feedback collection is used as a baseline, i.e., when the word similarity was not adapted. The order of the model pairs is irrelevant for the latter case, i.e., a specific BOT configuration yields the same effectiveness for all orderings. As the table reveals, the adaptation can have a strong positive impact on the effectiveness. On BR the maximum micro f-measure based on the word similarity adaptation amounts to (.458 vs. .656 $\widehat{=}$) 143% of the micro f-measure for the best BOT configuration without the adaptation. The effect on UA is similar, i.e., .405 vs. .594 $\widehat{=}$

TABLE 6.2: Maximum effectiveness of BOT configurations with and without adaptation

Dataset	$\sigma.w$	ϑ	*adaptation*	pr_μ	re_μ	F_μ
BR	2CS	.8	not applied	.538	.399	.458
	LEV	.8	applied	.751	.582	.656
UA	LEV	.7	not applied	.597	.307	.405
	LEV	.8	applied	.646	.550	.594

146%. In both cases this is due to an increase in the precision and more important in the recall. With regard to the recall, the adaptation achieves a relative performance of (.399 vs. .582 ≙) 146% on BR and of (.307 vs. .550 ≙) 179% on UA.

While these results show the potential of the adaptation, they only consider the maximum effectiveness and thus draw an optimistic picture. To refine the analysis, Table 6.3 summarizes the improvements that were achieved for each of the 15 BOT configurations. Here, for each BOT configuration the micro f-measure that was yielded when the adaptation was not applied served as a baseline. In this context, the improvement for a BOT configuration gained in a certain run is the difference between the baseline and the micro f-measure that was achieved in this run. Accordingly, a positive value indicates that the adaptation of the word similarities improved the overall micro f-measure. As 100 runs were carried out for each of the BOT configurations, the table summarizes the improvements in terms of the maximum, the minimum, and the average improvement for each of the 15 configurations.

TABLE 6.3: Improvements of the micro f-measure

		Average			*Maximum*			*Minimum*		
Dataset	ϑ	*LEV*	*LIN*	*2CS*	*LEV*	*LIN*	*2CS*	*LEV*	*LIN*	*2CS*
BR	.5	.09	.13	.13	.12	.15	.15	.06	.09	.10
	.6	.17	.18	.14	.20	.20	.16	.14	.14	.12
	.7	.22	.21	.15	.26	.24	.17	.19	.17	.13
	.8	.25	.22	.15	.29	.25	.18	.21	.18	.12
	.9	**.26**	.22	.17	**.29**	.26	.19	**.22**	.18	.14
UA	.5	.11	.09	.17	.14	.11	.20	.08	.05	.15
	.6	.10	.14	.18	.14	.18	.21	.07	.10	.15
	.7	.13	.17	.18	.16	.21	.22	.09	.13	.15
	.8	.17	.19	.15	.20	.23	.19	.14	.15	.12
	.9	**.24**	.23	.15	.28	**.28**	.19	**.19**	.17	.12

A first interesting result is that for each BOT configuration the micro f-measures were always improved regardless of the order in which the model pairs were matched. That is because the minimum values are all positive. Yet, the actual impact of the adaptation varies. On BR the difference to the baseline varies between .06 and .29. The situation is similar on UA where the improvements fall into the interval $[.05, .28]$. To this end, the variance can partly be explained by the different ordering of the model pairs. Here, the average difference between the maximum and the minimum effectiveness per BOT configuration is .06 on BR and .07 on UA. However, the results also show that the improvement depends on the word similarity and the threshold. To better understand the impact of these two features, Figure 6.2 outlines the distribution of the effectiveness values yielded by all configurations with a certain threshold or a certain word similarity. That means the distribution for each of the five different threshold values is based on 300 runs per dataset, i.e., 100 different orderings of the model pairs per word similarity. In contrast, the distribution for each word similarity relies on 500 runs, i.e., 100 different orderings of the model pairs per threshold value.

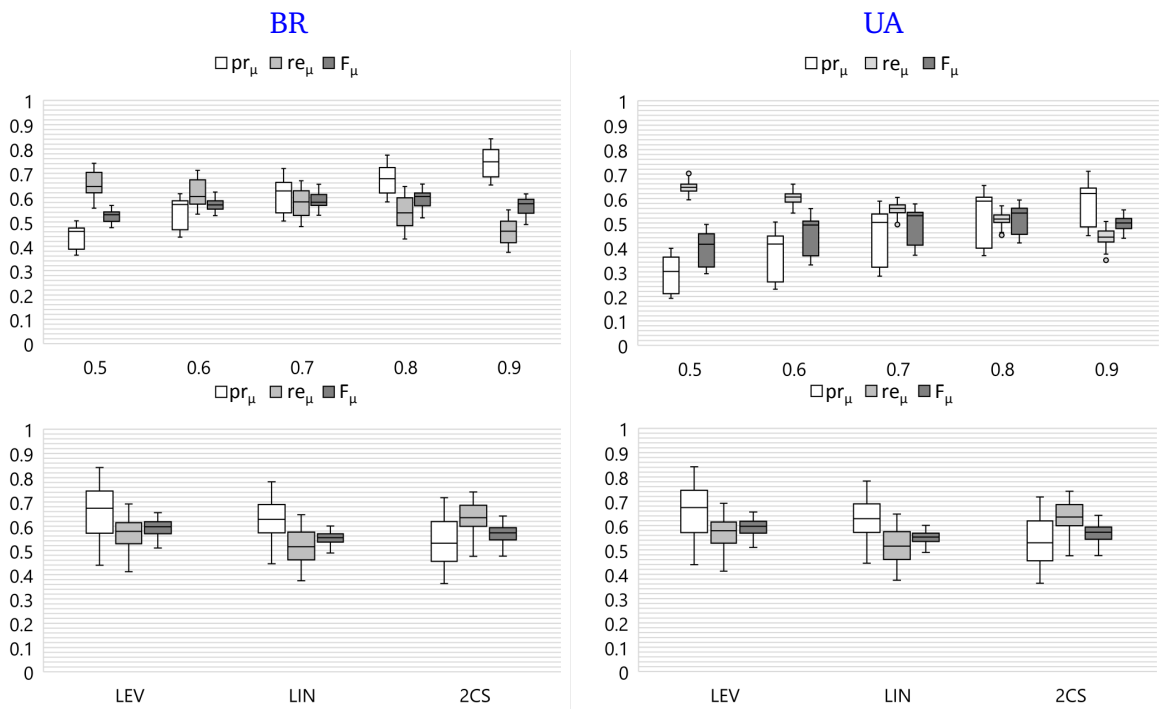

FIGURE 6.2: Overview of the effectiveness for the thresholds and word similarities

The figure shows that in each dataset the distributions of the micro f-measures are similar for all the three word similarities. However, there are differences with regard to the precision and recall values. That is, for LEV and LIN the precision takes higher values than the recall. For 2CS it is the other way around. In contrast to the word similarities the differences in the effectiveness are larger for the different threshold values. On both datasets an increase in the threshold is connected with an increase in the precision and a decrease in the recall. Moreover, the best micro f-measures are on average yielded for a threshold value of .7 or .8. This observation shows that the overall effectiveness achieved by adapting the word similarities depends on the quality of the respective BOT configuration.

While these analysis results address the overall effectiveness of the word similarity adaptation approach, the average f-measure yielded for each model pair with and without the adaptation are contrasted in Figure 6.3. As the figure reveals, the word adaptation achieves an improvement for the majority of the model pairs. On BR the f-measure is improved for 29 model pairs and on UA for 30. On both datasets there are more than 20 model pairs for which the f-measure with the adaptation is lower than .3 and can be lifted to approximately .5. However, the figure also shows that the micro f-measure is decreased for some of the model pairs and on average seems to be located at approximately .5 for all pairs. According to this result, model pairs for which a high effectiveness is already achieved without the similarity adaptation should be matched at the beginning when the effect of the word similarity adaptation is low and the high effectiveness can still be achieved.

In summary the results demonstrate that due to the positive impact on the effectiveness, the word similarity adaptation can be considered as a means to adjust the bag-of-words similarity in a way that it better reflects the domain characteristics of a certain model collection. Thus, the algorithm constitutes a lightweight supervised WSD approach that adjusts similarity values, but does not learn semantic relations between words. The results also revealed two problems that influence the improvements in the effectiveness. First, the effect can vary strongly depending on the specific ordering of the model pairs, the threshold and the word similarity. Second, the adaptation algorithm might lead to

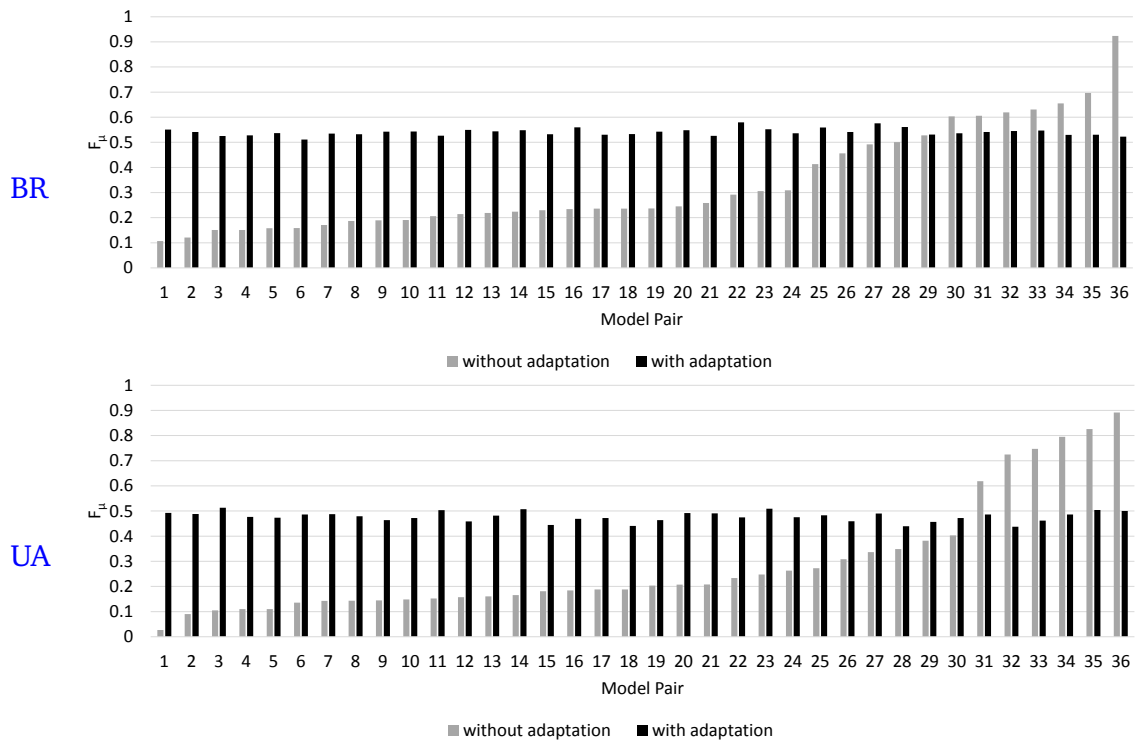

FIGURE 6.3: Average f-measure yielded per model pair with and without the adaptation

situations where the micro f-measure for a model pair is actually decreased compared to the effectiveness yielded without the adaptation. Thus, a matching technique that utilizes the word similarity adaptation algorithm should also incorporate strategies to mitigate these effects.

6.3 Transitivity

The second strategy to improve the effectiveness of matchers through expert feedback applies a well-known property in mathematics: *transitivity*. Generally speaking, transitivity can be interpreted in the following way: if two things are equal to the same thing, they are also equal to one another. In mathematical terms a binary relation $R \subseteq X \times X$ is transitive, if $\forall x_1, x_2, x_3 \in X : [(x_1, x_2) \in R \wedge (x_2, x_3) \in R] \Rightarrow (x_1, x_3) \in R$ [331].

Accordingly, the idea is here to decide whether an activity pair (a', a'') corresponds or not by analyzing the true alignments that were already discovered during feedback collection. In particular, the idea is to search these true alignments for an activity a

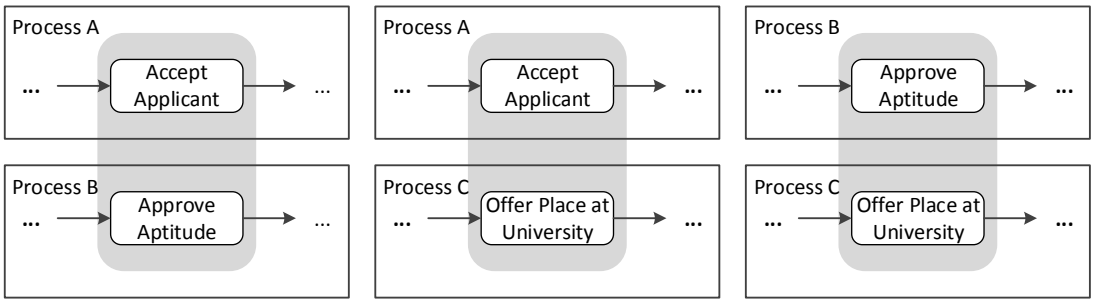

FIGURE 6.4: Example for transitive correspondences

that corresponds to a' as well as to a''. If such an activity a exists, it is considered as evidence towards the correspondence relation between (a', a''). An example of such transitive correspondences is shown in Figure 6.4. This example comprises three activities, "accept applicant" from process A, "approve aptitude" from process B, and "offer place at university" from process C. Because all three activities depict the task of determining, if an applicant is qualified for a certain course of study, each of the activities corresponds to both other activities. Accordingly, transitivity holds between these correspondences. Consequently, the alignments between process A and B as well as process A and C might be used to automatically infer the alignment between process B and C. Of course, any other constellation where two of the alignments are known is also conceivable.

In order to investigate to which degree transitivity exists in model collections, the gold standard alignments of the two development datasets are examined in the following. Moreover, the *global clustering coefficient* $\chi \in [0, 1]$ [332], also referred to as the graph transitivity index, is used as a means to measure the extent to which transitivity holds in the datasets. In graph theory, it provides information on the degree to which nodes tend to form clusters within a graph. It is also of interest for the analysis of social networks [333, 334] where it provides information on the existence of groups whose members share a certain relation, e.g., groups of friends.

The global clustering coefficient relies on a graph representation of the data where the nodes represent the elements and the edges are the relations between the elements. In the context of business process model matching this graph contains one node per activity in the model collection and the edges depict the correspondences that exist in the collection. From such a graph, the set of *triplets* is derived in order to compute the global clustering coefficient. Here, a triplet is a 3-tuple that consists of three distinct nodes from the graph.

Accordingly, an *activity triplet* contains three distinct activities from the model collection. Yet, as the goal is to examine how likely it is for three activities to transitively correspond, only those triplets that consist of activities from different process models are considered. The reason is that correspondences exist between different process models. Thus, triplets with activities from the same model contain at least two activities that do not correspond and can hence be ignored.

Definition 6.1 (Activity triplets) *Let* $\{P_i\}_{i=1}^{k}$ *with* $P_i = (N_i, A_i, E_i, \lambda_i, \tau_i)$ *be a collection of* $k \in \mathbb{N}_{\geq 2}$ *process models. Then, the set of activity triplets* A^3 *is defined as*

$$A^3 = \{(a, a', a'') | (a, a', a'') \in A_x \times A_y \times A_z \wedge 1 \leq x, y, z \leq k \wedge x \neq y \neq z \neq x\}$$

Given the set of activity triplets, the global clustering coefficient is defined as the ratio of the number of *transitive* activity triplets and the number of *potentially transitive* triplets. A transitive triplet is a triplet where each activity corresponds to both other activities. That is, a transitive triplet comprises three activities that correspond to each other. By contrast, potentially transitive triplets are all triplets for which at least one activity corresponds to both other activities. This means, a potentially transitive triplet satisfies the condition of the transitivity. Thus, the global clustering coefficient is the percentage of cases where the transitivity condition is fulfilled and transitivity actually holds. Consequently, if the global clustering coefficient is 1 all correspondences transitively inferred from the existence of two other correspondences truly exist. The lower the coefficient is the more often will a transitively inferred correspondence be incorrect, as the number of cases increases where transitivity is falsely concluded.

Definition 6.2 (Global clustering coefficient) *Let* $\{P_i\}_{i=1}^{k}$ *with* $P_i = (N_i, A_i, E_i, \lambda_i, \tau_i)$ *be a collection of* $k \in \mathbb{N}_{\geq 2}$ *process models and* A^3 *be the set of activity triplets. Further, let* $\{\mathscr{A}_j\}_{j=1}^{l}$ *be a set of* $l \in \mathbb{N}_{>2}$ *alignments where there is at most one alignment for a model pair, i.e.,* $\forall \mathscr{A}_{x,y} \in \{\mathscr{A}_j\} : x \neq y \Leftrightarrow \neg[dom(\mathscr{A}_x) = dom(\mathscr{A}_y) \wedge cod(\mathscr{A}_x) = cod(\mathscr{A}_y)] \wedge \neg[dom(\mathscr{A}_x) = cod(\mathscr{A}_y) \wedge cod(\mathscr{A}_x) = dom(\mathscr{A}_y)]$. *Lastly, let* $\mathscr{A}^* = \bigcup_{j=1}^{l} \mathscr{A}_j \cup \mathscr{A}_j^{-1}$ *denote the set of all correspondences where a correspondence relation between two activities* a *and* a'

is expressed by the two activity pairs (a, a') *and* (a', a)*. Then, the global clustering coefficient* χ *is defined as*

$$\chi = \frac{|\{(a,a',a'')|(a,a',a'') \in A^3 \wedge |\{(a,a'),(a,a''),(a',a'')\} \cap \mathscr{A}^*| = 3\}|}{|\{(a,a',a'')|(a,a',a'') \in A^3 \wedge |\{(a,a'),(a,a''),(a',a'')\} \cap \mathscr{A}^*| \geq 2\}|}$$

The values of the global clustering coefficient for the development datasets are presented in Table 6.4. On BR it is .479, meaning that not even half of the potentially transitive activity triplets are actually transitive. On UA it is even lower (.274) and only slightly more than one fourth of the potentially transitive activity triplets is transitive. These results suggest that transitivity does not hold within the datasets.

Nevertheless, a closer inspection of the results revealed two problems related to the global clustering coefficient. First, it does not correctly represent situations where transitivity includes complex and elementary correspondences. An example for this problem is shown in Figure 6.5. Here, there are three fragments of different process models which depict the task of making a decision whether to accept or to reject a student's application. While accepting and rejecting are distinct activities in process B and process C, process A only contains one general activity which subsumes the two activities. Consequently, the activities β_1, β_2 ,γ_1 and γ_2 correspond to activity α, but β_1 only corresponds to γ_1, and β_2 only to γ_2.

The graph representation of this example as well as of the respective transitive and potentially transitive activity triplets are depicted in Figure 6.6. The graph includes four potentially transitive activity triplets $(\alpha, \beta_1, \gamma_1)$, $(\alpha, \beta_1, \gamma_2)$, $(\alpha, \beta_2, \gamma_1)$ and $(\alpha, \beta_2, \gamma_2)$, but only $(\alpha, \beta_1, \gamma_1)$ and $(\alpha, \beta_2, \gamma_2)$ are transitive. Here, the global clustering coefficient χ yields a rather low value of 0.5.

TABLE 6.4: Results of the global clustering coefficient analysis on BR and UA

	Number of Triplets		
Dataset	*potentially transitive*	*transitive*	χ
BR	2686	1286	.479
UA	2770	760	.274

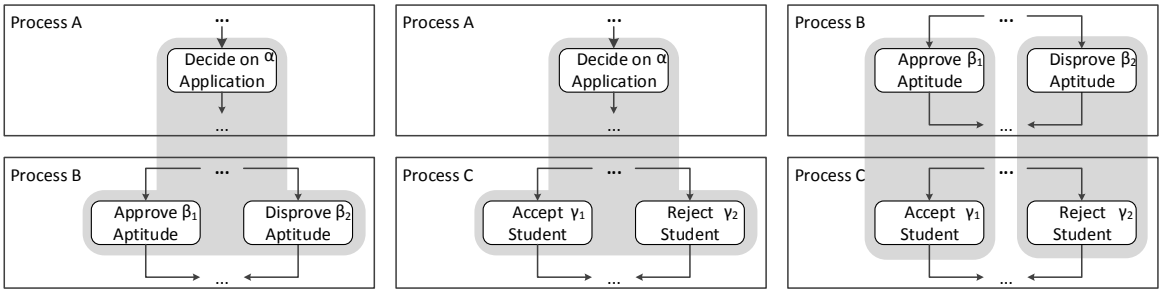

FIGURE 6.5: Example for transitive elementary and complex correspondences

The second problem related to the global clustering coefficient is that it does not necessarily reflect the observed effectiveness. Despite the low coefficient value in the example from Figures 6.5 and 6.6, transitively inferring activities might actually result in a high effectiveness depending on which alignment is inferred as outlined in Figure 6.7. In the example, there are three scenarios. In the first scenario, process B and C are matched based on the alignments between process A and B as well as between process A and C. Applying transitivity to detect correspondences results in the correspondences: (β_1, γ_1), (β_1, γ_2), (β_2, γ_1) and (β_2, γ_2). In this case, the recall is 1 and the precision 0.5 as the true correspondences (β_1, γ_1) and (β_2, γ_2) are found, but the non-corresponding activity pairs (β_1, γ_2) and (β_2, γ_1) are also suggested. In the second scenario, process A and C are matched and therefore the alignments between process A and B as well as between process B and C are used. Here, the two correspondences (α, γ_1) and (α, γ_2) are suggested. As these correspondences constitute the true alignment, the recall and the precision is 1. The same can be observed in the third scenario where process A and B are matched. This shows that the actual effectiveness yielded by transitively inferring correspondences can be different from what the global clustering coefficient suggests.

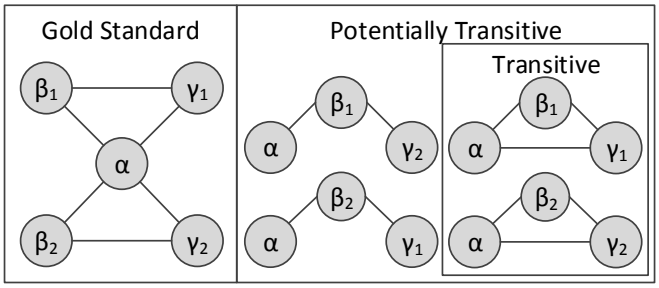

FIGURE 6.6: Potentially transitive activity triplets for the example

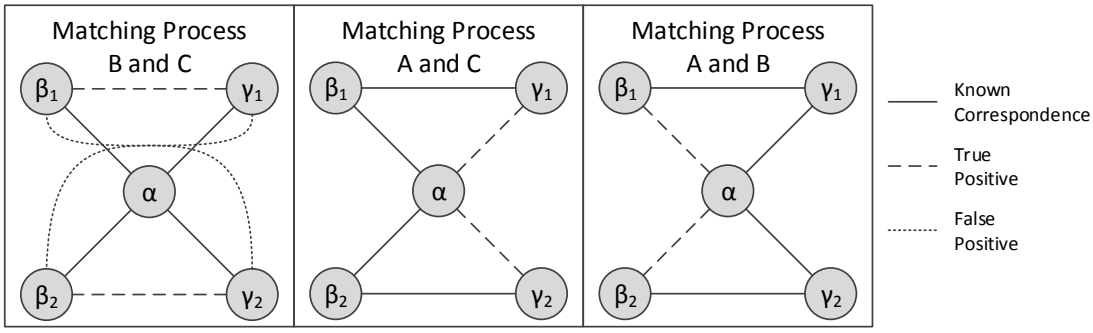

FIGURE 6.7: Possible matching scenarios in the example

As a consequence of these shortcomings, the analysis is refined by calculating the *local clustering coefficient* $\overline{\chi} \in [0,1]$ [335]. In contrast to the global clustering coefficient, the local cluster coefficient separately computes a score $\chi(a) \in [0,1]$ for each activity. In this regard, it does not rely on all potentially transitive activity triplets the activity is part of. Instead, for a given activity a it only considers those triplets (a, a', a'') where a corresponds to a' *and* to a''. This means, the local clustering coefficient focuses on the scenarios where a is the evidence for a correspondence relation between a' and a'' and it measures how often this evidence leads to the identification of a correspondence that exists. Consequently, the higher the score $\chi(a)$ is, the more reliable it is to use correspondences that contain a to transitively infer correspondences. The local clustering coefficient is the average of the activity coefficients yielded for all activities in the model collection. However, for an activity there might not be any triplet where the activity corresponds to both of the other two activities. Here, the determination of $\chi(a)$ would require a division by 0. Thus, the adapted version by [336] is applied. That is, the activity clustering coefficient $\chi(a)$ is set to a value of 0, if no activity triplets were determined. Further, the average of the coefficients is corrected based on the ratio of all such activities.

Definition 6.3 (Local clustering coefficient) *Let* $\{P_i\}_{i=1}^{k}$ *with* $P_i = (N_i, A_i, E_i, \lambda_i, \tau_i)$ *be a collection of* $k \in \mathbb{N}_{\geq 2}$ *process models. Further, let* $A^* = \bigcup_{i=1}^{k} A_i$ *denote the set of all activities and* A^3 *be the set of activity triplets in the model collection. Moreover, let* $\{\mathscr{A}_j\}_{j=1}^{l}$ *be a set of* $l \in \mathbb{N}_{>2}$ *alignments where there is at most one alignment for a model pair, i.e.,* $\forall \mathscr{A}_{x,y} \in \{\mathscr{A}_j\} : x \neq y \Leftrightarrow \neg[dom(\mathscr{A}_x) = dom(\mathscr{A}_y) \wedge cod(\mathscr{A}_x) = cod(\mathscr{A}_y)] \wedge \neg[dom(\mathscr{A}_x) = cod(\mathscr{A}_y) \wedge cod(\mathscr{A}_x) = dom(\mathscr{A}_y)]$. *Lastly, let* $\mathscr{A}^* = \bigcup_{j=1}^{l} \mathscr{A}_j \cup \mathscr{A}_j^{-1}$ *denote the set of all*

TABLE 6.5: The local clustering coefficients on BR and UA

Dataset	$\overline{\chi}$
BR	.842
UA	.745

correspondences where a correspondence relation between two activities a and a' is expressed by the two activity pairs (a, a') and (a', a). Based on the activity clustering coefficient

$$\chi(a) := \begin{cases} 0 & |\{a' | (a, a') \in \mathscr{A}^*\}| \leq 2 \\ \frac{|\{(a,a',a'') | (a,a',a'') \in A^3 \wedge \{(a,a'),(a,a''),(a',a'')\} \subseteq \mathscr{A}^*\}|}{|\{(a,a',a'') | (a,a',a'') \in A^3 \wedge \{(a,a'),(a,a'')\} \subseteq \mathscr{A}^*|} & else \end{cases}$$

the local clustering coefficient is defined as

$$\overline{\chi} = \left(1 - \frac{|\{a | a \in A^* \wedge |\{a' | (a, a') \in \mathscr{A}^*\}| \leq 2\}|}{|A^*|}\right)^{-1} \cdot \frac{1}{|A^*|} \sum_{a \in A^*} \chi(a)$$

Table 6.5 reports the local clustering coefficients for both development datasets. While the coefficient is .842 on BR, it is .745 on the UA. These high values show that the reliability of transitively inferring correspondence is high, but there are exceptions in which a correspondence might be falsely proposed, as e.g., in the case of complex correspondences. Overall, the results suggest that transitivity is a suitable strategy to discover correspondences. Further evidence in this regard is given by the evaluation of ADBOT which incorporates transitivity and is introduced in the next section.

6.4 The Adaptive Bag-of-Words Technique

The Adaptive Bag-of-Words Technique (ADBOT) relies on the word similarity adaptation algorithm and transitivity. At heart, ADBOT's design follows the process of feedback collection from Section 6.1 as outlined in Figure 6.8. In addition to the abstract process, ADBOT initially *prepares BOT configurations*. That is, following OPBOT's matching process ADBOT analyzes the model collection in order to configure three BOT configurations. The decision to rely on three configurations is motivated by the observation that the overall effectiveness achieved by adapting the word similarities is also determined by the quality

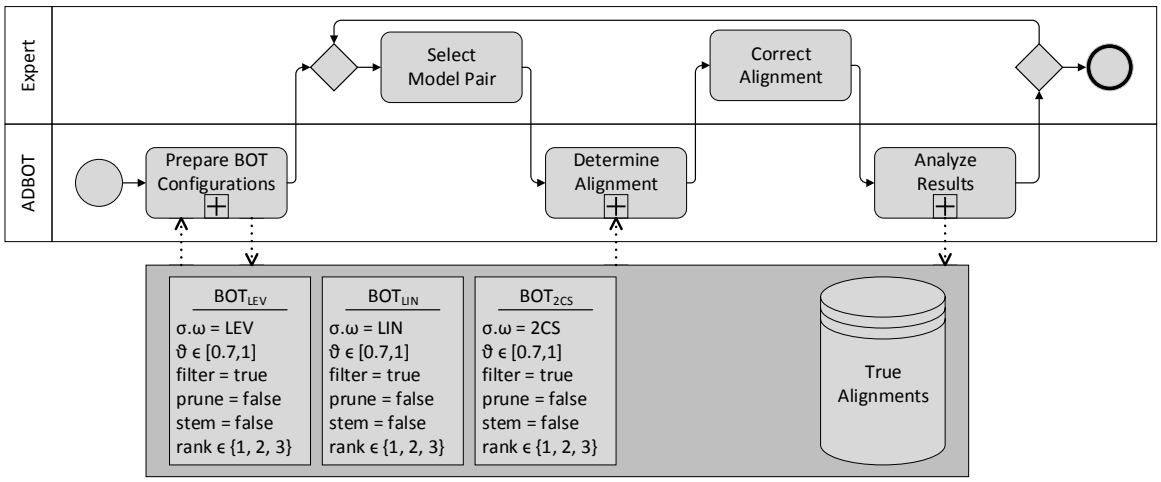

FIGURE 6.8: The ADBOT workflow

of the BOT configuration, i.e., by the threshold and the word similarity (cf. Section 6.2). Thus, three configurations are used here to increase the chance of yielding a strong improvement. Moreover, the idea is to achieve a high quality in early iterations when only a small amount of feedback has been analyzed and the domain adaptation is low. In each iteration, ADBOT relies on the BOT configurations and on the true alignments that were already discovered in order to *determine the alignment* for the model pair selected by the expert. Finally, ADBOT *analyzes the results* from the manual correction of the proposed alignment. Here, it uses the word similarity adaptation algorithm to adjust the BOT configurations. Moreover, it stores the true alignment to establish a knowledge base that can be exploited to transitively infer alignments. In the following each of the three steps is explained in more detail.

Prepare BOT Configurations (Figure 6.9). As outlined above, three BOT configurations are used in order to increase the chance of yielding a high effectiveness. In this regard, each BOT configuration applies filtering and discards stemming as well as pruning. Moreover, the configurations utilize different word similarities. To this end, there is one configuration for each of the three similarities that OPBOT uses (LEV, LIN, and 2CS). To prepare these configurations, OPBOT's search strategy is reused.

In this regard, ADBOT first *extracts the activity pairs* from the model collection. Then, for each of the BOT configurations it *computes the similarity scores* for all pairs. Based on the similarity values it *determines the threshold* for which the highest order relation

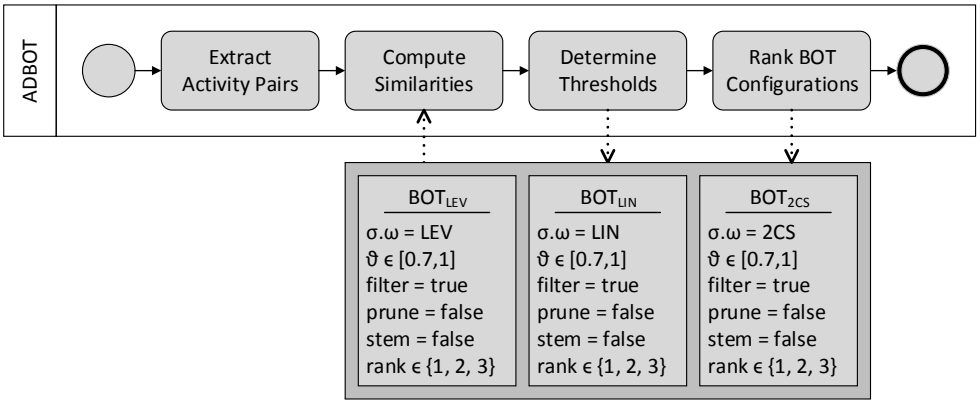

FIGURE 6.9: The preparation sub-workflow

score is yielded. Here, for all configurations all distinct similarity values that are equal to or higher than .7 are considered as possible threshold values. Note that uniformly considering .7 as the minimum threshold is based on the observation that the micro f-measures resulting from the word similarity adaptation tend to be highest for those threshold values (cf. Section 6.2). Finally, ADBOT *ranks the BOT configurations* according to their order relation score. Here, a rank of 1 is assigned to the best configuration with regard to the order relation score. Note that in contrast to OPBOT the configurations are ranked at this stage, but alignments are not proposed.

Determine Alignment (Figure 6.10). ADBOT's matching strategy considers correspondences that are transitively inferred to be more reliable than those that are determined based on BOT configurations. As a consequence, the first step is to *check the transitivity*. That is, for the process model pair (P', P'') that needs to be matched, the number of process models P is determined for which the true alignments between P and P' as well as between P and P'' are known. In case there is at least one such process model, ADBOT *uses transitivity* to match the process models. In this regard, ADBOT classifies an activity pair (a', a'') as corresponding, if in the determined models from the previous step there is at least one activity a which corresponds to a' and to a''. If no models were found in the first step, the process models are matched by the BOT configuration with the highest rank, i.e., which is predicted to yield the best effectiveness.

Analyze Results (Figure 6.11). The last of the three steps in ADBOT is carried out to learn from the feedback that the experts provided. To this end, the technique first *stores*

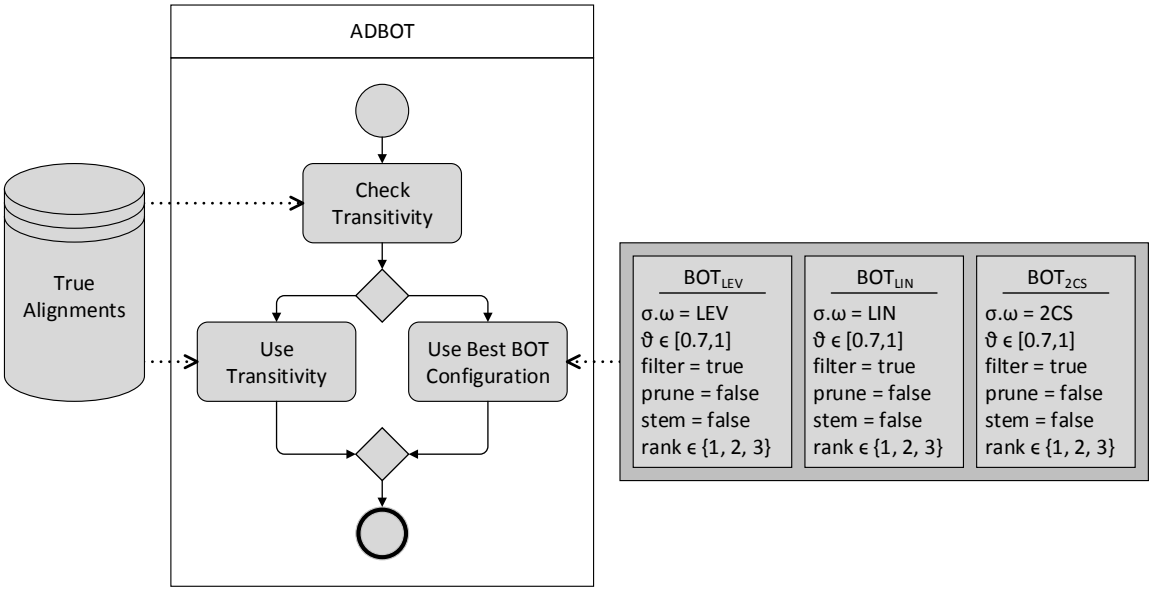

FIGURE 6.10: The determination sub-workflow

the alignment determined by the experts in order to transitively infer alignments from it in subsequent iterations. Then, it *adapts the word similarities* of each of the three BOT configurations. If a BOT configuration was not used in the current iteration to determine the proposed alignment, the configuration is applied to the process model pair and its result is compared to the alignment defined by the expert. The derived modifications are then used to carry out the adaption. In addition to this step, the last two steps aim to further improve the matching based on the BOT configurations. In particular, they aim to refine the automatic configuration based on the order relation score from the preparation workflow. First, the true alignments that have been discovered so far are used to *determine the thresholds*. This means, the true alignments are used as a baseline to assess the effectiveness of the BOT configurations. Here, for each of the three configurations all observed similarity values larger than .7 are considered as possible new threshold values and the threshold value for which the highest micro f-measure is yielded is selected as the new threshold. Then, the second step compares the determined effectiveness of the three resulting configurations in order to *rank the BOT configurations*. In this regard, the configuration with the highest micro f-measure is now ranked first. Note that in the last two steps empty alignments, i.e., those that do not contain any correspondences, are not considered. The reason is that regardless of the word similarity for such alignments the

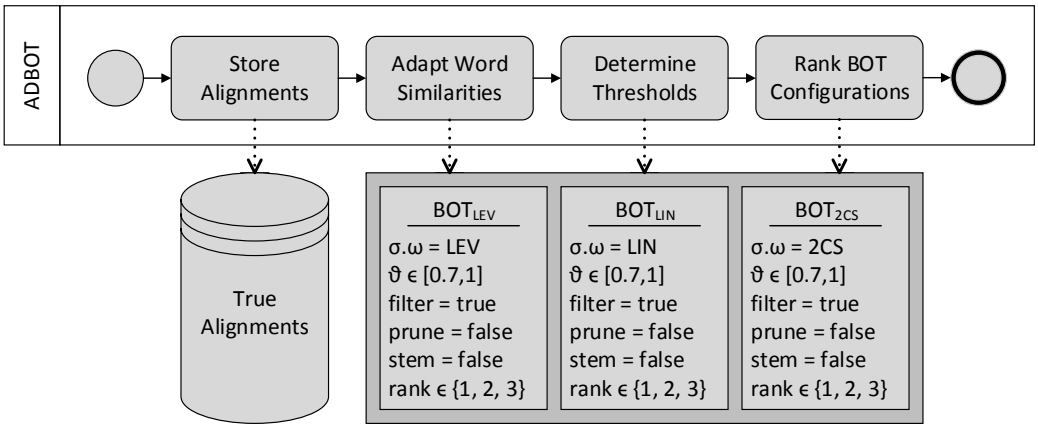

FIGURE 6.11: The analysis sub-workflow

best threshold is 1, as no other threshold value leads to the identification of more non-corresponding activity pairs than this value. However, for other alignments a threshold of 1 is typically too restrict and thus empty alignments might distort the re-configuration, especially in early iterations. As a consequence, the first re-configuration is carried out when the first non-empty true alignment was discovered.

6.5 Evaluation and Analysis

This section assesses the effectiveness of ADBOT in order to give further evidence that expert feedback can be exploited to improve matching techniques. In this regard, the effectiveness is separately studied on the development and the evaluation datasets. Additionally, various analyses refine the insights from the effectiveness evaluation. First, strategies to sort model pairs in order to maximize ADBOT's effectiveness are studied. Second, the reduction of expert workload is investigated. Third, transitivity is examined with regard to the evaluation datasets in order to give evidence towards its general validity. Last, relations between model collection characteristics and the improvements achieved through the analysis of expert feedback are investigated in order to better understand the limitations of the feedback analysis.

6.5.1 Effectiveness on the Development Datasets

So far, this chapter has independently studied the word similarity adaptation algorithm and transitivity on the development datasets. In this regard, it was shown that the word similarity adaptation algorithm has a positive impact on the effectiveness and that transitivity is a reliable means to infer correspondences. ADBOT incorporates both strategies as well as a continuous re-configuration of the BOT configurations inspired by OPBOT's search strategy. In order to examine the combination of these strategies, ADBOT is evaluated with regard to the development datasets. Like the word similarity adaptation algorithm, ADBOT's effectiveness depends on the order in which the model pairs are matched. That is, the order impacts the word similarity adaptation, determines the knowledge base used to transitively infer correspondences and influences the re-configuration of the BOT configurations . Thus, the 100 random orders from the analysis in Section 6.2 are reused here. To this end, for each development dataset the results for the run that yielded the minimum and that yielded the maximum micro f-measure are considered. Furthermore, the average micro and macro level effectiveness measures are reported. Additionally, the maximum micro f-measure for the word similarity adaptation algorithm (cf. Table 6.2) serves as a baseline to investigate whether the re-configuration and transitivity further improve the effectiveness. Moreover, ADBOT is contrasted to BOT's default (BOT_{ALL}) and optimal configuration (BOT_{MAX}), to OPBOT as well as to the best performing matchers from the contests [19, 20].

As shown in Table 6.6 the maximum and average micro f-measures for ADBOT are higher than the maximum micro f-measure for the word similarity adaptation algorithm (.675, .743 vs. .656) on BR. This result shows that the integration of the transitivity and the re-configuration of the BOT configurations can yield further improvements. Here, the improvements are due to an overall increase in the recall. On average it is .777 and the recall for the minimum and maximum micro f-measure differ only slightly. In contrast, the order in which the model pairs are matched, impacts the precision which on average is .598 and its absolute difference to the minimum and the maximum runs is approximately .1. In comparison to BOT and OPBOT as well as to the two state-of-the-art

TABLE 6.6: Effectiveness of ADBOT and other matchers on BR

Approach	pr_μ	re_μ	F_μ	pr_M	re_M	F_M
ADBOT (Min)	.496	.771	.603	.547	.790	.606
ADBOT (Avg)	.598	.777	.675	.655	.776	.667
ADBOT (Max)	.701	**.791**	**.743**	.701	**.791**	**.708**
Word Similarity Adaptation (Max)	**.751**	.582	.656	**.742**	.549	.584
OPBOT	.613	.452	.520	.583	.469	.499
BOT_{ALL}	.657	.344	.452	.615	.329	.382
BOT_{MAX}	.652	.452	.534	.633	.467	.511
RMM/NSCM	-	-	-	.68	.33	.45
pPalm-DS	.502	.422	.459	.499	.429	.426

matchers ADBOT clearly improves the effectiveness. All of these four matchers yield a micro f-measure that is lower than ADBOT's minimum micro f-measure. Here, BOT_{MAX} comes closest by yielding a relative performance of (.534 vs. .603 ≙) 88.6% with regard to the minimum f-measure of ADBOT and (.534 vs. .743 ≙) 71.9% with regard to the maximum. The micro recall of the other matcher ranges from (.344 vs .777 ≙) 44.3% to (.452 vs .777 ≙) 58.2% with regard to ADBOT's average recall. For RMM/NSCM the macro recall is (.33 vs .776 ≙) 42.5% of ADBOT's average macro recall. On average the precision of ADBOT is similar to those of the five matchers.

On UA the maxmimum micro f-measure of ADBOT is virtually equal to the maximum of the word similarity adaptation (.596 vs. .594). However, ADBOT improves the micro recall (.685 vs. .550) while it sacrifices precision (.527 vs. .646). Similar to BR, the micro recalls of the minimum and maximum runs differ only slightly and the average is .677. On the contrary, the precision varies strongly. This confirms the observation from BR that ADBOT improves the recall, but its precision depends on the order in which the model pairs are matched. While even the minimum micro f-measure improves the micro f-measure of BOT_{ALL} (.428 vs. .403), OPBOT and BOT_{MAX} might indeed yield a higher micro f-measure (.428 vs. .442). Yet, on average ADBOT outperforms both matchers (.496

TABLE 6.7: Effectiveness of ADBOT and other matchers on UA

Approach	pr_μ	re_μ	F_μ	pr_M	re_M	F_M
ADBOT (Min)	.315	.667	.428	.411	**.703**	.487
ADBOT (Avg)	.393	.677	.496	.493	.685	.526
ADBOT (Max)	.527	**.685**	**.596**	.569	.675	**.581**
Word Similarity Adaptation (Max)	**.646**	.550	.594	**.667**	.540	.558
OPBOT	.598	.350	.442	.578	.357	.412
BOT_{ALL}	.429	.380	.403	.455	.386	.382
BOT_{MAX}	.406	.486	.442	.443	.511	.453
RMM/NSCM	-	-	-	.37	.39	.38

vs. .442). This can again be traced back to the improvement in the micro recall. Here, OPBOT achieves (.35 vs .677 ≙) 51.7% of ADBOT's average micro recall and BOT_{MAX} (.486 vs .677 ≙) 65.1%. Additionally, RMM/NSCM's macro level effectiveness is lower than ADBOT's, as on average ADBOT yields a higher precision, recall, and f-measure. With regard to the macro f-measure RMM/NSCM only achieves (.38 vs .526 ≙) 72.2%.

In summary, the results show that the analysis of expert feedback can strongly improve the effectiveness of matching techniques. Moreover, the inclusion of transitivity and re-configuration can further increase the effectiveness of the word similarity adaptation as shown on BR. However, the magnitude of the improvement is bound by the ordering of the model pairs. ADBOT's effectiveness is typically higher than that of fully automated techniques from related work as well as from this thesis due to a huge increase in the recall. Yet, depending on the model pair ordering the precision of ADBOT might drop to a level at which the overall effectiveness of an automated technique is higher due to its higher precision. To examine the general validity of these findings, the next section repeats the analysis on the evaluation datasets.

6.5.2 Effectiveness on the Evaluation Datasets

In addition to the analysis of the development datasets this section assesses ADBOT's effectiveness with regard to the evaluation datasets. Similar to the previous analyses 100 random orderings of the model pairs were generated for each dataset. To characterize the effectiveness of ADBOT the runs that yielded the minimum and maximum micro f-measure as well as the average of the micro precision, recall, and f-measure are reported. Moreover, BOT_{ALL}, BOT_{MAX}, and OPBOT serve together with the best matcher from the second contest [20] for the SR dataset as a baseline. The respective effectiveness values for both datasets are presented in Table 6.8. Note that the word similarity adaptation was not evaluated separately, as the focus is on providing evidence towards ADBOT's effectiveness.

On SR the improvements that ADBOT achieves are low. That is, compared to BOT_{ALL}, BOT_{MAX}, and ADBOT the maximum micro f-measure is slightly higher (.625, .692, .658 vs. .711). On average ADBOT's micro f-measure is lower than that of BOT_{ALL} and BOT_{OPT} (.692, .658 vs. .654). Moreover, the minimum micro f-measure is lower than that of the fully automated techniques (.595). Overall, ADBOT's recall is generally similar to that of BOT_{ALL} and BOT_{OPT}, but the precision varies greatly. On this dataset AML-PM can be outperformed by the maximum micro f-measure (.68 vs. .711), but AML-PM generally seems to perform slightly better because the average micro f-measure of ADBOT is lower than the micro f-measure of ADBOT (.68 vs. .654). The reason for the marginal and sometimes even negative improvements on SR is that in this dataset each process model is matched only once and transitivity can thus not be exploited. Moreover, the process models originate from different business areas and thus the vocabulary is more diverse than in the other datasets. Consequently, the impact of the word similarity adaptation is low too. This shows that the improvement through feedback can only be exploited, if the obtained knowledge can actually be reused. A more detailed discussion of this problem is presented at the end of this section.

TABLE 6.8: Effectiveness of ADBOT and other matchers on SR and AW

	SR			*AW*		
Approach	pr_μ	re_μ	F_μ	pr_μ	re_μ	F_μ
ADBOT (Min)	.595	.595	.595	.877	.707	.783
ADBOT (Avg)	.797	.563	.654	.855	.840	.847
ADBOT (Max)	.854	.608	**.711**	.908	**.891**	**.899**
OPBOT	.599	**.653**	.625	.730	.339	.463
BOT_{ALL}	.774	.572	.658	**.959**	.251	.397
BOT_{MAX}	**.887**	.568	.692	.616	.552	.582
AML-PM	.786	.595	.677	-	-	-

On AW the effectiveness of ADBOT is drastically higher than this of BOT_{ALL}, BOT_{MAX}, and OPBOT. Here, BOT_{ALL} achieves (.397 vs .847 $\widehat{=}$) 46.9% of ADBOT's average micro f-measure, BOT_{MAX} (.582 vs .847 $\widehat{=}$) 68.7%, and OPBOT (.463 vs .847 $\widehat{=}$) 54.7%. Moreover, ADBOT's maximum f-measure reaches a value of .899. While the precision ranges in between that of OPBOT and BOT, the recall is strongly improved. Compared to the average micro recall, BOT_{ALL} yields only (.251 vs .840 $\widehat{=}$) 29.9%, BOT_{MAX} (.552 vs .840 $\widehat{=}$) 65.7%, and OPBOT (.339 vs .840 $\widehat{=}$) 40.4%.

Overall, the evaluation on the development datasets further confirms that expert feedback is a suitable means to improve the effectiveness of matching techniques. However, the results also show that the improvements differ depending on the order in which the model pairs are matched. This is especially a problem on the SR dataset where the improvements are rather small and might even be negative.

6.5.3 Maximization of the Effectiveness Improvements

According to the evaluation results from the previous sections, the order of the model pairs impacts the extent of the improvements and the overall effectiveness. Thus, it is essential to find a way to order model pairs such that the improvements are maximized. With that in mind, three strategies are examined to order model pairs.

The first strategy is referred to as the *equal labels* ordering. It is inspired by the observation that the word similarity adaptation lifts the effectiveness for most of the model pairs. Yet, there are a few exceptions where effectiveness is sacrificed (Section 6.2). Accordingly, the idea is to order the pairs in a way that the model pairs for which the effectiveness is generally high are matched at an early stage where the adaptation is low and the effectiveness for these model pairs is still high. Here, the number of equally labeled activity pairs is used as an indicator for the effectiveness. The rationale is that if there are many equally labeled activity pairs within a model pair, the effectiveness achieved by BOT is estimated to be high. To this end, for each model pair the number of equally labeled activity pairs is determined and normalized by the minimum number of activities in these two models. With regard to this indicator the model pairs are then sorted in descending order.

The second strategy builds upon the first one and aims to additionally boost the use of transitivity. Thus, it is referred to as the *transitivity* ordering. To obtain an ordering based on this strategy, the model pairs are sorted using the equal labels ordering first. From this ordering the top ranked model pair is removed and added as the first model pair to the transitivity ordering. Then, the next step is to choose one of the two models in this pair in order to match it with the remaining models. This way a set of alignments is established that can be used to transitively infer alignments between the remaining model pairs. Hence, for each of the two models in the selected pair the remaining model pairs in the equal label ordering that contain the model are selected. For the two resulting sets of model pairs the maximum position in the equal label ordering is determined. Finally, all model pairs in the set of model pairs with the smaller maximum position are removed from the equal labels ordering and added to the transitivity ordering in the same order they initially occurred in the equal label ordering. The set of model pairs with the smaller maximum position is chosen, because it is estimated to yield the higher effectiveness. That is because a smaller position corresponds to a higher equal labels indicator. Once the model pairs were added, the top ranked model pair from the remaining pairs in the equal label ordering is chosen and the same procedure is applied. This step is repeated as long as the equal label ordering contains model pairs.

Whereas the first two strategies result in a static ordering which is determined independent of the matching results, the third strategy is dynamic. It is based on the order relation score $\delta_{\rightarrow a}$ and hence called the *order relation* ordering. Like the transitivity ordering it is based on the equal labels ordering. At the beginning, it selects the top ranked model pair in the equal labels ordering and completes the first iteration of the process of feedback collection. That is, the model pair is matched, the alignment is corrected by the experts, and the results are analyzed. After this iteration the alignments for the remaining model pairs are computed and the model pair is chosen for which the alignment yields the highest order relation score. The rationale is that the order relation score is an indicator for the effectiveness and that the alignment with the highest score is likely to yield the highest effectiveness. If there are several model pairs for which the highest order relation score is yielded, the one with the smallest position in the equal labels ordering is selected. Then, the respective alignment is proposed to the expert and the next iteration of feedback collection is triggered. After each iteration the alignments for the remaining model pairs are re-calculated and the one with the highest order relation score is proposed to the expert.

To assess the ordering strategies, for each dataset ADBOT matched the model pairs in the respective orders. Based on the results the effectiveness was determined for each combination of the datasets and ordering strategies. Moreover, the runs with the maximum and the minimum micro f-measure from the 100 random runs serve as a baseline to examine the degree to which the strategies maximize ADBOT's effectiveness. Table 6.9 summarizes the results.

On all datasets the equal labels ordering yields very high micro f-measures compared to the maximum micro f-measures from the random orderings. Whereas on BR and UA it even outperforms the maximum, on SR and AW it yields a lower effectiveness which, however, is close to the maximum. Here, the most notable result is yielded on UA where ADBOT now achieves a relative micro f-measure of (.60 vs .68 $\widehat{=}$) 113.3% compared to the maximum random run. The transitivity ordering results in the highest effectiveness on BR. Yet, compared to the equal labels ordering it performs worse on UA and AW. Moreover, on SR it results in the same effectiveness as on this dataset each model is only aligned

TABLE 6.9: Comparison of strategies for the ordering of model pairs

	BR			*UA*			*SR*			*AW*		
Ordering	pr_μ	re_μ	F_μ	pr_μ	re_μ	F_μ	pr_μ	re_μ	F_μ	pr_μ	re_μ	F_μ
Random (Min)	.50	.77	.60	.32	.67	.43	.60	.60	.60	.88	.71	.78
Random (Max)	.70	.79	.74	.53	.69	.60	.85	**.61**	**.71**	.91	**.89**	**.90**
Equal Labels	.74	**.81**	**.78**	**.64**	**.71**	**.68**	**.90**	.55	.68	**.93**	.86	.89
Transitivity	**.78**	.79	**.78**	.53	.65	.58	**.90**	.55	.68	.90	.84	.87
Order Relation	.61	.73	.66	.46	.69	.55	.89	.55	.68	.88	.86	.87

once and the transitivity ordering does not change the equal labels ordering here. Lastly, the ordering based on the order relation score yields micro f-measures that are higher than the minimum f-measures from the random runs. But, it also results in the lowest effectiveness of all three strategies on BR, UA, and AW. On SR the micro f-measure is equal to the other two strategies, at a slightly higher recall and a marginally lower precision.

Overall, the results suggest that the three strategies can be used to maximize the effectiveness of ADBOT. Here, the equal labels ordering on average achieves the highest micro f-measure and is thus proposed as a strategy to optimize ADBOT's effectiveness.

6.5.4 Reduction of Expert Workload

The basic idea to reduce the workload for experts is to continue collecting feedback until no further improvements are expected. This can basically be implemented by assessing the effectiveness of ADBOT for the discovered alignments after each iteration. Once the effectiveness of ADBOT has not significantly changed for a few iterations, the feedback collection is turned off, i.e., the remaining activity pairs are matched automatically and there is no further interaction with the experts.

To examine whether such a strategy can actually be exploited, the effect of collecting feedback for only a subset of these model pairs is investigated. With regard to a certain ordering of model pairs this is done by turning off the feedback collection after a certain iteration i. Thus, for an ordering of length n there are $n-1$ scenarios, e.g., if there are five model pairs, feedback collection might be turned off after the first, the second, the

third, or the fourth iteration of the feedback collection process. Then, for each of the $n-1$ scenarios the overall micro f-measure that ADBOT achieves for all model pairs is measured. This includes the alignments that were determined during feedback collection as well as those that were computed after the feedback collection was turned off.

In the following all four datasets are considered. For each of the datasets the 100 random orderings as well as the equal labels ordering are investigated. As each ordering contains 36 model pairs there are 35 different scenarios for turning off the feedback collection. Thus, in total ADBOT is applied $(101 \times 35 =)$ 3535 times per dataset.

Based on these runs the effect of stopping the feedback collection is studied. That is, for stopping feedback after a certain iteration $i \in [1, 35]$ three micro f-measures are determined per dataset. First, the maximum and the minimum micro f-measures yielded by any of the random runs are considered. Moreover, the f-measure that ADBOT achieves for the equal labels ordering is investigated. Then, the development of these three micro f-measures is studied, in order to understand how the amount of feedback that is used to adjust ADBOT influences the overall effectiveness. For each dataset Figure 6.12 presents the curves for the development of the three f-measures. Note that in the diagrams the micro f-measures yielded by BOT and OPBOT are used as a baseline.

The first observation pertains the SR dataset. In contrast to the other datasets the development of the micro f-measures is quite stable. This confirms the observation that ADBOT only achieves little improvements on this dataset.

For the other three datasets there are two interesting results. The first result refers to the number of iterations that are needed to lift ADBOT's effectiveness above that of BOT and OPBOT. On BR the equal labels ordering leads to an improvement after twelve model pairs. Whereas the minimum f-measure for the random runs exceeds that of BOT and OPBOT after nine iterations, the maximum f-measure is already better after the first iteration. On UA the maximum f-measure of the random runs is also already higher than that of BOT and OPBOT when feedback is only collected for one model pair. While the minimum f-measure never exceeds the one of OPBOT on this dataset, eleven iterations need to be completed in order to improve the f-measure based on the equal labels ordering. Lastly, on AW the maximum f-measure is higher than that of BOT and OPBOT after two

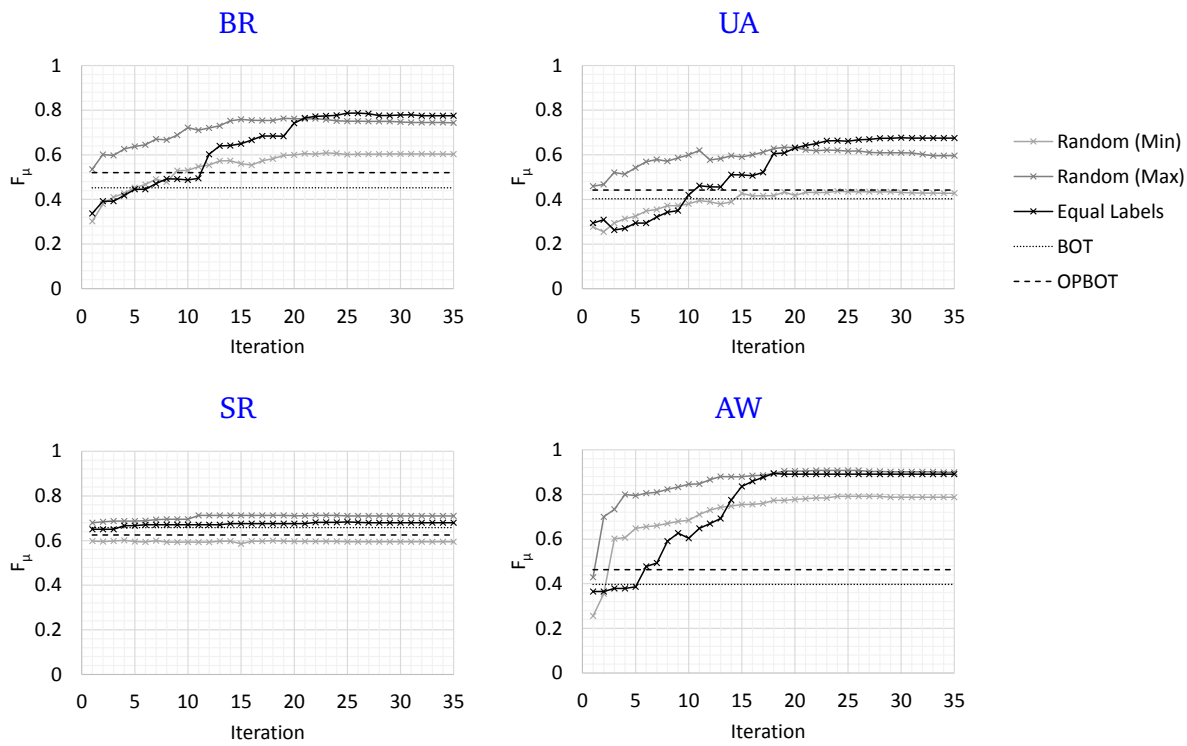

FIGURE 6.12: Micro f-measures for stopping to collect feedback after a certain iteration

and the minimum after three iterations. For the equal labels ordering six iterations need to be completed. Note that due to the nature of the equal labels ordering, it yields low micro f-measures for low iteration numbers. Here, model pairs for which a high effectiveness can be yielded without feedback analysis are matched at the beginning. As for these models the experts only need to perform a few modifications, only a few adjustments to the word similarities are made and accordingly the improvements are rather small. Yet, the results show that the amount of feedback needed to improve the effectiveness can be reduced to a few iterations.

The second result refers to the maximum effectiveness. On all datasets all curves level off between 15 and 20 iterations. That means, in order to achieve a close-to-the-maximum effectiveness, feedback does not need to be collected for all model pairs. Instead, it is sufficient to turn off feedback after 50% of the model pairs were matched. Note that here close-to-the-maximum refers to the maximal effectiveness that can be yielded for the specific ordering, not for any ordering.

These results show that the positive impact of feedback collection is almost immediate and that the maximum effectiveness can be achieved by collecting feedback for about 50%

of the model pairs. Thus, the results confirm that it is possible to reduce the workload for experts. Moreover, they provide further evidence towards the positive impact of the feedback analysis.

6.5.5 Transitivity in the Evaluation Datasets

The next analysis addresses the transitivity in the evaluation datasets. Yet, as each process model in the SR dataset is aligned to exactly one process model, transitivity cannot be examined here. Thus, the focus is on the AW dataset. To check the degree to which transitivity exists the global χ and the local clustering coefficient $\overline{\chi}$ were computed as defined in Section 6.3. Here, the local clustering coefficient is improved (AW: $\overline{\chi} = .918$ vs. BR: $\overline{\chi} = .842$, UA: $\overline{\chi} = .745$). Moreover, the global clustering coefficient is strongly increased ($\chi = .724$ vs. BR: $\chi = .479$, UA: $\chi = .274$). Hence, both values indicate that transitivity holds within the gold standard of the AW dataset. Accordingly, further evidence is provided that transitivity holds between correspondences in a model collection and can reliably be used to determine correspondences.

6.5.6 Limitations of the Feedback Analysis

The evaluation revealed that the improvements turn out differently across the datasets. Here, the largest improvement in comparison to BOT and OPBOT could be observed on AW. By contrast, on the SR dataset the average micro f-measure for ADBOT was similar to the micro f-measure of BOT and only slightly better than that of OPBOT. Thus, the question arises: *under which circumstances does ADBOT yield high improvements*? To examine this question, the improvements for a dataset are measured in terms of the indicator I_F. It is defined as the difference of ADBOT's micro f-measure in combination with the equal labels ordering and the maximum micro f-measure yielded by BOT_{ALL} and OPBOT.

A prerequisite for the improvements is that the knowledge gained through user feedback can actually be reused. This on the one hand pertains the word similarity adaptation algorithm. In order to yield improvements based on this component, the word pairs for

TABLE 6.10: Model collection characteristics vs. improvements gained by analyzing feedback

Dataset	I_w	I_c	I_F
AW	3.547	3.918	.427
BR	4.295	3.374	.255
UA	3.605	2.978	.233
SR	1.159	1.000	.022

which the similarity is adjusted must reappear in other process model pairs. To examine the extent to which word pairs reappear in different model pairs within the datasets, the indicator I_w is used here. To compute this indicator, all distinct word pairs that occur in a model collection are considered. To this end, the set of word pairs is determined by iterating over the set of all model pairs in a collection and by considering all possible activity pairs in these model pairs. For each activity pair all word pairs that contain a word from the first activity's bag-of-words and one from the bag-of-words of the second activity are added to the set of word pairs, if they are not already in the set. Note that two word pairs that contain the same words in a different ordering are considered equal. For each of the determined word pairs, the number of process model pairs they occur in is determined. Then, the indicator I_w is the average of these numbers. Consequently, the higher the indicator value, the more likely it is that a word pair for which the word similarity score has been adapted reappears and the adapted similarity value can be used. That means, the higher the indicator value the more likely it is that the derived knowledge can be reused.

The second part of ADBOT's matching process is based on transitivity. Here, the local clustering coefficients already revealed that transitivity holds across the datasets except for SR where each process model occurs only once in the list of model pairs. In addition to the local clustering coefficients, the indicator I_c is considered here. It is based on all activities for which there is at least one correspondence in the dataset. In particular, it is defined as the average number of correspondences these activities are part of. Hence, the higher the value for the indicator the more often an activity can be reused to transitively infer correspondences.

For each dataset Table 6.10 presents the three indicator values. Here, the datasets are sorted in descending order with regard to the improvement indicator I_F. The table shows a positive correlation between I_c and I_F. That means, the more often activities are part of correspondences, the larger is the improvement. Although I_w and I_F are not correlated that strongly, the three datasets AW, BR and UA for which strong improvements in the micro f-measure were yielded are characterized by high values for I_w. In contrast, a value of 1.159 on SR shows that word pairs in this dataset tend to occur in only one process model pair and thus the word similarity adaptation algorithm can only yield small improvements. While these results are not statistically significant they show that the improvements gained through feedback analysis depend on the degree to which the derived knowledge can be reused. Consequently, a successful application of ADBOT is limited to situations where the vocabulary used in different models overlaps and where process models are aligned to more than one other process model.

6.6 Summary

This section dealt with Sub-hypothesis H4 and examined the analysis of expert feedback to improve the effectiveness of matching techniques. As a first step in this regard, options to collect feedback were discussed. Here, a framework for designing feedback collection tasks in the context of process model matching was presented. This framework was developed by the author of this thesis in cooperation with other researchers [25]. It categorizes the questions that are asked to collect feedback and the answers that are expected. From this framework the specific process of feedback collection was derived. It serves as a basis for the analysis strategies and at heart works by automatically detecting an alignment and then asking experts to correct it. Based on the feedback of the experts, a matching technique can then adjust its matching process to better reflect the domain characteristics of the model collection.

Next, the chapter examined two strategies to learn from the expert feedback. On the one hand, the option to adapt the word similarities was investigated. Here, the analysis on the development datasets showed that feedback can be used to adapt the word similarities

applied by a BOT configuration in a way that the overall effectiveness of the configuration is improved. However, it was also revealed that the improvements depend on the order in which the process models are matched and on the word similarity that is adapted as well as the threshold parameter. Moreover, the adaptation improves the effectiveness for most of the model pairs, but exceptions have to be considered where the f-measure is sacrificed. On the other hand, the transitivity of alignments within a model collection was analyzed. In this regard, the examination of the development datasets showed that an activity a which is aligned to two other activities a' and a'' is a reliable evidence for the correspondence relation between a' and a''. However, it was shown that this strategy is limited by some exceptions which can partly be explained by the existence of complex correspondences.

Based on these two strategies ADBOT was introduced. This matching technique follows the process of feedback collection as introduced in the beginning of the chapter. It comprises three BOT configurations for which the underlying word similarities are adapted in each iteration of the feedback collection process. Moreover, at the beginning OPBOT's search strategy is applied to adjust the thresholds of these configurations and to rank them. During the collection of feedback the discovered alignments are used to estimate the effectiveness and to re-adjust the thresholds and the ranking of the configurations. In each iteration the process models are then matched by the best BOT configuration, or if possible, the correspondences are inferred transitively.

In addition to the examination of the two strategies, the final analysis of ADBOT's effectiveness further confirmed Sub-hypothesis H4. That is, improvements were obtained on all four datasets. Yet, depending on the order in which the model pairs are matched, ADBOT's effectiveness varied. Thus, strategies to maximize the effectiveness by ordering the model pairs were examined. Here, it was revealed that sorting model pairs according to the share of equally labeled activity pairs is a promising strategy. That is, with this ordering the effectiveness of ADBOT could be pushed close to or even beyond the maximum effectiveness that has been observed for any random ordering. Next, it was shown that the workload for experts can be minimized. In this context, empirical observation suggested that feedback for only a small share of the model pairs is sufficient to yield improvements

in comparison to automated strategies. Additionally, turning off the feedback collection after approximately 50% of the model pairs already results in a close-to-the-maximum effectiveness. Thus, experts are not required to correct all alignments. Furthermore, the finding that transitivity holds between alignments in a model collection was confirmed through an investigation of the AW dataset. Finally, the analysis also showed that the feedback analysis is limited to situations where the knowledge derived from it can be reused. With regard to ADBOT this is the case when the vocabulary used in different models overlaps and when several process models need to be aligned to each other. All findings verify the positive effect of analyzing expert feedback on the effectiveness and thus confirm Sub-hypothesis H4.

7

Discussing the Results

This chapter concludes the thesis. It summarizes the contributions of this thesis in Section 7.1. Then, it discusses the threats to validity which limit the findings in Section 7.2. Finally, the chapter presents directions for future research in Section 7.3.

7.1 Summary of the Contributions

The contributions to the field of process model matching and BPM are manifold. On an abstract level the contributions fall in one of two categories. On the one hand, there are the matching techniques: BOT, OPBOT, and ADBOT. They build on each other, they are applicable in different contexts, and in comparison to the state of the art they yield a high effectiveness. According to the ISR framework [22], these matching techniques constitute a contribution to the business environment where they help to implement the

business need. On the other hand, this thesis contains many analyses that examine many different matching propositions. That is they explicate the challenges related to process model matching as well as the suitability of strategies to tackle these problems. The according results do not only justify the design decisions underlying the three matching techniques, but – even more important – foster future research which can build on them. Thus, as demanded by the ISR framework this thesis also enriches the scientific knowledge base. In the following, the particular contributions arising from the verification of the sub-hypotheses are summarized.

Sub-hypothesis H1: *The identification of correspondences between business process models is a challenge for organizations which is not sufficiently supported by existing approaches.* To confirm this hypothesis, an overview of the use cases for process model matching techniques showcased the practical applications and verified the business need. Additionally, the state of the art on process model matching was analyzed based on a systematic literature review. In this regard, it was revealed that matching techniques from related work are generally designed to be applicable in a broad variety of scenarios. Yet, the effectiveness of matching techniques is rather low and the validity of design decisions as well as of assumptions has rarely been studied. These shortcomings motivated the research in this thesis, but also provide guidance for further research.

Sub-hypothesis H2: *Label-based matching techniques yield a varying and generally insufficient effectiveness.* In the context of this sub-hypothesis, the Bag-of-Words Technique was developed based on an analysis of the development datasets that incrementally studied the effects of different design decisions. Here, it was shown that treating labels as sets of words is a promising approach to the label-based comparison of activities. In this regard, the unification of words through stemming and of the level of abstraction through pruning was considered, but the proposed approaches have only a marginal impact on the effectiveness. The analyses of BOT showed that the effectiveness of this approach is bound by the similarity measures used to compare the words in the labels. It was argued that for a successful application, measures that reflect the domain characteristics of model collections are needed, but typically not available. This finding is backed

up by the knowledge acquisition bottleneck which has been discussed in the literature [225, 271, 272]. Further evidence was given by assessing BOT's effectiveness on the evaluation datasets and comparing it to state-of-the-art matchers. It was shown that the configuration which maximizes BOT's effectiveness varies across all datasets and performs better than the state-of-the-art matchers. Moreover, it was shown that the results of the suggested default configuration are comparable to the state-of-the-art matchers. To improve the effectiveness of the default configuration, a semi-manual configuration approach was studied. This study revealed that sometimes a huge manual effort is necessary to yield a high-performing configuration and thus the default configuration might directly be applied. Lastly, a qualitative analysis of BOT's misclassifications substantiated the finding that the comparison of the domain-specific vocabulary in model collections is not sufficiently supported by common word similarity measures.

Sub-hypothesis H3: *The optimization of the effectiveness of label-based matching techniques is enabled by the analysis of control flow information.* Control flow information that is captured in process models has been widely exploited by matching techniques in prior research. Yet, the usefulness of this information for the identification of correspondences has not been studied. Thus, three approaches to integrate this information into matching techniques were empirically analyzed on the development datasets. First, it was revealed that comparing control flow properties of activities is not suited to identify correspondences. That is, for none of the control flow similarities considered in this thesis, it could be observed that they yield values which are unique to corresponding or non-corresponding activity pairs. Second, a common assumption in the literature is that complex correspondences can be derived from the graph structure of the process models. However, the analysis of this proposition showed that according approaches face two problems. They rule out a significant amount of actual complex correspondences and in turn yield an extensive number of potential candidates which are actually not corresponding. Third, the use of control flow information to investigate the consistency of alignments was investigated. In this regard, the most important finding with regard to Sub-hypothesis H3 was revealed. The analysis suggested that control flow information

is suited to analyze the consistency of alignments. An alignment is consistent, if the control flow relations between the activities from the first model resemble the relations between their corresponding counterparts in the second model. In particular, the order relation score $\delta_{\rightarrow a}$ was introduced. It was shown that the values yielded by applying this score to alignments are positively correlated to the effectiveness of the alignments. Besides providing guidance for the development of matchers in future work the finding was used to design the Order Preserving Bag-of-Words Technique. OPBOT estimates the effectiveness of different BOT configurations by computing order relation scores for the alignments that they propose for a given model collection. By combining the most promising results it improves the effectiveness of BOT's default configuration. However, whereas the default BOT configuration can directly be applied to a model pair, OPBOT relies on the analysis of an entire model collection and is thus only applicable in situations where such a collection exists. The suitability of OPBOT's search strategy was verified by evaluating its effectiveness on all datasets. Yet, the evaluation also showed that OPBOT's effectiveness is limited by the reduced configuration space it considers. Additionally, it could be demonstrated that the automatic configuration of BOT implemented by OPBOT makes the semi-manual configuration approach studied in Chapter 4 obsolete. The reason is that in most cases OPBOT yielded a higher quality than a BOT configuration that is trained on alignments that were manually provided for 25% of the model pairs in a collection. Then, the analysis of the order relation score on the evaluation datasets gave further evidence to the general validity of the finding that control flow information is suitable to investigate the consistency of alignments. Finally, it was shown that the idea of estimating the effectiveness of matching techniques based on the order relation score is portable to the more general problem of matcher selection.

Sub-hypothesis H4: *The effectiveness of matching techniques is improved by the utilization of expert feedback.* While the effectiveness of BOT and OPBOT is bound by the degree to which the word similarities reflect the domain characteristics of the model collection, the Adaptive Bag-of-Words Technique is based on the idea that the effectiveness can be improved, if feedback provided by experts is analyzed and used to adjust the matcher to the

domain characteristics. In this context, different ways to collect feedback were discussed and one particular approach was selected. This approach consists in iteratively matching model pairs from a model collection. For each pair the automatically determined alignment is presented to the expert who corrects the alignment. These corrections are then used to adapt the matching process. Here, evidence from the development datasets suggested that such feedback can be exploited to adapt word similarities and to transitively infer correspondences. Consequently, both strategies were integrated into ADBOT. It comprises three BOT configurations for which it adapts the word similarities and it also stores the true alignments derived from the expert feedback. In each iteration of the feedback collection process it matches a model pair by using the best ranked BOT configuration, or by transitively inferring the alignment from the already discovered alignments, if that is possible. Additionally, it uses OPBOT's search to initially set up the BOT configurations and later refines the configurations based on the feedback. The final evaluation showed that ADBOT outperforms BOT, OPBOT, and the state-of-the-art matchers. In this regard, strategies to determine an order in which the model pairs are matched were proposed. While all strategies maximize ADBOT's effectiveness, the equal labels ordering lead to the best effectiveness on average. Furthermore, it was revealed that feedback does not need to be collected for all model pairs in order to obtain improvements in the effectiveness. Instead, only a few iterations are sufficient to yield improvements and collecting feedback for about 50% of the model pairs results in a close-to-the-maximum effectiveness. In comparison to BOT and OPBOT the examined strategies are limited to situations where experts are available to correct automatically determined alignments. Moreover, the analysis of the datasets revealed that it is necessary that the knowledge which is gained by learning from the feedback can actually be reused.

In summary, the thesis revealed that little evidence towards design decisions is given in the literature and that the state-of-the-art matching techniques yield a generally low effectiveness. It was then demonstrated that the effectiveness of fully automated matching techniques is typically limited by the degree to which the underlying assessment of the label similarity reflects the domain characteristics of the model collection. However, it was also shown that automatically configuring label-based matching techniques by

examining control flow relations between activities constitutes a strategy to optimize the effectiveness. Additionally, evidence was provided that the analysis of expert feedback allows to adjust matching techniques to the domain characteristics and to increase the effectiveness. Thus, the findings verify the main research hypothesis:

> **H0**: The adaptation of business process model matching techniques to model collections is necessary to ensure a high effectiveness and the analysis of the control flow as well as of expert feedback provides means to implement this adaptation.

7.2 Threats to Validity

The validity of the contributions that the previous section summarized is limited by a few threats. Such threats typically concern the *internal* and the *external validity* of the findings [337]. Moreover, threats in empirical research are also related to the *construct* and the *conclusion validity* of the results [338, 339]. In the following, all four types will be discussed.

In general, the conclusion validity refers to the degree to which the relationship between the treatment and the outcome holds [338]. In the context of this thesis, this refers to the degree to which the effectiveness of the matching techniques can actually be traced back to their design. In this regard, the research approach underlying this thesis was designed to minimize the threats to the conclusion validity. Instead of solely relying on the evaluation of matching techniques to verify their effectiveness, the research design explicitly incorporated empirical analyses to foster the understanding of the challenges related to business process model matching as well as of the impacts of various design decisions.

In this connection, the internal validity pertains the causality of a relationship between the treatment and the outcome [338]. Similar to the conclusion validity the internal validity was also addressed by the research design. That is, established qualitative and quantitative methods were applied to conduct the analyses. Moreover, throughout the thesis these research methods were made explicit, so that the analyses results and their

limitations are comprehensible. Additionally, three of the four empirical datasets are publicly available, so that the analyses can be repeated. Lastly, following established guidelines [37], development and evaluation data was separated. This way, a more realistic assessment of the effects of design decisions and the effectiveness of the matching techniques was ensured and the threat of drawing overly optimistic pictures was limited with regard to the findings.

The construct validity is determined by the degree to which the chosen constructs reflect the cause and the outcome [338]. Accordingly, in this thesis the construct validity is threatened by how the effectiveness of the matching techniques is measured. Here, an established setup from comparative evaluations of process model matching techniques [19, 20] as well as from the field of schema and ontology matching [26–29] has been applied. However, the use of a binary gold standards compromises the construct validity. That is, these gold standards define whether correspondence relations between activities hold or not. Hence, the standards suggest that there is a ground truth which represents the commonly shared perception of experts. Yet, in line with [24], the author of this thesis in collaboration with other researchers found that the perception of experts regarding the correspondence relations between activities is more diverse than a binary gold standards suggests [25]. To mitigate this threat there were four different datasets used in this thesis. Each of the datasets comprised a gold standard which was created by different experts. Thus, overall the gold standards reflect the opinion of a broad variety of experts.

Finally, the external validity is concerned with the degree to which the results can be generalized [338]. The need for process model matching techniques arises from the existence of model collections that comprise hundreds or thousands of models. With that in mind, the use of 144 model pairs cannot be regarded as an exhaustive evaluation. This number of model pairs most notably limits the degree to which real-life situations are reflected. In this regard, the most serious limitation is the use of three out of four datasets which consist of process models that all refer to the same abstract process. Yet, as outlined in the literature analysis, this is a problem for all works in this field. Additionally, it was shown that the size of the empirical data in this thesis constitutes a comprehensive dataset collection in comparison to other works in the field. With the two evaluation datasets

that were developed in the context of this thesis, the author also aimed to improve the situation. Nevertheless, the author of this thesis acknowledges that a broader dataset collection is required to further substantiate the external validity of the findings.

7.3 Future Research

The research in this thesis provides the basis for further research on process model matching and related fields. First, the results showed that the use of universal knowledge sources in order to assess the similarity of activities based on their labels is likely to yield a poor effectiveness. However, the results also demonstrated that by designing more flexible techniques which are adaptive towards model collection characteristics, the assessment of the domain characteristics can be improved and the effectiveness of matchers can be lifted. In line with these observations, the author sees the improvement of the flexibility of matchers as a promising research direction. Here, more sophisticated linguistic models might be used as a basis for unsupervised and supervised methods that aim to adjust matching techniques to the characteristics of model collections.

Second, as discussed in Section 7.2 there are threats that limit the external and the construct validity of the results presented in this thesis. Consequently, future work should address the respective issues by improving the evaluation setup. Regarding the external validity the size of the data and the coverage of matching scenarios needs to be extended. This warrants a more reliable assessment of the general validity of matching techniques. However, there are two major challenges related to the extension of experimental data collections. On the one hand, real-world models or even entire model collections are needed. This requires the willingness of organizations to share their models and collections. In this regard, emphasis should be put on covering a broad range of scenarios where models are compared within and across functional areas in an intra- as well as an inter-organizational context. Another way to collect such data is to rely on models created by students in the context of business process lectures. This strategy is supported by experiments which revealed that students and experts perform similar when analyzing business process models [40]. Again, it should be taken care that the models cover a

diverse range of scenarios. If not bound to contractual obligations researchers should also share their data collections to ensure availability of versatile model collections. In this context, a first promising step are initiatives like the model matching contests [19, 20] or the BPM Academic Initiative[31] which aim to provide empirical datasets to researchers. On the other hand, in addition to the models the extension of empirical data requires the creation of new gold standards. Here, it is conceivable to outsource the collection of gold standards to crowd-workers who performed similar to experts in experiments [25].

The construct validity is limited due to the use of binary gold standards. In this regard, more fine-grain gold standards should be created that better reflect semantic relationships between activities and the diverse points of view that experts may have. Additionally, different gold standards for the model collections could be introduced that refer to a variety of use cases. Here, crowd-sourcing is a promising alternative to ensure the availability of assessor time. Linked to the introduction of non-binary gold standards is the demand to conduct research on the perception of experts, in order to foster the understanding of the nature of correspondence relations. This will not only inform a more realistic evaluation of matchers, but also the design of better matching techniques.

In prior research many approaches that rely on process model matchers abstract from the use of matchers and consider the results to be given. In this regard, integrating process model matchers into these approaches is related to a couple of challenges. First, the result quality of the matchers might impact the quality of the overall approach. Second, these approaches are typically automated and might be extended in order to collect feedback that can be used to improve the quality of the matching techniques and thus of the approaches. Third, the existence of complex correspondences which contain sub-graphs that are not necessarily connected has often been overlooked. Accordingly, future research needs to address these challenges in order to prepare these approaches for practical application.

Finally, process model matching has focussed on the design of matching techniques that automate the matching process. Involving experts into this process has however not been studied in the field of process model matching. Thus, research in this regard should focus on assisting experts in understanding relations between process models

[31] http://bpmai.org/download/index.html, accessed: 13/01/2017

and in manually identifying correspondences. Furthermore, the interpretation of the results of matching techniques needs to be studied in order to ease the application of the techniques.

A

Identified Literature

TABLE A.1: References identified during the literature search with topic classification and first source of occurrence (part I)

Reference	Topic	First Source
[340]	Querying	Springer
[341]	Similarity Search	IEEE Explore
[342]	Similarity Search	IEEE Explore
[160]	Model Matching	[116]
[343]	Similarity Search	Google Scholar
[344]	Similarity Search	Springer
[161]	Model Matching	[116]
[345]	Similarity Search	IEEE Explore
[116]	Model Matching	Matching Contest
[180]	Pattern Matching	IEEE Explore
[346]	Similarity Search	ACM Digital
[347]	Process Querying	ACM Digital
[348]	Similarity Search	IEEE Explore
[7]	Similarity Search	Springer
[349]	Pattern Matching	IEEE Explore
[350]	Similarity Search	IEEE Explore
[189]	Model Matching	ACM Digital
[351]	Process Annotation	ACM Digital
[154]	Modeling Support	IEEE Explore
[352]	Process Querying	ACM Digital
[353]	Process Annotation	ACM Digital
[162]	Model Matching	Matching Contest
[354]	Clone Detection	Google Scholar
[190]	Model Matching	IEEE Explore
[191] [192]	Process Model Matching	IEEE Explore

TABLE A.2: References identified during the literature search with topic classification and first source of occurrence (part II)

Reference	Topic	First Source
[355]	Similarity Search	IEEE Explore
[177]	Similarity Search	IEEE Explore
[16]	Process Model Matching	Matching Contest
[153]	Modeling Support	Google Scholar
[194]	Process Model Matching	Springer Link
[178]	Similarity Search	IEEE Explore
[182]	Business-IT Alignment	ACM Digital
[197]	Process Model Matching	Springer Link
[198]	Process Model Matching	Matching Contest
[201]	Process Model Matching	Springer Link
[19]	Process Model Matching	Matching Contest
[202]	Process Model Matching	Emeral Insight
[179]	Similarity Search	IEEE Explore
[204]	Process Model Matching	Springer Link
[205]	Process Model Matching	Springer Link
[356]	Similarity Search	IEEE Explore
[150]	Clone Detection	ACM Digital
[357]	Similarity Search	IEEE Explore
[181]	Modeling Support	Springer Link
[176]	Process Model Matching	Springer Link
[175]	Process Model Matching	Springer Link
[20]	Process Model Matching	Matching Contest

Acronyms

2G	Bigram Similarity.
3G	Trigram Similarity.
4G	Quadrigram Similarity.
2CS	Two Words Contextual Similarity.
3CS	Three Words Contextual Similarity.
4CS	Four Words Contextual Similarity.
5CS	Five Words Contextual Similarity.
7PMG	Seven Process Modeling Guidelines.
ADBOT	Adaptive Bag-of-Words Technique.
AN	Activity-Noun labeling style.
AW	Alma Web.
BOT	Bag-of-Words Technique.
BPEL	Web Services Business Process Execution Language.
BPM	Business Process Management.
BPMN	Business Process Model and Notation.
BPR	Business Process Reengineering.
BR	Birth Registration.

CoPF Co-occurrence Pruning Function.

DES descriptive labeling style.

EPC Event Driven Process Chain.
EQL Equal String Similarity.
ERP Enterprise Resource Planning.

FreqPF Frequency Pruning Function.

H/S Hirst-St.Onge Similarity.
HAM Normalized Hamming Similarity.

IS Information Systems.
ISR Information Systems Research.

J/C Jiang-Conrath Similarity.
J/W Jaro Winkler Measure.
JWI MIT Java Wordnet Interface.

L/C Leacock-Chodorow Similarity.
LCS Longest Common Sub-Sequence Similarity.
LESK Lesk Similarity.
LEV Levenshtein Similarity.
LIN Lin Similarity.

MaxPF Maximum Pruning Function.

NA No-Action labeling style.

OPBOT	Order Preserving Bag-of-Words Technique.
PR	Precision Recall.
PSA	Porter Stemming Algorithm.
RES	Resnik Similarity.
ROC	Receiver Operating Characteristic.
RPST	Refined Process Structure Tree.
SR	SAP Reference Model.
SUB	Sub-String Similarity.
UA	University Admission.
UML	Unified Modeling Language.
VO	Verb-Object labeling style.
W/P	Wu-Palmer Similarity.
WFMS	Workflow Management Systems.
WSA	WordNet Stemming Algorithm.
WSD	Word Sense Disambiguation.

Symbols

a	Activity.
ap	Activity pair.
A	Set of activities.
$\mathscr{A}$	Alignment.
$\mathbb{A}$	Sequence of alignments.
ϖ	Bag-of-words.
O^{∞}	Set of bag-of-words.
$+$	Exclusive order relation.
$\parallel$	Interleaving order relation.
$\mathscr{B}$	Behavioral profile.
$\rightsquigarrow$	Strict order relation.
χ	Global clustering coefficent.
$\overline{\chi}$	Local clustering coefficent.
cod	Co-domain of a function.
c	Correspondence.
$a \rightarrow$	End distance of an activity.
$\rightarrow a$	Start distance of an activity.
dom	Domain of a function.

E	Set of edges.
$\mathscr{E}$	Set of events.
FN	Set of false negatives.
FP	Set of false positives.
F	F-measure.
F_M	Macro fmeasure.
F_μ	Micro fmeasure.
$frag$	Fragment.
F	Set of fragment.
$\mathscr{G}$	Set of gateways.
λ	Label function.
$\mathscr{L}$	Set of labels.
ι	Lane assignment function.
L	Set of lanes.
$match$	Matching process.
$\mathbb{N}$	Natural numbers.
$\bullet a \bullet$	Neighborhood of an activity.
$a \bullet$	Downstream neighborhood of an activity.
$\bullet a$	upstream neighborhood of an activity.
n	Node.
N	Set of nodes.
$norm$	Normalization function.
δ	Order relation score of a model collection.

γ	Order relation score of an alignment.
$\rightarrow$	Path.
$\overset{*}{\rightarrow}$	Set of paths.
Θ	Set of places.
$\mathscr{P}$	Powerset.
pr	Precision.
pr_M	Macro precision.
pr_μ	Micro precision.
P	Process model.
$\mathbb{P}$	Sequence of pair of process models.
MP	Pair of process models.
Π	Absolute property function.
π	Normalized property function.
$prune$	Pruning function.
re	Recall.
re_M	Macro recall.
re_μ	Micro recall.
R	Refined process structure tree.
act	Activity function for the RPST.
$depth$	Depth function for the RPST.
$\downarrow R$	RPST distance of an activity.
$\bullet R \bullet$	RPST neighborhood of an activity.
$\sigma.$	Similarity function.
$\sigma.\varpi$	Bag-of-words similarity function.
$\sigma.\lambda$	Label similarity function.
$\sigma.\pi$	Property similarity function.
$\sigma.w$	Word similarity function.

ρ	Spearman's rank correlation coefficient.
s	Stem of a word.
st	Stemming function.
$supp$	Support of a function.
ϑ	Threshold.
tok	Tokenization function.
θ	Trace.
Θ	Set of traces.
T	Set of transitions.
A^3	Set of activity triplets.
TN	Set of true negatives.
TP	Set of true positives.
τ	Type function.
$\mathscr{T}$	Set of types.
$\succ_P$	Weak order relation.
w	Word.
$\mathscr{W}$	Set of words.

References

[1] M. Hammer and J. Champy. *Reengineering the Corporation: A Manifesto for Business Revolution* (Harper Business, New York, NY, USA, 1993). 2, 29

[2] H. Smith and P. Fingar. *Business Process Management: The Third Wave* (Meghan-Kiffer Press, Tampa, FL, USA, 2003). 2, 29, 74

[3] R. van der Meulen and J. Rivera. *Gartner says by 2016, 70 percent of the most profitable companies will manage their business processes using real-time predictive analytics or extreme collaboration* (2013). Http://www.gartner.com/newsroom/id/2349215, Accessed: 03/08/2016. 2

[4] M. Dumas, M. La Rosa, J. Mendling, and H. A. Reijers. *Fundamentals of Business Process Management* (Springer, Heidelberg, Germany, 2013). 2, 29, 30, 34

[5] M. Weske. *Business Process Management: Concepts, Languages, Architectures, 2nd Edition* (Springer, Berlin, Germany, 2012). 2, 26, 27, 30, 162

[6] C. C. Ekanayake, M. La Rosa, A. H. ter Hofstede, and M.-C. Fauvet. *Fragment-based version management for repositories of business process models*. In *2011th Confederated international conference On the move to meaningful internet systems*, pp. 20–37 (Hersonissos, Greece, 2011). 3, 65

[7] R. Akkiraju and A. Ivan. *8th international conference on service-oriented computing*. In *Discovering Business Process Similarities: An Empirical Study with SAP Best Practice Business Processes*, pp. 515–526 (San Francisco, CA, USA, 2010). 3, 65, 250

[8] P. Frederiks and T. van der Weide. *Information modeling: The process and the required competencies of its participants*. In *9th International Conference on Applications of Natural Languages to Information Systems*, pp. 123–134 (Salford, UK, 2004). 3, 33

[9] P. J. M. Frederiks and T. P. van der Weide. *Information modeling: The process and the required competencies of its participants*. Data & Knowledge Engineering **58**(1), 4–20 (2006). 33, 34

[10] S. J. Hoppenbrouwers, H. A. Proper, and van der Weide, Theo P. *A fundamental view on the process of conceptual modeling*. In *24th International Conference on Conceptual Modeling*, pp. 128–143 (Klagenfurt, Austria, 2005).

[11] P. Rittgen. *Negotiating models*. In *Advanced Information Systems Engineering*, pp. 561–573 (2007). 3, 33, 38

[12] R. M. Dijkman. *A classification of differences between similar business processes*. In *11th IEEE International Enterprise Distributed Object Computing Conference*, pp. 37–50 (Annapolis, MD, USA, 2007). 3, 70

[13] R. M. Dijkman. *Diagnosing differences between business process models*. In *6th International Conference on Business Process Management*, pp. 261–277 (Milan, Italy, 2008). 3, 50, 70

[14] A. Awad, S. Sakr, M. Kunze, and M. Weske. *Design by selection: A reuse-based approach for business process modeling*. In *30th International Conference on Conceptual Modeling*, pp. 332–345 (Brussels, Belgium, 2011). 4, 66

[15] S. Sakr, A. Awad, and M. Kunze. *Querying process models repositories by aggregated graph search*. In *International Workshop on Reuse in Business Process Management*, pp. 573–585 (Tallinn, Estonia, 2012). 4, 66

[16] M. C. Branco, J. Troya, K. Czarnecki, J. Küster, and H. Völzer. *Matching business process workflows across abstraction levels*. In *ACM/IEEE 15th International Conference on Model Driven Engineering Languages and Systems*, pp. 626–641 (Innsbruck, Austria, 2012). 4, 67, 79, 80, 85, 86, 99, 158, 171, 172, 177, 251

[17] C. Li, M. Reichert, and A. Wombacher. *The minadept clustering approach for discovering reference process models out of process variants*. International Journal of Cooperative Information Systems **19**(3–4), 159–203 (2010). 5, 65

[18] B. N. Yahya and H. Bae. *Generating reference business process model using heuristic approach based on activity proximity*. In *3rd International Conference on Intelligent Decision Technologies*, pp. 469–478 (Piraeus, Greece, 2011). 5, 68

[19] U. Cayoglu, R. Dijkman, M. Dumas, P. Fettke, L. García-Bañuelos, P. Hake, C. Klinkmüller, H. Leopold, A. Ludwig, P. Loos, J. Mendling, A. Oberweis, A. Schoknecht, E. Sheetrit, T. Thaler, M. Ullrich, I. Weber, and M. Weidlich. *The process model matching contest 2013*. In *3rd International Workshop on Process Model Collections: Management and Reuse*, pp. 442–463 (Beijing, China, 2013). 5, 11, 13, 17, 19, 23, 63, 73, 80, 82, 83, 85, 86, 87, 88, 89, 92, 93, 97, 136, 137, 145, 146, 156, 187, 188, 224, 245, 247, 251

[20] G. Antunes, M. Bakhshandeh, J. Borbinha, J. Cardoso, S. Dadashnia, C. Di Francescomarino, M. Dragoni, P. Fettke, A. Gal, C. Ghidini, P. Hake, A. Khiat, C. Klinkmüller, E. Kuss, H. Leopold, P. Loos, C. Meilicke, T. Niesen, C. Pesquita, T. Peus, A. Schoknecht, E. Sheetrit, A. Sonntag, H. Stuckenschmidt, T. Thaler, I. Weber, and M. Weidlich. *The process model matching contest 2015*. In *6th International Workshop on Enterprise Modelling and Information Systems Architectures*, pp. 127–155 (Innsbruck, Austria, 2015). 5, 11, 13, 17, 19, 23, 63, 73, 74, 82, 85, 86, 87, 88, 89, 90, 92, 93, 136, 137, 138, 145, 146, 187, 188, 189, 191, 195, 224, 227, 245, 247, 251

[21] H. Österle, J. Becker, U. Frank, T. Hess, D. Karagiannis, H. Krcmar, P. Loos, P. Mertens, A. Oberweis, and E. J. Sinz. *Memorandum on design-oriented information systems research*. European Journal of Information Systems **20**(1), 7–10 (2011). 8

[22] A. R. Hevner, S. T. March, J. Park, and S. Ram. *Design science in information systems research*. MIS Quarterly **28**(1), 75–105 (2004). 8, 9, 17, 239

[23] J. Recker. *Scientific Research in Information Systems: A Beginner's Guide* (Springer, Berlin, Germany, 2013). 10, 15

[24] S. P. Harter. *Variations in relevance assessments and the measurement of retrieval effectiveness*. Journal of the American Society for Information Science **47**(1), 37–49 (1996). 11, 245

[25] C. Rodríguez, C. Klinkmüller, I. Weber, F. Daniel, and F. Casati. *Activity matching with human intelligence*. In *BPM Forum at the 14th International Conference on Business Process Management* (Rio de Janeiro, Brazil, 2016). 11, 19, 200, 236, 245, 247

[26] Z. Dragisic, K. Eckert, J. Euzenat, D. Faria, A. Ferrara, R. Granada, V. Ivanova, E. Jiménez-Ruiz, A. O. Kempf, P. Lambrix, S. Montanelli, H. Paulheim, D. Ritze, P. Shvaiko, A. Solimando, C. T. dos Santos, O. Zamazal, and B. C. Grau. *Results of the ontology alignment evaluation initiative 2014*. In *9th International Workshop on Ontology Matching*, pp. 61–104 (Trentino, Italy, 2014). 11, 17, 63, 73, 245

[27] B. C. Grau, Z. Dragisic, K. Eckert, J. Euzenat, A. Ferrara, R. Granada, V. Ivanova, E. Jiménez-Ruiz, A. O. Kempf, P. Lambrix, A. Nikolov, H. Paulheim, D. Ritze, F. Scharffe, P. Shvaiko, C. T. dos Santos, and O. Zamazal. *Results of the ontology alignment evaluation initiative 2013*. In *8th International Workshop on Ontology Matching*, pp. 61–100 (Sydney, Australia, 2013). 17, 63

[28] Z. Bellahsene, A. Bonifati, F. Duchateau, and Y. Velegrakis. *On evaluating schema matching and mapping*. In Z. Bellahsene, A. Bonifati, and E. Rahm, eds., *Schema Matching and Mapping*, pp. 253–291 (Springer, Berlin, Heidelberg, Germany, 2011). 16, 63, 202

[29] H.-H. Do, S. Melnik, and E. Rahm. *Comparison of schema matching evaluations*. In *2nd International Workshop on Web Databases*, pp. 221–237 (Erfurt, Germany, 2002). 16, 63, 245

[30] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval* (Cambridge University Press, Cambridge, England, 2008). 11, 14, 16, 63, 125, 130

[31] J. Rowley and F. Slack. *Conducting a literature review*. Management Research News **27**(6), 31–39 (2004). 11

[32] J. vom Brocke, A. Simons, B. Niehaves, B. Niehaves, K. Reimer, R. Plattfaut, and A. Cleven. *Reconstructing the giant: On the importance of rigour in documenting the literature search process*. In *17th European Conference on Information Systems*, pp. 2206–2217 (Verona, Italy, 2009). 11, 12, 13, 73

[33] H. M. Cooper. *Organizing knowledge syntheses: A taxonomy of literature reviews*. Knowledge in Society **1**(1), 104–126 (1988). 12

[34] P. Salipante, W. Notz, and J. Bigelow. *A matrix approach to literature reviews*. Research in organizational behavior: an annual series of analytical essays and critical reviews **4**, 321–348 (1982). 13

[35] J. Webster and R. T. Watson. *Analyzing the past to prepare for the future: Writing a literature review*. MIS Quarterly **26**(2), xiii–xxiii (2002). 13

[36] M. Sanderson and M. Braschler. *Best practices for test collection creation and information retrieval system evaluation: Trebleclef technical report*. Information Research **18**(2) (2009). 13

[37] J. Zobel. *Writing for Computer Science* (Springer, Heidelberg, Germany, 2004). 13, 87, 88, 93, 245

[38] J. Mendling, H. A. Reijers, and J. Recker. *Activity labeling in process modeling: Empirical insights and recommendations*. Information Systems **35**(4), 467–482 (2010). 14, 69, 103

[39] H. Leopold. *Natural Language in Business Process Models: Theoretical Foundations, Techniques, and Applications* (Springer, Cham, Switzerland, 2013). 33, 40, 69, 103, 104, 122

[40] H. A. Reijers and J. Mendling. *A study into the factors that influence the understandability of business process models*. Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on **41**(3), 449–462 (2011). 14, 89, 246

[41] B. Carterette, V. Pavlu, E. Kanoulas, J. A. Aslam, and J. Allan. *Evaluation over thousands of queries*. In *31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 651–658 (Singapore, Singapore, 2008). 14

[42] J. W. Creswell. *Research Design: Qualitative, Quantitative, and Mixed Methods Approaches* (Sage Publications, Thousands Oaks, CA, USA, 2003). 15

[43] F. J. Massey Jr. *The kolmogorov-smirnov test for goodness of fit*. Journal of the American Statistical Association **46**(253), 68–78 (1951). 16, 168

[44] P.-N. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining* (Pearson Education Limited, Harlow, 2014). 16, 169

[45] P. Mayring. *Qualitative Inhaltsanalyse: Grundlagen und Techniken* (Beltz, Weinheim, Germany, 2010). 16, 141, 171

[46] P. Mayring. *Qualitative content analysis*. Forum Qualitative Social Research **1**(2) (2000). 16, 141

[47] M. Sanderson. *Test collection based evaluation of information retrieval systems*. Foundations and Trends in Information Retrieval **4**(4), 247–375 (2010). 16, 63

[48] Ontology Alignment Evaluation Initiative. *Towards a methodology for evaluating alignment and matching algorithms version 1.0* (2005). Http://oaei.ontologymatching.org/doc/oaei-methods.1.pdf, Accessed: 03/08/2016. 17, 63

[49] C. Klinkmüller, S. Mutke, A. Ludwig, and Bogdan Franczyk. *Towards automated logistics service comparison - decision support for logistics network management*.

In *14th International Conference on Enterprise Information Systems*, pp. 259–264 (Wroclaw, Poland, 2012). 19, 23, 67

[50] C. Klinkmüller, I. Weber, J. Mendling, H. Leopold, and A. Ludwig. *Increasing recall of process model matching by improved activity label matching*. In *11th International Conference on Business Process Management*, pp. 211–218 (2013). 23, 75, 89

[51] C. Klinkmüller, H. Leopold, I. Weber, J. Mendling, and A. Ludwig. *Listen to me: Improving process model matching through user feedback*. In *10th International Conference on Business Process Management*, pp. 84–100 (Eindhoven, The Netherlands, 2014). 19, 23, 75, 89

[52] C. Klinkmüller and I. Weber. *Analyzing control flow information to improve the effectiveness of process model matching techniques*. Decision Support Systems (2017). URL http://www.sciencedirect.com/science/article/pii/S0167923617301148. 19, 23

[53] S. Zehr and C. Klinkmüller. *Prozesskonsolidierung – ein automatisierbares vorgehensmodell*. Versicherungsforen – Themendossier **14**(21), 6–9 (2014). 20

[54] C. Klinkmüller. *Innovationen im versicherungsbetrieb: Die wirtschaftsfakultät der uni leipzig entwickelt vorgehensmodell zur prozesskonsolidierung. interview by mareen rühle* (2015). Http://blog.versicherungsforen.net/2015/09/innovationen-im-versicherungsbetrieb-die-wirtschaftsfakultaet-der-uni-leipzig-entwickelt-vorgehensmodell-zur-prozesskonsolidierung/, Accessed: 03/08/2016. 20

[55] P. Lawrence. *Workflow Handbook 1997* (John Wiley & Sons, New York, NY, USA, 1997). 27

[56] F. W. Taylor. *The principles of scientific management* (Harper & Brothers, New York, 1911). 29

[57] T. H. Davenport and J. E. Short. *The new industrial engineering: Information technology and business process redesign*. Sloan Management Review **31**(4), 11–27 (1990). 29

[58] M. Hammer. *Reengineering work: Don't automate, obliterate*. Harvard Business Review pp. 104–112 (1990). 29

[59] W. E. Deming. *Statistical techniques in industry*. Advanced Management **18**(11), 8–12 (1953). 29

[60] W. A. Shewhart. *Statistical Method from the Viewpoint of Quality Control* (Dover Publications, Mineola, NY, USA, 1986). 30

[61] S. Conger. *Six sigma and business process management*. In J. vom Brocke and M. Rosemann, eds., *Handbook on Business Process Management 1*, pp. 127–148 (Springer, Berlin, Germany, 2010). 30

[62] W. M. van der Aalst, A. H. ter Hofstede, and M. Weske. *Business process management: A survey*. In *1st International Conference on Business Process Management*, pp. 1–12 (Eindhoven, The Netherlands, 2003). 30

[63] M. Hammer. *What is business process management?* In J. Brocke and M. Rosemann, eds., *Handbook on Business Process Management 1*, pp. 3–16 (Springer, Berlin, Germany, 2010).

[64] J. Mendling. *Metrics for Process Models: Empirical Foundations of Verification, Error Prediction, and Guidelines for Correctness* (Springer, Berlin, Germany, 2008). 33, 70, 90

[65] M. zur Muehlen. *Workflow-based Process Controlling: Foundation, Design, and Application of Workflow-driven Process Information Systems* (Logos Verlag, Berlin, Germany, 2002). 30

[66] J. L. Whitten and L. D. Bentley. *Systems Analysis and Design Methods* (McGraw-Hill, New York, NY, USA, 2007). 32

[67] H. Stachowiak. *Allgemeine Modelltheorie* (Springer, Vienna, Austria, 1973). 32, 33

[68] B. Curtis, M. I. Kellner, and J. Over. *Process modeling*. Communications of the ACM **35**(9), 75–90 (1992). 33, 149

[69] S. Jablonski and C. Bussler. *Workflow Management: Modeling Concepts, Architecture and Implementation* (International Thomson Computer Press, London, 1996). 33, 149

[70] R. Schuette and T. Rotthowe. *The guidelines of modeling – an approach to enhance the quality in information models*. In *17th International Conference on Conceptual Modeling*, pp. 240–254 (Singapore, Singapore, 1998). 33, 35

[71] J. F. M. Burg. *Linguistic Instruments in Requirements Engineering* (IOS Press, Amsterdam, Netherlands, 1996). 33

[72] C. Hahn, J. Recker, and J. Mendling. *An exploratory study of it-enabled collaborative process modeling*. In *6th International Workshop on Business Process Design*, pp. 61–72 (Hoboken, NJ, USA, 2011). 34

[73] J. Pinggera, S. Zugal, M. Weidlich, D. Fahland, B. Weber, J. Mendling, and H. Reijers. *Tracing the process of process modeling with modeling phase diagrams*. In *2nd International Workshop on Empirical Research in Business Process Management*, pp. 370–382 (Clermont-Ferrand, France, 2012).

[74] P. Rittgen. *Success factors of e-collaboration in business process modeling*. In *22nd International Conference on Advanced Information Systems Engineering*, pp. 24–37 (Hammamet, Tunisia, 2010).

[75] I. Weber, J. Hoffmann, J. Mendling, and J. Nitzsche. *Towards a methodology for semantic business process modeling and configuration*. In *2nd International Workshop on Business Oriented Aspects concerning Semantics and Methodologies in Service-oriented Computing*, pp. 176–187 (Vienna, Austria, 2007). 34

[76] N. J. Cooke. *Varieties of knowledge elicitation techniques*. International Journal of Human-Computer Studies **41**(6), 801–849 (1994). 34

[77] A. Sharp and P. McDermott. *Workflow Modeling: Tools for Process Improvement and Application Development* (Norwood, MA, USA, Artech House Inc, 2008). 34

[78] D. Karagiannis and H. Kühn. *Metamodelling platforms*. In *3rd International Conference on E-Commerce and Web Technologies*, p. 182 (Aix-en-Provence, France, 2002). 34, 35, 39

[79] H. Kühn. *Methodenintegration im Business Engineering* (PhD thesis, Universität Wien, Vienna, Austria, 2004). 34

[80] Y. Wand and R. Weber. *Research commentary: information systems and conceptual modeling-a research agenda*. Information Systems Research **13**(4), 363–376 (2002). 35

[81] J. Becker, M. Rosemann, and R. Schuette. *Grundsätze ordnungsmäßiger modellierung*. Wirtschaftsinformatik **37**(5), 435–445 (1995). 35

[82] J. Becker, M. Rosemann, and C. von Uthmann. *Guidelines of business process modeling*. In W. van der Aalst, J. Desel, and A. Oberweis, eds., *Business Process Management: Models, Techniques, and Empirical Studies*, chap. Guidelines of Business Process Modeling, pp. 30–49 (Springer, Berlin, Heidelberg, Germany, 2000). 35

[83] A. Olivé. *Conceptual Modeling of Information Systems* (Springer, Berlin, 2007). 36

[84] J. J. van Griethuysen, ed. *Concepts and terminology for the conceptual schema and the information base* (International Organization for Standardization, 1982). 36

[85] J. Mendling, H. A. Reijers, and W. M. van der Aalst. *Seven process modeling guidelines (7pmg)*. Information and Software Technology **52**(2), 127–136 (2010). 36, 90

[86] H. Leopold, J. Mendling, and A. Polyvyanyy. *Supporting process model validation through natural language generation*. IEEE Transactions on Software Engineering, **40**(8), 818–840 (2014). 37

[87] J. Hoffmann, I. Weber, and G. Governatori. *On compliance checking for clausal constraints in annotated process models*. Information Systems Frontiers **14**(2), 155–177 (2012). 37, 67

[88] Y. Liu, S. Muller, and K. Xu. *A static compliance-checking framework for business process models*. IBM Systems Journal **46**(2), 335–361 (2007).

[89] S. Sadiq and G. Governatori. *Managing regulatory compliance in business processes*. In J. Vom Brocke and M. Rosemann, eds., *Handbook on Business Process Management 2*, pp. 159–175 (Springer, Berlin, Germany, 2010). 37, 67

[90] D. Fahland, C. Favre, J. Koehler, N. Lohmann, H. Völzer, and K. Wolf. *Analysis on demand: Instantaneous soundness checking of industrial business process models*. Data & Knowledge Engineering **70**(5), 448–466 (2011). 37, 70

[91] van der Aalst, Wil M.P. *Verification of workflow nets*. In *18th International Conference on the Application and Theory of Petri Nets*, pp. 407–426 (Toulouse, France, 1997). 70

[92] W. van der Aalst, K. M. van Hee, A. ter Hofstede, N. Sidorova, H. Verbeek, M. Voorhoeve, and M. T. Wynn. *Soundness of workflow nets: classification, decidability, and analysis*. Formal Aspects of Computing **23**(3), 333–363 (2011). 37, 164

[93] J. Dehnert and P. Rittgen. *Relaxed soundness of business processes*. In *13th International Conference on Advanced Information Systems Engineering*, pp. 157–170 (Interlaken, Switzerland, 2001). 38

[94] P. R. Telang and M. P. Singh. *Specifying and verifying cross-organizational business models: An agent-oriented approach*. IEEE Transactions on Services Computing **5**(3), 305–318 (2012). 38

[95] W. M. van der Aalst. *Modeling and analyzing interorganizational workflows*. In *1st International Conference on Application of Concurrency to System Design*, pp. 262–272 (Fukushima, Japan, 1998). 38

[96] N. Sidorova, C. Stahl, and N. Trčka. *Soundness verification for conceptual workflow nets with data: Early detection of errors with the most precision possible*. Information Systems **36**(7), 1026–1043 (2011). 38

[97] S. X. Sun, J. L. Zhao, J. F. Nunamaker, and O. R. Liu Sheng. *Formulating the data-flow perspective for business process management*. Information Systems Research **17**(4), 374–391 (2006). 38

[98] J. Krogstie, G. Sindre, and H. Jørgensen. *Process models representing knowledge for action: A revised quality framework*. European Journal of Information Systems **15**(1), 91–102 (2006). 38

[99] J. Krogstie. *Conceptual modeling for computerized information systems support in organizations*. Ph.D. thesis, University of Trondheim (1995). 38

[100] J. Krogstie and H. D. Jørgensen. *Quality of interactive models*. In *International Workshop on Conceptual Modeling Quality*, pp. 351–363 (Tampere, Finland, 2002). 38

[101] R. Stamper. *Signs, information, norms and systems*. In B. Holmqvist, P. B. Andersen, and H. Klein, eds., *Signs of work: semiosis and information processing in organizations*, pp. 349–398 (De Gruyter, Berlin, Germany, 1996). 38

[102] Y. Wand and R. Weber. *An ontological analysis of some fundamental information systems concepts*. In *9th International Conference on Information Systems*, pp. 213–225 (Minneapolis, MI, USA, 1988). 38

[103] Y. Wand and R. Weber. *An ontological model of an information system*. IEEE Transactions on Software Engineering **16**(11), 1282–1292 (1990).

[104] Y. Wand and R. Weber. *On the deep structure of information systems*. Information Systems Journal **5**(3), 203–223 (1995). 38

[105] M. Bunge. *Treatise on Basic Philosophy (Volume 3): Ontology I: The Furniture of the World* (D. Reidel Publishing Company, Dodrecht, Netherlands, 1977). 38

[106] M. Rosemann, P. Green, and M. Indulska. *A reference methodology for conducting ontological analyses*. In *23rd International Conference on Conceptual Modeling*, pp. 110–121 (Shanghai, China, 2004). 38

[107] Y. Wand and R. Y. Wang. *Anchoring data quality dimensions in ontological foundations*. Communications of the ACM **39**(11), 86–95 (1996). 38

[108] H. Nelson, G. Poels, M. Genero, and M. Piattini. *A conceptual modeling quality framework*. Software Quality Journal **20**(1), 201–228 (2012). 38

[109] D. Harel and B. Rumpe. *Modeling Languages: Syntax, Semantics and All That Stuff, Part I: The Basic Stuff* (Technical Report, Weizmann Science Press of Israel, Rehovot, Israel, 2000). 39

[110] C. A. Petri. *Kommunikation mit Automaten* (PhD Thesis, Universität Hamburg, Hamburg, Germany, 1962). 41, 49

[111] R. M. Dijkman, M. Dumas, and C. Ouyang. *Semantics and analysis of business process models in bpmn*. Information and Software Technology **50**(12), 1281–1294 (2008). 41

[112] van der Aalst, Wil M. P. *Formalization and verification of event-driven process chains*. Information and Software Technology **41**(10), 639–650 (1999). 41, 50

[113] N. Lohmann, E. Verbeek, and R. Dijkman. *Petri net transformations for business processes – a survey*. In K. Jensen and van der Aalst, Wil M.P, eds., *Transactions on Petri Nets and Other Models of Concurrency II*, pp. 46–63 (Springer, Berlin Heidelberg, 2009). 41

[114] M. C. Fauvet, M. La Rosa, M. Sadegh, A. Alshareef, R. M. Dijkman, L. García-Bañuelos, H. A. Reijers, van der Aalst, W.M.P., M. Dumas, and J. Mendling. *Managing process model collections with apromore*. In *8th International Conference on Service Oriented Computing*, pp. 699–701 (San Francisco, CA, USA, 2010). 41

[115] M. La Rosa, H. A. Reijers, W. M. van der Aalst, R. M. Dijkman, J. Mendling, M. Dumas, and L. García-Bañuelos. *Apromore: An advanced process model repository*. Expert Systems with Applications **38**(6), 7029–7040 (2011). 41

[116] R. M. Dijkman, M. Dumas, L. García-Bañuelos, and R. Kaarik. *Aligning business process models*. In *13th IEEE International on Enterprise Distributed Object Computing Conference*, pp. 45–53 (Auckland, New Zealand, 2009). 41, 67, 68, 72, 76, 77, 78, 80, 81, 82, 85, 86, 97, 100, 171, 179, 250

[117] R. M. Dijkman, M. Dumas, and L. García-Bañuelos. *Graph matching algorithms for business process model similarity search*. In *7th International Conference on Business Process Management*, pp. 48–63 (Ulm, Germany, 2009). 41, 66

[118] A. Polyvyanyy and M. Weidlich. *Towards a compendium of process technologies - the jbpt library for process model analysis*. In *CAiSE Forum at the 25th International Conference on Advanced Information Systems Engineering*, pp. 106–113 (Valencia, Spain, 2013). 42

[119] P. A. Bernstein, J. Madhavan, and E. Rahm. *Generic schema matching, ten years later*. Proceedings of the VLDB Endowment **4**(11), 695–701 (2011). 42, 72

[120] E. Rahm and P. A. Bernstein. *A survey of approaches to automatic schema matching*. The VLDB Journal **10**(4), 334–350 (2001). 42, 72

[121] Object Management Group. *Business process model and notation (bpmn)* (2011). Http://www.omg.org/spec/BPMN/2.0/PDF,Accessed: 03/08/2016. 43

[122] M. zur Muehlen and J. Recker. *How much language is enough? theoretical and practical use of the business process modeling notation*. In *20th International Conference on Advanced Information Systems Engineering*, pp. 465–479 (Montpellier, France, 2008). 44, 69

[123] R. M. Dijkman, M. Dumas, and C. Ouyang. *Formal Semantics and Analysis of BPMN Process Models using Petri Nets* (Technical Report, Queensland University of Technology, Brisbane, Australia, 2007). 45

[124] A.-W. Scheer. *ARIS - Vom Geschäftsprozess zum Anwendungssystem* (Springer, Berlin, Germany, 2002). 46

[125] A.-W. Scheer, M. Nüttgens, and V. Zimmermann. *Objektorientierte Ereignisgesteuerte Prozeßkette (oEPK) - Methode und Anwendung* (Technical Report, Universität des Saarlandes, Saarbrücken, Germany, 1997). 48

[126] P. Rittgen. *Modified EPCs and Their Formal Semantics* (Technical Report, Universität Koblenz-Landau, Koblenz, Germany, 1999). 48

[127] J. Mendling, G. Neumann, and M. Nüttgens. *Yet another event-driven process chain*. In *3rd International Conference on Business Process Management*, pp. 428–433 (Nancy, France, 2005). 48

[128] K. Sarshar, P. Dominitzki, and P. Loos. *Einsatz von ereignisgesteuerten prozessketten zur modellierung von prozessen in der krankenhausdomäne – eine empirische methodenevaluation*. In *4. Workshop der Gesellschaft für Informatik e.V. (GI) und Treffen ihres Arbeitkreises "Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten (WI-EPK)"*, pp. 97–116 (Hamburg, Germany, 2005). 48

[129] W. M. P. van der Aalst. *The application of petri nets to workflow management*. The Journal of Circuits, Systems and Computers **8**(1), 21–66 (1998). 49

[130] T. Murata. *Petri nets: Properties, analysis and applications*. Proceedings of the IEEE **77**(4), 541–580 (1989). 50

[131] Object Management Group. *Omg unified modeling language (omg uml)* (2015). Http://www.omg.org/spec/UML/2.5/PDF/, Accessed: 03/08/2016. 51

[132] OASIS. *Web services business process execution language version 2.0* (2007). Http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.pdf, Accessed: 03/08/2016. 52

[133] H. A. Reijers, T. Slaats, and C. Stahl. *Declarative modeling: An academic dream or the future for bpm?* In *11th International Conference on Business Process Management*, pp. 307–322 (Beijing, China, 2013). 52

[134] W. M. van der Aalst, M. Pesic, and H. Schonenberg. *Declarative workflows: Balancing between flexibility and support*. Computer Science - Research and Development **23**(2), 99–113 (2009). 52

[135] T. T. Hildebrandt and R. R. Mukkamala. *Declarative event-based workflow as distributed dynamic condition response graphs*. In *3rd Workshop on Programming Language Approaches to Concurrency and Communication-centric Software*, pp. 59–73 (Paphos, Cyprus, 2010). 52

[136] M. Montali. *Specification and Verification of Declarative Open Interaction Models - A Logic-Based Approach* (Springer, Berlin, Germany, 2010). 52

[137] D. Fahland, D. Lübke, J. Mendling, H. Reijers, B. Weber, M. Weidlich, and S. Zugal. *Declarative versus imperative process modeling languages: The issue of understandability*. In *14th International Conference on Exploring Modeling Methods in Systems Analysis and Design*, pp. 353–366 (Amsterdam, The Netherlands, 2009). 52

[138] J. Euzenat and P. Shvaiko. *Ontology Matching* (Springer, Berlin, Germany, 2013). 56, 72, 96, 99, 100

[139] S. M. Falconer and N. F. Noy. *Interactive techniques to support ontology matching*. In Z. Bellahsene, A. Bonifati, and E. Rahm, eds., *Schema Matching and Mapping*, pp. 29–51 (Springer Berlin Heidelberg, Berlin, Germany, 2011). 60, 199, 203

[140] E. Rahm. *Towards large-scale schema and ontology matching*. In Z. Bellahsene, A. Bonifati, and E. Rahm, eds., *Schema Matching and Mapping*, pp. 3–27 (Springer, Berlin, Germany, 2011). 60, 61, 77, 184

[141] M. La Rosa, M. Dumas, R. Uba, and R. Dijkman. *Business process model merging: An approach to business process consolidation*. ACM Trans. Softw. Eng. Methodol. **22**(2), 11:1–11:42 (2013). 62, 65

[142] J. H. Hayes, A. Dekhtyar, and J. Osborne. *Improving requirements tracing via information retrieval*. In *11th IEEE International Requirements Engineering Conference*, pp. 138–147 (Montery Bay, CA, USA, 2003). 64

[143] R. M. Dijkman, M. La Rosa, and H. A. Reijers. *Managing large collections of business process models – current techniques and challenges*. Computers in Industry **63**(2), 91–97 (2012). 65

[144] F. Gottschalk, T. A. Wagemakers, M. H. Jansen-Vullers, W. M. van der Aalst, and M. La Rosa. *Configurable process models: Experiences from a municipality case study*. In *21st International Conference on Advanced Information Systems Engineering*, pp. 486–500 (Amsterdam, The Netherlands, 2009). 65, 69

[145] M. Ehrig, A. Koschmider, and A. Oberweis. *Measuring similarity between semantic business process models*. In *4th Asia-Pacific Conference on Conceptual Modelling*, pp. 71–80 (Ballarat, Australia, 2007). 66, 76

[146] R. M. Dijkman, M. Dumas, B. van Dongen, R. Käärik, and J. Mendling. *Similarity of business process models: Metrics and evaluation*. Information Systems **36**(2), 498–516 (2011). 66, 76, 90, 179

[147] M. Becker and R. Laue. *A comparative survey of business process similarity measures*. Computers in Industry **63**(2), 148–167 (2012). 66, 179

[148] R. Uba, M. Dumas, L. García-Bañuelos, and M. La Rosa. *Clone detection in repositories of business process models*. In *9th International Conference on Business Process Management*, pp. 248–264 (Clermont-Ferrand, France, 2011). 66

[149] C. C. Ekanayake, M. Dumas, L. García-Bañuelos, M. La Rosa, and A. H. ter Hofstede. *Approximate clone detection in repositories of business process models*. In *10th International Conference on Business Process Management*, pp. 302–318 (Tallinn, Estonia, 2012).

[150] M. La Rosa, M. Dumas, C. C. Ekanayake, L. García-Bañuelos, J. Recker, and ter Hofstede, Arthur H.M. *Detecting approximate clones in business process model repositories*. Information Systems **49**, 102–125 (2015). 66, 251

[151] A. Awad. *Bpmn-q: A language to query business processes*. In *2nd International Workshop on Enterprise Modelling and Information Systems Architectures*, pp. 115–128 (St. Goar / Rhine, Germany, 2007). 66

[152] T. Jin, J. Wang, N. Wu, M. La Rosa, and A. H. ter Hofstede. *Efficient and accurate retrieval of business process models through indexing*. In *2010th Confederated international conference On the Move to Meaningful Internet Systems*, pp. 402–409 (Hersonissos, Greece, 2010). 66

[153] N. N. Chan, W. Gaaloul, and S. Tata. *Assisting business process design by activity neighborhood context matching*. In *10th International Conference on Service-Oriented Computing*, pp. 541–549 (Shanghai, China, 2012). 66, 75, 251

[154] F. Niedermann, S. Radeschutz, and B. Mitschang. *Design-time process optimization through optimization patterns and process model matching*. In *12th IEEE Conference on Commerce and Enterprise Computing*, pp. 48–55 (Shanghai, China, 2010). 66, 250

[155] C. Klinkmüller, R. Kunkel, A. Ludwig, and B. Franczyk. *The logistics service engineering and management platform: Features, architecture, implementation*. In *14th International Conference on Business Information Systems*, pp. 242–253 (Poznan, Poland, 2011). 67

[156] A. S. Jadhav and R. M. Sonar. *Evaluating and selecting software packages: A review*. Information and Software Technology **51**(3), 555–563 (2009). 67

[157] P. Soffer, B. Golany, and D. Dori. *Aligning an {ERP} system with enterprise requirements: An object-process based approach*. Computers in Industry **56**(6), 639–662 (2005). 67

[158] S. Kent. *Model driven engineering*. In *Third International Conference on Integrated Formal Methods*, pp. 286–298 (Turku, Finland, 2002). 67

[159] J. Hutchinson, M. Rouncefield, and J. Whittle. *Model-driven engineering practices in industry*. In *33rd International Conference on Software Engineering*, pp. 633–642 (Waikiki, HI, USA, 2011). 67

[160] S. Brockmans, M. Ehrig, A. Koschmider, A. Oberweis, and R. Studer. *Semantic alignment of business processes*. In J. Manolopoulos, Y.and Filipe, P. Constantopoulos, and J. Cordeiro, eds., *8th International Conference on Enterprise Information Systems*, pp. 191–196 (Paphos, Cyprus, 2006). 67, 75, 76, 85, 145, 183, 250

[161] S. Nejati, M. Sabetzadeh, M. Chechik, S. Easterbrook, and P. Zave. *Matching and merging of statecharts specifications*. In *29th International Conference on Software Engineering*, pp. 54–64 (Minneapolis, MN, USA, 2007). 76, 85, 155, 156, 250

[162] M. Weidlich, R. Dijkman, and J. Mendling. *The icop framework: Identification of correspondences between process models*. In *22nd International Conference on Advanced Information Systems Engineering*, pp. 483–498 (Hammamet, Tunisia, 2010). 67, 77, 78, 80, 85, 86, 87, 155, 157, 158, 164, 178, 250

[163] M. Weidlich, J. Mendling, and M. Weske. *A foundational approach for managing process variability*. In *23rd International Conference on Advanced Information Systems Engineering*, pp. 267–282 (London, UK, 2011). 68, 163, 164

[164] A. Martens, P. Fettke, and P. Loos. *Inductive development of reference process models based on factor analysis*. In *12th International Conference on Wirtschaftsinformatik*, pp. 438–452 (Osnabrück, Germany, 2015). 68

[165] F. Pittke, H. Leopold, and J. Mendling. *When language meets language: Anti patterns resulting from mixing natural and modeling language*. In *International Workshop on Process Model Collections: Management and Reuse*, pp. 118–129 (Eindhoven, The Netherlands, 2014). 69, 106

[166] B. Weber, M. Reichert, J. Mendling, and H. A. Reijers. *Refactoring large process model repositories*. Computers in Industry **62**(5), 467–486 (2011). 69

[167] I. Moreno-Montes de Oca, M. Snoeck, H. A. Reijers, and A. Rodríguez-Morffi. *A systematic literature review of studies on business process modeling quality.* Information and Software Technology **58**, 187-205 (2015). 70

[168] S. L. Salzberg. *On comparing classifiers: Pitfalls to avoid and a recommended approach*. Data Mining and Knowledge Discovery **1**(3), 317–328 (1997). 71, 86

[169] J. Demšar. *Statistical comparisons of classifiers over multiple data sets*. J. Mach. Learn. Res. **7**, 1–30 (2006). 71, 86

[170] P. Shvaiko and J. Euzenat. *A survey of schema-based matching approaches*. In S. Spaccapietra, ed., *Journal on Data Semantics IV*, pp. 146–171 (Springer, Berlin, Germany, 2005). 72

[171] P. Shvaiko and J. Euzenat. *Ontology matching: State of the art and future challenges*. IEEE Transactions on Knowledge and Data Engineering **25**(1), 158–176 (2013). 72, 136, 139, 145, 183

[172] Z. Bellahsene, A. Bonifati, and E. Rahm, eds. *Schema Matching and Mapping*. Data-Centric Systems and Applications (Springer, Berlin, Germany, 2011). 72

[173] S. Melnik, H. Garcia-Molina, and E. Rahm. *Similarity flooding: a versatile graph matching algorithm and its application to schema matching*. In *18th International Conference on Data Engineering*, pp. 117–128 (San Jose, CA, USA, 2002). 73

[174] D. Faria, C. Pesquita, E. Santos, I. F. Cruz, and F. M. Couto. *Agreement maker light results for oaei 2013*. In *8th International Conference on Ontology Matching*, pp. 101–108 (2013). 73, 83

[175] S.-M.-R. Beheshti, B. Benatallah, S. Sakr, D. Grigori, H. R. Motahari-Nezhad, M. C. Barukh, A. Gater, and S. H. Ryu. *Process Analytics: Concepts and Techniques for Querying and Analyzing Process Data* (Springer, Cham, Switzerland, 2016). 75, 251

[176] C. Tsagkani. *Graph-based process model matching*. In *Doctoral Consortium at the 12th International Conference on Business Process Management*, pp. 573–577 (Eindhoven, The Netherlands, 2014). 75, 251

[177] Y. Belhoul, M. Haddad, E. Duchene, and H. Kheddouci. *String comparators based algorithms for process model matchmaking*. In *IEEE 9th International Conference on Services Computing*, pp. 649–656 (Honolulu, Hawaii, USA, 2012). 75, 251

[178] Y. Belhoul, M. Haddad, A. Gater, D. Grigori, H. Kheddouci, and M. Bouzeghoub. *Spectral graph approach for process model matchmaking*. In *IEEE 10th International Conference on Services Computing*, pp. 408–415 (Santa Clara, California, 2013). 251

[179] F. Kacimi and A. Tari. *Vectorial signature for matching business process graphs*. In *International Conference on Advanced Networking Distributed Systems and Applications*, pp. 93–98 (Bejaia, Algeria, 2014). 251

[180] V. Gacitua-Decar and C. Pahl. *Automatic business process pattern matching for enterprise services design*. In *4th International Workshop on Service- and Process-Oriented Software Engineering*, pp. 111–118 (Bangalore, India, 2009). 75, 250

[181] K. Ternai, M. Khobreh, and F. Ansari. *An ontology matching approach for improvement of business process management*. In M. Fathi, ed., *Integrated Systems: Innovations and Applications*, pp. 111–130 (Springer, Cham, Switzerland, 2015). 75, 251

[182] K. Dahman, F. Charoy, and C. Godart. *Alignment and change propagation between business processes and service-oriented architectures*. In *IEEE 10th International Conference on Services Computing*, pp. 168–175 (Santa Clara Marriott, CA, USA, 2013). 75, 251

[183] H. Genrich and K. Lautenbach. *System modelling with high-level petri nets*. Theoretical Computer Science **13**(1), 109–135 (1981). 75

[184] D. Harel. *Statecharts: a visual formalism for complex systems*. Science of Computer Programming **8**(3), 231–274 (1987). 76

[185] G. W. Bond. *Echarts: The concise user manual: Version 1.3 beta* (2008). Http://echarts.org/Downloads/View-document/An-Introduction-to-ECharts-The-Concise-User-Manual-2008-05-20-v1.3-beta.html, Accessed: 03/08/2016. 76

[186] H. Bunke. *On a relation between graph edit distance and maximum common subgraph*. Pattern Recognition Letters **18**(8), 689–694 (1997). 76

[187] P. E. Hart, N. J. Nilsson, and B. Raphael. *A formal basis for the heuristic determination of minimum cost paths*. IEEE Transactions on Systems Science and Cybernetics **4**, 100–107 (1968). 76, 155

[188] P. E. Hart, N. J. Nilsson, and B. Raphael. *Correction to "a formal basis for the heuristic determination of minimum cost paths"*. ACM SIGART Bulletin **37**, 28–29 (1972). 76, 155

[189] A. Gater, D. Grigori, and M. Bouzeghoub. *Complex mapping discovery for semantic process model alignment*. In *12th International Conference on Information Integration and Web-based Applications & Services*, pp. 317–324 (Paris, France, 2010). 77, 78, 85, 250

[190] A. Gater, D. Grigori, M. Haddad, M. Bouzeghoub, and H. Kheddouci. *A summary-based approach for enhancing process model matchmaking*. In *IEEE 4th International Conference on Service-Oriented Computing and Applications*, pp. 1–8 (Irvine, CA, USA, 2011). 78, 85, 86, 87, 250

[191] C. Gerth, M. Luckey, J. M. Küster, and G. Engels. *Precise mappings between business process models in versioning scenarios*. In *IEEE 9th International Conference on Services Computing*, pp. 218–225 (Washington, DC, USA, 2011). 78, 85, 250

[192] C. Gerth. *Business Process Models. Change Management* (Springer, Heidelberg, Germany, 2014). 78, 250

[193] C. Gerth, M. Luckey, J. M. Küster, and G. Engels. *Detection of semantically equivalent fragments for business process model change management*. In *IEEE 8th International Conference on Services Computing*, pp. 57–64 (Miami, FL, USA, 2010). 78, 158

[194] H. Leopold, M. Niepert, M. Weidlich, J. Mendling, R. M. Dijkman, and H. Stuckenschmidt. *Probabilistic optimization of semantic process model matching*. In *10th International Conference on Business Process Management*, pp. 319–334 (Tallinn, Estonia, 2012). 79, 81, 85, 86, 87, 88, 89, 122, 164, 178, 251

[195] H. Leopold, S. Smirnov, and J. Mendling. *On the refactoring of activity labels in business process models*. Information Systems **37**(5), 443–459 (2012). 79, 122

[196] M. Richardson and P. Domingos. *Markov logic networks*. Machine Learning **62**(1–2), 107–136 (2006). 79

[197] M. Weidlich, T. Sagi, H. Leopold, A. Gal, and J. Mendling. *Predicting the quality of process model matching*. In *11th International Conference on Business Process Management*, pp. 203–210 (Beijing, China, 2013). 79, 80, 85, 158, 183, 199, 251

[198] M. Weidlich, E. Sheetrit, M. C. Branco, and A. Gal. *Matching business process models using positional passage-based language models*. In *International Conference on Conceptual Modeling*, pp. 130–137 (Hong Kong, China, 2013). 80, 82, 85, 158, 251

[199] Y. Lv and C. Zhai. *Positional language models for information retrieval*. In *32Nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 299–306 (Boston, MA, USA, 2009). 80

[200] J. Lin. *Divergence measures based on the shannon entropy*. IEEE Transactions on Information Theory **37**(1), 145–151 (2006). 80

[201] M. H. Baumann, M. Baumann, S. Schönig, and S. Jablonski. *Towards multi-perspective process model similarity matching*. In *10th International Workshop on Enterprise and Organizational Modeling and Simulation*, pp. 21–37 (Thessaloniki, Greece, 2014). 81, 82, 85, 155, 178, 251

[202] J. Fengel. *Semantic technologies for aligning heterogeneous business process models*. Business Process Management Journal **20**(4), 549–570 (2014). 81, 85, 86, 251

[203] W3C. *Owl 2 web ontology language – document overview (second edition)* (2012). Https://www.w3.org/TR/owl2-overview/, Accessed: 03/08/2016. 81

[204] J. Ling, L. Zhang, and Q. Feng. *Business process model alignment: An approach to support fast discovering complex matches*. In K. Mertins, F. Bénaben, R. Poler, and J.-P. Bourrières, eds., *Enterprise Interoperability VI*, pp. 41–51 (Springer, Cham, Switzerland, 2014). 82, 85, 86, 171, 172, 251

[205] M. Baumann, M. H. Baumann, S. Schönig, and S. Jablonski. *Resource-aware process model similarity matching*. In *1st Workshop on Resource Management in Service-Oriented Computing*, pp. 96–107 (Paris, France, 2015). 82, 85, 251

[206] R. W. Hamming. *Error detecting and error correcting codes*. The Bell System Technical Journal **29**(2), 147–160 (1950). 99

[207] S. B. Needleman and C. D. Wunsch. *A general method applicable to the search for similarities in the amino acid sequence of two proteins*. Journal of Molecular Biology **48**(3), 443–453 (1970). 100

[208] V. I. Levenshtein. *Binary codes capable of correcting deletions, insertions and reversals*. Soviet Physics Doklady **10**, 707–710 (1966). 100

[209] M. A. Jaro. *Advances in record-linkage methodology as applied to matching the 1985 census of tampa, florida*. Journal of the American Statistical Association **84**(406), 414–420 (1989). 100

[210] W. E. Winkler. *String comparator metrics and enhanced decision rules in the fellegi-sunter model of record linkage*. Proceedings of the Section on Survey Research **13**, 354–359 (1990). 100

[211] K. C. Kang, S. G. Cohen, J. A. Hess, W. E. Novak, and A. S. Peterson. *Feature-oriented domain analysis (foda) feasibility study*. Tech. rep., Carnegie-Mellon University Software Engineering Institute (1990). 100

[212] K. Czarnecki and U. W. Eisenecker. *Generative Programming: Methods, Tools, and Applications* (ACM Press / Addison-Wesley Publishing Co., New York, NY, USA, 2000). 100

[213] Z. S. Harris. *Distributional structure*. Word **10**(23), 146–162 (1954). 106

[214] C. D. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing* (MIT Press, Cambridge, MA, USA, 1999). 106, 107, 108, 109

[215] Y. Zhang, R. Jin, and Z.-H. Zhou. *Understanding bag-of-words model: a statistical framework*. International Journal of Machine Learning and Cybernetics **1**(1–4), 43–52 (2010). 106

[216] J. B. Lovins. *Development of a stemming algorithm*. Mechanical Translation and Computational Linguistics **11**(1–2), 22–31 (1968). 108

[217] C. D. Paice. *Another stemmer*. SIGIR Forum **24**(3), 56–61 (1990). 108

[218] R. Krovetz. *Viewing morphology as an inference process*. In *16th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 191–202 (Pittsburgh, PA, USA, 1993). 108

[219] J. Xu and W. B. Croft. *Corpus-based stemming using cooccurrence of word variants*. ACM Transactions on Information Systems **16**(1), 61–81 (1998).

[220] F. Peng, N. Ahmed, X. Li, and Y. Lu. *Context sensitive stemming for web search*. In *30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 639–646 (Amsterdam, The Netherlands, 2007). 108

[221] M. F. Porter. *An algorithm for suffix stripping*. Program: Electronic Library & Information Systems **40**(3), 211–218 (1980). 108

[222] G. A. Miller. *Wordnet: A lexical database for english*. Communications of the ACM **38**(11), 39–41 (1995). 109, 114, 116

[223] M. Finlayson. *Java libraries for accessing the princeton wordnet: Comparison and evaluation*. In *7th Global Wordnet Conference*, pp. 78–85 (Tartu, Estonia, 2014). 109

[224] D. A. Cruse. *The lexicon*. In M. Aronoff and J. Rees-Miller, eds., *The Handbook of Linguistics*, pp. 238–264 (Blackwell, Oxford, England, 2008). 113, 114

[225] R. Navigli. *Word sense disambiguation: A survey*. ACM Computing Surveys **41**(2), 10:1–10:69 (2009). 114, 115, 116, 145, 204, 241

[226] P. M. Roget. *Roget's International Thesaurus* (Harper Collins, New York, NY, USA, 2011). 114

[227] J. Bernard. *The Macquarie Thesaurus* (Macquarie Library, Sydney, Australia, 1984). 114

[228] C. Fellbaum, ed. *WordNet: An Electronic Lexical Database (Language, Speech, and Communication)* (MIT Press, Cambridge and MA, 1998). 114

[229] B. Hamp and H. Feldweg. *Germanet - a lexical-semantic net for german*. In *ACL workshop Automatic Information Extraction and Building of Lexical Semantic Resources for NLP Applications*, pp. 9–15 (Madrid, Spain, 1997). 114

[230] V. Henrich and E. Hinrichs. *Gernedit - the germanet editing tool*. In *7th International Conference on Language Resources and Evaluation*, pp. 2228–2235 (Valletta, Malta, 2010). 114

[231] Q. Pradet, G. d. Chalendar, and J. Desormeaux Baguenier. *Wonef, an improved, expanded and evaluated automatic french translation of wordnet*. In *7th Global Wordnet Conference*, pp. 32–39 (Tartu, Estonia, 2014). 114

[232] P. Vossen, ed. *EuroWordNet: A multilingual database with lexical semantic networks* (Kluwer, Dodrecht, The Netherlands, 1998). 114

[233] R. Navigli and S. P. Ponzetto. *Babelnet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network*. Artificial Intelligence **193**, 217–250 (2012). 114

[234] T. R. Gruber. *Toward principles for the design of ontologies used for knowledge sharing?* International Journal of Human-Computer Studies **43**(5–6), 907–928 (1995). 115

[235] A. Pease, I. Niles, and J. Li. *The suggested upper merged ontology: A large ontology for the semantic web and its applications*. In *AAAI-2002 Workshop on Ontologies and the Semantic Web* (Alberta, Canada, 2002). 115

[236] H. Kucera and W. N. Francis. *Computational Analysis of Present-Day American English* (Technical Report, Brown University, Providence, RI, USA, 1997). 115

[237] J. H. Clear. *The british national corpus*. In G. P. Landow and P. Delany, eds., *The Digital Word: Text-Based Computing in the Humanities*, pp. 163–187 (MIT Press, Cambridge, MA, USA, 1993). 115

[238] G. A. Miller, C. Leacock, R. Tengi, and R. T. Bunker. *A semantic concordance*. In *Workshop on Human Language Technology*, pp. 303–308 (Cambridge, MA, USA, 1993). 115

[239] C. Leacock, G. Towell, and E. Voorhees. *Corpus-based statistical sense resolution*. In *Workshop on Human Language Technology*, pp. 260–265 (Cambridge, MA, USA, 1993). 115

[240] T. Chklovski and R. Mihalcea. *Building a sense tagged corpus with open mind word expert*. In *The ACL-02 Workshop on Word Sense Disambiguation: Recent Successes and Future Directions*, pp. 116–122 (Philadelphia, PA, USA, 2002). 115

[241] T. Brants and A. Franz. *Web 1t 5-gram version 1 ldc2006t13. dvd* (2006). 115

[242] E. Agirre and M. Stevenson. *Knowledge sources for wsd*. In E. Agirre and P. Edmonds, eds., *Word Sense Disambiguation*, pp. 217–251 (Springer, Dodrecht, The Netherlands, 2006). 115

[243] T. Pedersen, S. Patwardhan, and J. Michelizzi. *Wordnet::similarity: Measuring the relatedness of concepts*. In *Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 38–41 (Boston, MA, USA, 2004). 116

[244] T. Pedersen, S. Patwardhan, and J. Michelizzi. *Wordnet: Similarity - measuring the relatedness of concepts*. In *16th Conference on Innovative Applications of Artificial Intelligence*, pp. 1024–1025 (San Jose, CA, USA, 2004). 116

[245] M. Lesk. *Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from an ice cream cone*. In *5th International Conference on Systems Documentation*, pp. 24–26 (Toronto, Canada, 1986). 117

[246] S. Banerjee and T. Pedersen. *An adapted lesk algorithm for word sense disambiguation using wordnet*. In *3rd International Conference on Intelligent Text Processing and Computational Linguistics*, pp. 136–145 (Mexico City, Mexico, 2002). 117

[247] C. Leacock and M. Chodorow. *Combining local context and wordnet similarity for word sense identification*. In C. Fellbaum, ed., *WordNet: An Electronic Lexical Database (Language, Speech, and Communication)*, pp. 265–283 (MIT Press, Cambridge and MA, 1998). 117

[248] P. Resnik. *Using information content to evaluate semantic similarity in a taxonomy*. In *International Joint Conference on Artificial Intelligence*, pp. 448–453 (Montreal, Canada, 1995). 117

[249] Z. Wu and M. Palmer. *Verbs semantics and lexical selection*. In *32Nd Annual Meeting on Association for Computational Linguistics*, pp. 133–138 (Las Cruces, NM, USA, 1994). 117

[250] J. J. Jiang and D. W. Conrath. *Semantic similarity based on corpus statistics and lexical taxonomy*. In *10th International Conference on Research in Computational Linguistics*, pp. 19–33 (Taipei, Taiwan, 1997). 117

[251] D. Lin. *An information-theoretic definition of similarity*. In *15th International Conference on Machine Learning*, pp. 296–304 (Madison, WI, USA, 1998). 117

[252] G. Hirst and D. St-Onge. *Lexical chains as representation of context for the detection and correction malapropisms*. In C. Fellbaum, ed., *WordNet: An Electronic Lexical Database (Language, Speech, and Communication)*, pp. 285–304 (MIT Press, Cambridge, MA, USA, 1998). 117

[253] T. Pedersen. *Unsupervised corpus-based methods for wsd*. In E. Agirre and P. Edmonds, eds., *Word Sense Disambiguation*, pp. 133–166 (Springer, Dodrecht, The Netherlands, 2006). 117

[254] J. Han, M. Kamber, and J. Pei. *Data Mining: Concepts and Techniques* (Morgan Kaufmann, Waltham, MA, USA, 2012). 117

[255] M. Benson. *The structure of the collocational dictionary*. International Journal of Lexicography **2**(1), 1–14 (1989). 122

[256] V. Seretan. *Syntax-Based Collocation Extraction* (Springer, Dodrecht, The Netherlands, 2011). 122

[257] J. S. Justeson and S. M. Katz. *Technical terminology: some linguistic properties and an algorithm for identification in text*. Natural Language Engineering **1**, 9–27 (1995). 122

[258] A. Voutilainen. *Hand-crafted rules: Syntactic wordclass tagging*. In H. Halteren, ed., *Syntactic Wordclass Tagging*, pp. 217–246 (Springer, Dordrecht, The Netherlands, 1999). 122

[259] T. Brants. *Tnt: A statistical part-of-speech tagger*. In *6th Conference on Applied Natural Language Processing*, pp. 224–231 (Seattle, WA, USA, 2000). 122

[260] H. Stahl, J. Müller, and M. Lang. *An efficient top-down parsing algorithm for understanding speech by using stochastic syntactic and semantic models*. In *IEEE*

International Conference on Acoustics, Speech, and Signal Processing, vol. 1, pp. 397–400 (Atlanta, GA, USA, 1996). 122

[261] E. Charniak. *Statistical parsing with a context-free grammar and word statistics*. In *14th National Conference on Artificial Intelligence and Ninth Conference on Innovative Applications of Artificial Intelligence*, pp. 598–603 (Providence, RI, USA, 1997). 122

[262] F. Smadja. *Retrieving collocations from text: Xtract*. Computational Linguistics **19**(1), 143–177 (1993). 123

[263] D. Lin. *Automatic identification of non-compositional phrases*. In *37th Annual Meeting of the Association for Computational Linguistics on Computational Linguistics*, pp. 317–324 (College Park, MD, USA, 1999). 123

[264] N. Calzolari and R. Bindi. *Acquisition of lexical information from a large textual italian corpus*. In *13th Conference on Computational Linguistics*, pp. 54–59 (Helsinki, Finland, 1990). 123

[265] T. Baldwin and S. N. Kim. *Multiword expressions*. In N. Indurkhya and F. J. Damerau, eds., *Handbook of Natural Language Processing*, pp. 267–292 (Chapman & Hall/CRC, Boca Raton, FL, USA, 2010). 123

[266] G. Salton and C. Buckley. *Term-weighting approaches in automatic text retrieval*. Information Processing & Management **24**(5), 513–523 (1988). 125

[267] K. Spärck Jones. *A statistical interpretation of term specificity and its application in retrieval*. Journal of Documentation **28**, 11–21 (1972). 125

[268] J. Davis and M. Goadrich. *The relationship between precision-recall and roc curves*. In *23rd International Conference on Machine Learning*, pp. 233–240 (Pittsburgh, PA, USA, 2006). 131

[269] Z. Bellahsene and F. Duchateau. *Tuning for schema matching*. In Z. Bellahsene, A. Bonifati, and E. Rahm, eds., *Schema Matching and Mapping*, pp. 293–316 (Springer, Berlin, Heidelberg, Germany, 2011). 136, 139, 183, 202

[270] P. Shvaiko and J. Euzenat. *Ten challenges for ontology matching*. In *2008 Confederated International Conferences On the Move to Meaningful Internet Systems*, pp. 1164–1182 (Monterrey, Mexico, 2008). 136, 139, 183

[271] W. A. Gale, K. W. Church, and D. Yarowsky. *A method for disambiguating word senses in a large corpus*. Computers and the Humanities **26**(5–6), 415–439 (1992). 145, 204, 241

[272] H. T. Ng. *Getting serious about word sense disambiguation*. In *ACL SIGLEX Workshop on Tagging Text with Lexical Semantics: Why, What, and How?*, pp. 1–7 (Washington, DC, USA, 1997). 145, 204, 241

[273] A. Aleksovski. *Using background knowledge in ontology matching* (PhD thesis, Vrije Universiteit Amsterdam, Amsterdam, Netherlands, 2008). 145

[274] J. Madhavan, P. A. Bernstein, A. Doan, and A. Halevy. *Corpus-based schema matching*. In *21st International Conference on Data Engineering*, pp. 57–68 (Washington, DC, USA, 2005). 145

[275] B. Saha, I. Stanoi, and K. L. Clarkson. *Schema covering: a step towards enabling reuse in information integration*. In *26th IEEE International Conference on Data Engineering*, pp. 285–296 (Long Beach, CA, USA, 2010). 145

[276] V. Mascardi, A. Locoro, and P. Rosso. *Automatic ontology matching via upper ontologies: A systematic evaluation*. IEEE Transactions on Knowledge and Data Engineering **22**(5), 609–623 (2010). 145

[277] P. Jain, P. Z. Yeh, K. Verma, R. G. Vasquez, M. Damova, P. Hitzler, and A. P. Sheth. *Contextual ontology alignment of lod with an upper ontology: A case study with proton*. In *8th Extended Semantic Web Conference*, pp. 80–92 (Heraklion, Crete, Greece, 2011).

[278] M. Sabou, M. d'Aquin, and E. Motta. *Exploring the semantic web as background knowledge for ontology matching*. In S. Spaccapietra, J. Z. Pan, P. Thiran, T. Halpin,

S. Staab, V. Svatek, P. Shvaiko, and J. Roddick, eds., *Journal on Data Semantics XI*, pp. 156–190 (Springer, Berlin, Germany, 2008).

[279] J. Shamdasani, T. Hauer, P. Bloodsworth, A. Branson, M. Odeh, and R. McClatchey. *Semantic matching using the umls*. In *6th European Semantic Web Conference*, pp. 203–217 (Heraklion, Crete, Greece, 2009). 145

[280] R. Diestel. *Graph Theory* (Springer, Berlin, Germany, 2010). 153

[281] E. W. Dijkstra. *A note on two problems in connexion with graphs*. Numerische Mathematik **1**(1), 269–271 (1959). 155

[282] T. H. Cormen, C. Stein, R. L. Rivest, and C. E. Leiserson. *Introduction to Algorithms* (The MIT Press, Cambrigde, MA, USA, 2009). 155

[283] J. Vanhatalo, H. Völzer, and J. Koehler. *The refined process structure tree*. In *6th International Conference on Business Process Management*, pp. 100–115 (Milan, Italy, 2008). 157, 158

[284] R. E. Tarjan and J. Valdes. *Prime subprogram parsing of a program*. In *7th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pp. 95–105 (1980). 158

[285] J. E. Hopcroft and R. E. Tarjan. *Dividing a graph into triconnected components*. SIAM Journal on Computing **2**(3), 135–158 (1973). 158

[286] R. C. Johnson. *Efficient Program Analysis Using Dependence Flow Graphs* (PhD thesis, Cornell University, Ithaca, NY, USA, 1994). 158

[287] R. C. Johnson, D. Pearson, and K. Pingali. *The program structure tree: Computing control regions in linear time*. In *ACM SIGPLAN Conference on Programming Language Design and Implementation*, pp. 171–185 (Orlando, FL, USA, 1994). 158

[288] C. Ouyang, M. Dumas, A. H. ter Hofstede, and W. M. van der Aalst. *From bpmn process models to bpel web services*. In *2006 IEEE International Conference on Web Services*, pp. 285–292 (Chicago, IL, USA, 2006). 158

[289] C. Ouyang, M. Dumas, Aalst, Wil M. P. Van Der, Hofstede, Ter Arthur H. M., and J. Mendling. *From business process models to process-oriented software systems*. ACM Transactions on Software Engineering and Methodology **19**(1), 2:1–2:37 (2009). 158

[290] J. Vanhatalo, H. Völzer, and J. Koehler. *The refined process structure tree*. Data & Knowledge Engineering **68**(9), 793–818 (2009). 158

[291] G. Di Battista and R. Tamassia. *On-line maintenance of triconnected components with spqr-trees*. Algorithmica **15**(4), 302–318 (1996). 158

[292] C. Gutwenger and P. Mutzel. *A linear time implementation of spqr-trees*. In J. Marks, ed., *Graph Drawing*, pp. 77–90 (Springer, Berlin, Germany, 2001). 158

[293] A. Polyvyanyy, J. Vanhatalo, and H. Völzer. *Simplified computation and generalization of the refined process structure tree*. In *7th International Workshop on Web Services and Formal Methods*, pp. 25–41 (Hoboken, NJ, USA, 2011). 158

[294] A. Polyvyanyy, L. García-Bañuelos, and M. Dumas. *Structuring acyclic process models*. Information Systems **37**(6), 518–538 (2012). 159, 160

[295] W. M. P. Van Der Aalst, A. H. M. Ter Hofstede, B. Kiepuszewski, and A. P. Barros. *Workflow patterns*. Distributed Parallel Databases **14**(1), 5–51 (2003). 162

[296] N. Russell, A. H. M. ter Hofstede, W. M. P. van der Aalst, and N. Mulyar. *Workflow ControlFlow Patterns: A Revised View* (Technical Report, BPMcenter.org, Eindhoven, The Netherlands, 2006). 162

[297] W. M. P. van der Aalst, A. K. A. de Medeiros, and A. J. M. M. Weijters. *Process equivalence: Comparing two process models based on observed behavior*. In *4th International Conference on Business Process Management*, pp. 129–144 (Vienna, Austria, 2006). 163

[298] A. A. de Medeiros, W. van der Aalst, and A. Weijters. *Quantifying process equivalence based on observed behavior*. Data & Knowledge Engineering **64**, 55–74 (2008). 163

[299] R. Lipton. *The reachability problem requires exponential space*. Tech. rep., Technical Report, Yale University, New Haven, CT, USA (1976). 163

[300] A. Valmari. *The state explosion problem*. In W. Reisig and G. Rozenberg, eds., *Lectures on Petri Nets I: Basic Models*, pp. 429–528 (Springer, London, UK, 1998). 163

[301] M. Weidlich, J. Mendling, and M. Weske. *Efficient consistency measurement based on behavioral profiles of process models*. IEEE Transactions on Software Engineering **37**(3), 410–429 (2011). 163

[302] M. Weidlich. *Behavioral Profiles – A relational approach to behaviour consistency* (PhD thesis, Universität Potsdam, Potsdam, Germany, 2011). 163

[303] M. Weidlich, A. Polyvyanyy, J. Mendling, and M. Weske. *Efficient computation of causal behavioural profiles using structural decomposition*. In *31st International Conference on Applications and Theory of Petri Nets*, pp. 63–83 (Braga, Portugal, 2010). 164

[304] M. Weidlich, F. Elliger, and M. Weske. *Generalised computation of behavioural profiles based on petri-net unfoldings*. In *7th International Workshop on Web Services and Formal Methods*, pp. 101–115 (Hoboken, NJ, USA, 2011). 164

[305] M. Kunze, M. Weidlich, and M. Weske. *Behavioral Similarity – A Proper Metric*. In *9th International Conference on Business Process Management*, pp. 166–181 (2011). 164

[306] M. Weidlich, A. Polyvyanyy, J. Mendling, and M. Weske. *Causal behavioural profiles - efficient computation, applications, and evaluation*. Fundamenta Informaticae **113**(3-4), 399–435 (2011). 165

[307] C. E. Shannon. *A mathematical theory of communication*. Bell System Technical Journal **27**, 379–423 (1948). 169

[308] C. E. Shannon. *Prediction and entropy of printed english*. Bell System Technical Journal **30**, 50–64 (1951). 169

[309] S. Smirnov, M. Weidlich, J. Mendling, and M. Weske. *Action patterns in business process model repositories*. Computers in Industry **63**(2), 98–111 (2012). 179

[310] P. Schumacher and M. Minor. *Towards a trace index based workflow similarity function*. In C. Lutz and M. Thielscher, eds., *KI 2014: Advances in Artificial Intelligence: 37th Annual German Conference on AI, Stuttgart, Germany, September 22-26, 2014. Proceedings*, pp. 225–230 (Springer, 2014). 179

[311] R. Bergmann and Y. Gil. *Similarity assessment and efficient retrieval of semantic workflows*. Information Systems **40**, 115–127 (2014). 179

[312] D. Sánchez-Charles, V. Muntés-Mulero, J. Carmona, and M. Solé. *Process model comparison based on cophenetic distance*. In *Business Process Management (BPM) Forum, Rio de Janeiro, Brazil, September 18-22, 2016, Proceedings*, pp. 141–158 (Springer, 2016). 179

[313] C. Spearman. *The proof and measurement of association between two things*. The American Journal of Psychology **15**, 72–101 (1904). 181

[314] E. Peukert, J. Eberius, and E. Rahm. *Amc - a framework for modelling and comparing matching systems as matching processes*. In *IEEE 27th International Conference on Data Engineering*, pp. 1304–1307 (Hannover, Germany, 2011). 183, 199

[315] Y. Lee, M. Sayyadian, A. Doan, and A. S. Rosenthal. *etuner: Tuning schema matching software using synthetic scenarios*. VLDB Journal **16**(1), 97–122 (2007). 183

[316] V. Spiliopoulos and G. Vouros. *Synthesizing ontology alignment methods using the max-sum algorithm*. IEEE Transactions on Knowledge and Data Engineering **24**(5), 940–951 (2012). 183

[317] R. S. Michalski, J. G. Carbonell, and T. M. Mitchell. *Machine Learning: An Artificial Intelligence Approach* (Springer, Berlin, Germany, 1985). 197, 198

[318] M. Mohri, A. Rostamizadeh, and A. Talwalkar. *Foundations of Machine Learning* (MIT Press, Cambridge, MA, USA, 2012). 198

[319] P. Wegner. *Why interaction is more powerful than algorithms*. Communications of the ACM **40**(5), 80–91 (1997). 198

[320] P. Wegner and D. Goldin. *Interaction as a framework for modeling*. In G. Goos, J. Hartmanis, J. van Leeuwen, P. Chen, J. Akoka, H. Kangassalu, and B. Thalheim, eds., *Conceptual Modeling*, pp. 243–257 (Springer, Berlin, Germany, 1999). 198

[321] S. M. Falconer and M.-A. Storey. *A cognitive support framework for ontology mapping*. In *6th International Semantic Web Conference*, pp. 114–127 (Busan, Korea, 2007). 199

[322] S. M. Falconer. *Cognitive Support for Semi-automatic Ontology Mapping* (PhD thesis, University of Victoria, Victoria, BC, Canada, 2009). 199

[323] H.-H. Do. *Schema Matching and Mapping-Based Data Integration* (PhD thesis, Leipzig University, Leipzig, Germany, 2006). 199

[324] N. F. Noy and M. A. Musen. *The {PROMPT} suite: interactive tools for ontology merging and mapping*. International Journal of Human-Computer Studies **59**(6), 983–1024 (2003). 199

[325] R. McCann, W. Shen, and A. Doan. *Matching schemas in online communities: A web 2.0 approach*. In *24th IEEE International Conference on Data Engineering*, pp. 110–119 (Cancun, Mexico, 2008). 199

[326] K. Belhajjame, N. W. Paton, A. A. Fernandes, C. Hedeler, and S. M. Embury. *User feedback as a first class citizen in information integration systems*. In *5th Biennial Conference on Innovative Data Systems Research*, pp. 175–183 (Asilomar, CA, USA, 2011). 199

[327] S. R. Jeffery, M. J. Franklin, and A. Y. Halevy. *Pay-as-you-go user feedback for dataspace systems*. In *28th ACM SIGMOD/PODS Conference*, pp. 847–860 (2008). 199

[328] H.-H. Do and E. Rahm. *Matching large schemas: Approaches and evaluation*. Information Systems **32**(6), 857–885 (2007). 199

[329] S. Duan, A. Fokoue, and K. Srinivas. *One size does not fit all: Customizing ontology alignment using user feedback*. In *9th International Semantic Web Conference*, pp. 177–192 (2010). 200

[330] I. F. Cruz, F. P. Antonelli, and C. Stroe. *Agreementmaker: Efficient matching for large real-world schemas and ontologies*. Proceedings of the VLDB Endowment **2**(2), 1586–1589 (2009). 200

[331] I. N. Bronshtein, K. A. Semendyayev, G. Musiol, and H. Muehlig. *Handbook of Mathematics* (Springer, Berlin, Germany, 2007). 213

[332] S. Wasserman and K. Faust. *Social Network Analysis: Methods and Applications* (Cambridge University Press, Cambridge, England, 1994). 214

[333] R. D. Luce and A. D. Perry. *A method of matrix analysis of group structure*. Psychometrika **14**(2), 95–116 (1949). 214

[334] P. W. Holland and S. Leinhardt. *Transitivity in structural models of small groups*. Small Group Research **2**(2), 107–124 (1971). 214

[335] D. J. Watts and S. H. Strogatz. *Collective dynamics of'small-world'networks*. Nature **393**(6684), 409–410 (1998). 218

[336] M. Kaiser. *Mean clustering coefficients: the role of isolated nodes and leafs on clustering measures for small-world networks*. New Journal of Physics **10**(8) (2008). 218

[337] D. T. Campbell and J. C. Stanley. *Experimental and Quasi-Experimental Designs for Research* (Houghton Mifflin, Boston, MA, USA, 1963). 244

[338] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén. *Experimentation in Software Engineering: An Introduction* (Springer, Berlin, Germany, 2012). 244, 245

[339] T. D. Cook and D. T. Campbell. *Quasi-experimentation: Design and Analysis Issues for Field Settings* (Houghton Mifflin, Boston, MA, USA, 1979). 244

[340] H. Zhuge. *A process matching approach for flexible workflow process reuse*. Information and Software Technology **44**(8), 445–450 (2002). 250

[341] A. Wombacher, P. Fankhauser, B. Mahleko, and E. Neuhold. *Matchmaking for business processes*. In *IEEE International Conference on E-Commerce*, pp. 7–11 (Newport Beach, CA, USA, 2003). 250

[342] A. Wombacher, P. Fankhauser, B. Mahleko, and E. Neuhold. *Matchmaking for business processes based on choreographies*. In *IEEE International Conference on e-Technology, e-Commerce and e-Service*, pp. 359–368 (Taipei, Taiwan, 2004). 250

[343] P. Suwannopas and T. Senivongse. *Discovering semantic web services with process specifications*. In *6th International IFIP Conference on Distributed Applications and Interoperable Systems*, pp. 113–127 (Bologna, Italy, 2006). 250

[344] L. Lei, Z. Duan, and B. Yu. *Semantic matching of web services for collaborative business processes*. In *10th International Conference on Computer Supported Cooperative Work in Design*, pp. 479–488 (Nanjing, China, 2007). 250

[345] D. Deutch and T. Milo. *Evaluating top-k queries over business processes*. In *25th International Conference on Data Engineering*, pp. 1195–1198 (Shanghai, China, 2009). 250

[346] J. Gao and L. Zhang. *On measuring semantic similarity of business process models*. In *5th International Conference on Interoperability for Enterprise Software and Applications*, pp. 289–293 (Beijing, China, 2009). 250

[347] J. J. Jung. *Semantic business process integration based on ontology alignment*. Expert Systems with Applications **36**(8), 11013–11020 (2009). 250

[348] J. Zhu and H. K. Pung. *Process matching: A structural approach for business process search*. In *Computation World: Future Computing, Service Computation, Cognitive, Adaptive, Content, Patterns*, pp. 227–232 (Athens, Greece, 2009). 250

[349] V. Gacitua-Decar and C. Pahl. *Towards reuse of business processes patterns to design services*. In W. Binder and S. Dustdar, eds., *Emerging Web Services Technology Volume III*, pp. 15–36 (Birkhäuser Basel, Basel, Switzerland, 2010). 250

[350] A. Gater, D. Grigori, and M. Bouzeghoub. *Owl-s process model matchmaking*. In *2010 IEEE International Conference on Web Services*, pp. 640–641 (Miami, FL, USA, 2010). 250

[351] G. Kim and Y. Suhh. *Ontology-based semantic matching for business process management*. SIGMIS Database **41**(4), 98–118 (2010). 250

[352] S. Sakr and A. Awad. *A framework for querying graph-based business process models*. In *19th International Conference on World Wide Web*, pp. 1297–1300 (Raleigh, NC, USA, 2010). 250

[353] P. Tonella and C. Di Francescomarino. *Supporting ontology-based semantic annotation of business processes with automated suggestions*. International Journal of Information System Modeling and Design **1**(2), 59–84 (2010). 250

[354] R. M. Dijkman, B. Gfeller, J. Küster, and H. Völzer. *Identifying refactoring opportunities in process model repositories*. Information and Software Technology **53**(9), 937–948 (2011). 250

[355] S. Abbas and H. Seba. *A module-based approach for structural matching of process models*. In *IEEE 5th International Conference on Service-Oriented Computing and Applications*, pp. 1–8 (Taipei, Taiwan, 2012). 251

[356] Y. Belhoul, S. Yahiaoui, M. Haddad, A. Gater, H. Kheddouci, and M. Bouzeghoub. *A graph approach for enhancing process models matchmaking*. In *IEEE 12th International Conference on Services Computing*, pp. 773–776 (New York, NY, USA, 2015). 251

[357] M. L. Sebu and H. Ciocârlie. *Business process similarity metric supporting one-to-many relationship*. In *IEEE 10th Jubilee International Symposium on Applied*

Computational Intelligence and Informatics, pp. 429–435 (Timisoara, Romania, 2015). 251