Analyzing Control Flow Information to Improve the Effectiveness of Process Model Matching Techniques

Christopher Klinkmüller[a,b,*], Ingo Weber[b,c]

[a] *Department of Computing, Macquarie University, NSW 2109, Australia*
[b] *Data61, CSIRO, Locked Bag 9013, Alexandria, NSW 1435, Australia*
[c] *University of New South Wales, UNSW Sydney, NSW 2052, Australia*

Abstract

Process model matchers automatically identify activities that represent similar functionality in different process models. As such, they support various tasks in business process management including model collection management and process design. Yet, comparative evaluations revealed that state-of-the-art matchers fall short of offering high performance across varied datasets. To facilitate the development of more effective matchers, we systematically study, if and how the analysis of *control flow information* in process models can contribute to the matching process. In particular, we empirically examine the *validity of analysis options* and use our findings to automate the *adaptation of matcher configurations* to model collections.

Keywords: BPM, process similarity, process model matching

1. Introduction

Many organizations employ process models as a tool to document, develop, evaluate, and automate processes. Over the course of time, model

*Corresponding author
Email addresses: christopher.klinkmuller@data61.csiro.au (Christopher Klinkmüller), ingo.weber@data61.csiro.au (Ingo Weber)

collections can grow to sizes of thousands of models, e.g., the China railway company has more than 200,000 process models [1]. Large collection sizes combined with different notations, vocabularies, and abstraction levels lead to blurred relationships between models, and in effect decrease their utility.

In such scenarios, *process model matchers* can provide support by automatically detecting correspondences between process models, i.e., activities that constitute similar functionality. Whereas *purely label-based* matchers, e.g., those in [2, 3], identify correspondences by solely comparing the natural language descriptions of the activities, *control flow considering* matchers, e.g., in [4, 5, 6, 7, 8], additionally exploit *control flow information* by analyzing ordering constraints between activities stemming from the structure and execution semantics of the models. Despite the attention that the development of matchers has gained, contests with comparative evaluations [9, 10] showed that existing matchers yield an overall low effectiveness, i.e., their results contain many irrelevant and only a few true correspondences.

To implement the comparison of the activity descriptions, an extensive body of knowledge from natural language processing [11], information retrieval [12], or schema and ontology matching [13, 14] is available. In contrast, control flow information is a unique feature of process models. Although many matchers consider such information, there is only *limited evidence* that this information contributes to the detection of correspondences[1]. That is because prior research mainly focused on evaluating the matchers' overall effectiveness, but did *not* study the contribution of their components.

[1] We here summarize the findings of a literature review that we conducted. We discuss this review in more detail in the associated online appendix (https://arxiv.org/abs/1707.01089).

Some works, e.g., [3, 15] and the matching contests [9, 10], perform black box evaluations. While such an approach allows to assess and compare the effectiveness of matchers, it does not permit the examination of the influence of the matchers' components. For example, the matcher in [15] comprises components to compute label similarities, investigate the activities' graph neighborhood, detect fragments, and check the consistency. Clearly, the reported overall effectiveness of .73 allows no insights into the contribution of each component. Other works, e.g., in [4, 7, 6, 8], compare the effectiveness of matcher variants. However, as discussed in [16, 17] such a comparison needs to be treated with care, because without further statistical analyses differences might have been observed simply by chance – especially as the reported differences are rather small, e.g., the effectiveness in [5] differs by $\approx .06$ and in [7] by $\approx .05$. Moreover, we even find contradicting conclusions, e.g., consistency checks improve the effectiveness in [6], but reduce it in [7]. In addition to these limitations, the contribution of control flow information is also questioned by the fact that, with one exception, the top matchers on all datasets in the matching contests [9, 10] solely exploit labels.

The low matcher effectiveness and the limited validity of the use of control flow information lead us to the research question: *How can control flow information be used to improve the results of process model matchers?* To answer this question, we systematically study *control flow propositions* which constitute cause-effect relationships regarding the use of control flow information. First, we introduce such propositions, relate them to existing matchers, and validate them based on empirical analyses. The result, a set of *(in-)validated control flow propositions,* can be utilized to guide the design of more effective matchers. Second, based on the validated propositions we address a central challenge in the matcher development: *automatically*

optimizing their configuration [18] to suit varying contexts. To this end and in contrast to prior work, our methodology is based on a clear separation of development and evaluation data, as per [19]: *development data* was used to study the propositions and the automatic configuration, whereas we relied on *evaluation data* to examine the general validity of our results.

This paper extends our own prior work. First, in [20] we examined the control flow based comparison of activities. By studying further propositions we here compete this analysis which is the only proposition analysis in the process model matching literature so far (see our literature review[1]). Second, we submitted an early version of our self-configuring matcher to the matching contest in 2015 [10]. Instead of relying on three pre-defined configurations of a variant of our label-based matcher from [2], the extended version searches a configuration space that, depending on the model collection, can contain thousands of configurations. Furthermore, we compare the automated configuration to a semi-manual approach and study the selection of matchers. Finally, we use data that was not used in prior work.

This article is organized as follows. First, Section 2 introduces basic terminology and the research design. While Section 3 outlines and studies the control flow propositions, Section 4 presents and evaluates the automatic matcher configuration. Afterwards, Section 5 discusses the findings and Section 6 summarizes related work. Finally, Section 7 concludes the paper.

2. Background

In this section, we introduce basic terminology and the research design.

2.1. Problem Illustration and Terminology

We adopt Dijkman et al.'s view of process models as graphs [5]: nodes are assigned a type (activity, gateway, event, etc.), and might have a label; edges depict the control flow by connecting nodes. Here, we apply common normalization operations [13] to unify the syntactical format of the labels.

Definition 1 (Process model). Let $\mathcal{L}$ be a set of labels and $\mathcal{T}$ be a set of types. A process model p is a tuple (N, E, λ, τ, A), in which

- N is the set of nodes;
- $E \subseteq N \times N$ is the set of edges;
- $\lambda : N \to \mathcal{L}$ is a function that maps nodes to normalized labels;
- $\tau : N \to \mathcal{T}$ is a function that assigns each node to one type; and
- $A = \{a | a \in N \wedge \tau(a) = activity\}$ is the set of activities.

Following terminology from ontology matching [13], the comparison of two process models for identifying corresponding activities that constitute similar functionality is referred to as the *matching process*. Software that automates this process is called a *matching technique* or *matcher*. Given two process models, the matching process generates an *alignment* which comprises corresponding activities. Here, correspondences are bidirectional and thus the result of a matching process is independent of the model order. The alignment is a set of *1:1-correspondences*. Such a 1:1-correspondence (or just correspondence) is an activity pair with exactly one activity from each of the two process models. Yet, due to different levels of abstraction there might be complex correspondences: *m:n-correspondences* can exist between

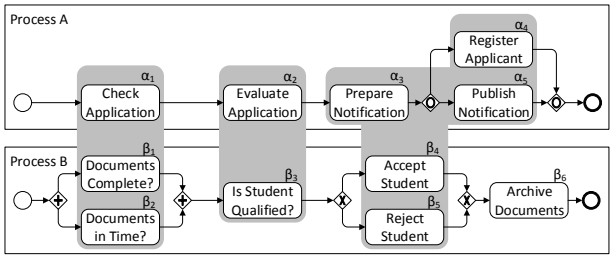

Figure 1: An example for a process model alignment

corresponding sets of activities (also called *fragments*); if one set only consists of one activity, it is referred to as a *1:n-correspondence*. Such complex correspondences are represented as the set of all 1:1-correspondences that contain these activities. The matching process can be *configured* via *parameters*, e.g., a threshold, and *resources*, e.g., a domain-specific dictionary.

Definition 2 (Matching process, Alignment, Correspondence). For two process models $p = (N,E,\lambda,\tau,A)$, $p' = (N', E', \lambda', \tau', A')$ and the sets of parameters π and resources r, the matching process is defined as a function

$$\mathscr{A} = match(p, p', \pi, r)$$

where $\mathscr{A} \subseteq A \times A'$ is an alignment over p and p'. Each $c = (a, a')$ with $c \in \mathscr{A}$ is called a correspondence. Complex correspondences between activity sets (A_s, A'_s) with $A_s \subseteq A$ and $A'_s \subseteq A'$ are expressed as the set of all correspondences between these activity sets, i.e., $\forall a_s \in A_s, a'_s \in A'_s : (a_s, a'_s) \in \mathscr{A}$. Moreover, we require the matching process to be independent of the order of the process models, i.e., $\mathscr{A} = match(p, p', \pi, r) \Rightarrow \mathscr{A}^{-1} = match(p', p, \pi, r)$.

Figure 1 shows an example alignment. Here, both models outline a process of formally assessing and deciding on a student application. Hence, the alignment $\mathscr{A} = \{(\alpha_1, \beta_1), (\alpha_1, \beta_2), (\alpha_2, \beta_3), (\alpha_3, \beta_4), (\alpha_3, \beta_5), (\alpha_4, \beta_4), (\alpha_4, \beta_5), (\alpha_5, \beta_4), (\alpha_5, \beta_5)\}$ contains a 1:1-correspondence between α_2 and β_3, a

1:n-correspondence between α_1 and β_1, β_2, as well as an m:n-correspondences between $\alpha_3, \alpha_4, \alpha_5$ and β_4, β_5. Further, β_6 has no corresponding activity.

The major goal of matcher development is the maximization of the *effectiveness* (also referred to as *performance*) which can be assessed by applying the matcher to a set of model pairs and comparing its results to the truly existing correspondences. Each correspondence detected by the matcher is a *true positive* (TP); all activity pairs that are suggested by the matcher, but do not correspond are *false positives* (FP); and correspondences missed by the matcher are *false negatives* (FN). Based on this classification, our main indicator is the *f-measure* ($F = 2 \cdot \frac{P \cdot R}{P+R}$) – a well-known measure from information retrieval [12]. Here, the *precision* ($P = \frac{TP}{TP+FP}$) is the share of correct correspondences among the proposed ones, and the *recall* ($R = \frac{TP}{TP+FN}$) is the share of correctly identified correspondences.

Regarding the example alignment, the correspondences $\mathscr{A}_m = \{(\alpha_1, \beta_1), (\alpha_2, \beta_2), (\alpha_3, \beta_3), (\alpha_4, \beta_4), (\alpha_5, \beta_5)\}$ proposed by a matcher comprise three true positives, six false negatives, and two false positives. Thus, the matcher identifies 33% of the existing correspondences ($R = .\overline{3}$), 60% of its suggestions are correct ($P = .6$), and the overall f-measure is $F = .429$.

When viewed at the level of a model collection, there are two options to calculate F, P, and R: at the *micro level* P_μ, R_μ, F_μ are computed over the union of correspondences from all model pairs, while at the macro level P_M, R_M, F_M are determined per model pair and then averaged. At the macro level, measures might be distorted by differences in the number of possible correspondences per model pair. Thus, we focus on the micro-level.

Table 1: Descriptive statistics of the development and the evaluation datasets

	Models			*Activities*			*Correspondences*				*Activity*
	#	*Pairs*	*Language*	*Min*	*Max*	*Ø*	*#*	*1:1*	*1:n*	*m:n*	*Pairs*
BR	9	36	English	9	25	19.3	584	156	95	13	13,358
UA	9	36	English	13	48	27.6	531	251	77	1	26,853
AM	72	36	English	1	43	9.3	222	137	16	3	4.559
CP	9	36	German	3	22	7.4	375	27	53	25	1,866

2.2. Empirical Research Methodology

To examine the validity of control flow propositions, we conducted a series of empirical analyses. In this regard, the replication of propositions from the literature was hindered because prior research only provided high-level descriptions (e.g., [9, 10]), referred to the model level rather than the activity level (e.g., [21]), or focused a certain modeling notation (e.g., [22]). Moreover, source code is generally not accessible. To nevertheless assure that the considered propositions and thus our analyses are aligned with prior work, we oriented the propositions towards a classification of existing propositions (see Section 3 and the online appendix[1] for more details). Moreover, we enable replicability and extensibility of our analyses by releasing the source code[2] and by relying on three publicly available datasets[3,4].

We first examined the propositions on two *development datasets*. Universal applicability of a proposition mandates that each specific instantiation has a positive contribution on any dataset. While that cannot be proven empirically, we can disprove it with a counter example. Therefore, if we find empirical evidence indicating that a certain proposition does not hold,

[2] https://bitbucket.org/cklinkmueller/control_flow_analysis

[3] https://ai.wu.ac.at/emisa2015/contest.php, accessed: 18/04/2017

[4] http://www.henrikleopold.com/downloads/, accessed: 18/04/2017

this invalidates the universal applicability of this proposition and we dismiss it. In contrast, if a proposition is not invalidated we *cannot* conclude that it is universally applicable or generally valid – we can only collect evidence towards that. Hence, we extended the analysis of those propositions by incorporating them into a matcher. This matcher was then evaluated and analyzed with regard to the development datasets and two additional *evaluation datasets*.

Each development and evaluation dataset contains 36 process model pairs and a gold standard which captures true correspondences determined by experts. Table 1 summarizes the datasets. The development datasets, *birth registration* (BR) and the *university admission* (UA), were introduced by other researches in [6, 9] and also used for matcher evaluation in [2, 10, 20]. The BR dataset refers to the registration of newborn children in four different countries, and the UA dataset contains models of the application procedures for students at nine German universities. Each dataset comprises nine Petri net models with English labels. The comparison of each process model with all other models results in 36 distinct pairs per collection. According to [6, 9], for each dataset two experts separately created a gold standard and a third expert resolved differences. Note that in the second matching contest [10] another gold standard for UA was introduced, which contains only correspondences in which the assigned roles also correspond. As our focus here is on the control flow and not the organizational perspective, we utilize the original gold standards from [6, 9].

The two evaluation datasets were developed in the scope of our work. The *Asset Management* (AM) dataset is based on the SAP reference model which was used in related research from the field of business process management, e.g., [23]. Out of the 604 EPC models, 72 distinct models dealing with

different aspects from finance and accounting were selected and arranged in 36 model pairs. We made this dataset available to the process model matching contest 2015 [10]. The second evaluation dataset stems from a commercial consolidation project (CP) at a German university, where processes of independent faculties within the university were unified. The dataset contains nine BPMN process models (and thus 36 model pairs) concerned with managing examination results.[5] The AM dataset has English and the CP dataset German labels. Similar to the BR and UA datasets two experts independently created gold standards. Then, automatically identified differences were resolved in a discussion.

3. Validation of Control Flow Propositions

In this section, we study the use of control flow information based on a classification of existing control flow propositions. In line with prior research, we distinguish three *use cases*: *compare activities*, *detect candidates for complex correspondences*, and *check consistency*. Moreover, per use case we consider up to three types of control flow *encodings*: paths in the process *graph*, properties of nested *fragment hierarchies*, or *execution semantics*. More details on how this classification reflects existing propositions are provided in the online appendix[1]. Here, we focus on analyzing specific propositions for each use case. Moreover, to guarantee an unbiased assessment of the propositions, we abstract from specific approaches to label comparison.

[5] Due to contractual obligations, we cannot release the CP dataset publicly.

3.1. Compare Activities

For the first use case, we summarize our earlier analysis [20], in which we utilized the development datasets to study the comparison of activities. That is, we assessed the discriminative power of various similarity scores where each score relies on a particular control flow property y, e.g., distance from the start node, that is represented by a *property function* $\pi_y : A \to [0, 1]$. Such a property function returns a numerical value for each activity in a model, and is linearly normalized to the interval $[0, 1]$. To establish similarity between two activities based on y, their values for π_y are compared by a *control flow similarity function* $\sigma.\pi_y : A \times A \to [0, 1]$ where a value of 1 indicates equality, 0 total dissimilarity, and values in between degrees of similarity. We generically define the similarity score of an activity pair for any given π_y as 1 minus the absolute difference of the relative property values: $\sigma.\pi_y(a, a') = 1 - |\pi_y(a) - \pi_y(a')|$.

For the graph encoding we considered two property functions that measure the position of activities in a model. The *start distance* π_{sta} (*end distance* π_{end}) is based on the minimum number of activities on any path connecting any start (end) node to the activity. Moreover, the *graph neighborhood* π_{nei} of an activity a is the number of activities that are connected to a via an outgoing or incoming path containing no other activity.

Similarly, we defined a position and a neighborhood property function for the hierarchy encoding, both based on the *refined process structure tree* (RPST) [24] that maps models to fragment hierarchies. The RPST can only be computed for models with one start and one end node. Yet, models with multiple start or exit nodes can be transformed into such models without changing the structural relationships between the model elements [25]. Both functions first determine the lowest non-trivial fragment fr that contains the

activity. Then, the *depth* π_{dep} is the depth of fr and the *number of siblings* π_{sib} is the number of activities in fr.

Finally, we considered the execution semantics in terms of the behavioral profile [26] of a process model, that captures which activities are carried out in sequence, in parallel, or mutually exclusive. The *sequence* property $\pi_{\rightsquigarrow}$ is based on the number of activities that are executed before a certain activity. The *parallel* $\pi_{\parallel}$ and the *alternative* $\pi_{\times}$ property yield the number of activities that are executed in parallel with or alternatively to the activity. Note that a reliable analysis of the execution semantics in general and the behavioral profiles in particular requires models to be *sound* [26]. Thus, we could only examine six model pairs on UA where the majority of models is not sound.

For each similarity score we assessed its discriminative power by comparing the value distributions yielded for non-corresponding and corresponding activity pairs on both development datasets. Figure 2 shows the value distributions for π_{sta} and π_{dep} on BR, which are representative for all scores. Here, the large overlaps of the distributions indicate that limiting the search to activity pairs with certain similarity values for one of the scores will still yield many non-corresponding pairs and rule out many correspondences. Based on a Kolmogorov-Smirnov test [27] over the development datasets and all 8 properties we found that the value distributions only significantly differ for π_{sta} and $\pi_{\rightsquigarrow}$ on both datasets. Yet, the information gain [28] as a measure for the goodness of classification showed that the discriminative power, and hence the utility for matching, of these properties is very low.

3.2. Detect Candidates for Complex Correspondences

To support the identification of complex correspondences, control flow information is often used to detect candidates for complex correspondences,

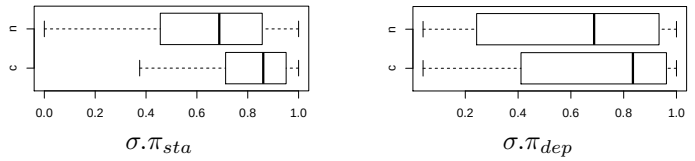

Figure 2: Box plots for non-corresponding (n) and corresponding (c) activity pairs on BR

in terms of connected activity sets. For example, the matcher in [4] derives such sets from fragment hierarchies and the approach in [5] requires candidates to be connected sub-graphs. Here, it is implicitly assumed that the control flow and the functional dependencies in such fragments are related.

To examine the respective propositions, we derived all distinct corresponding activity sets from the complex correspondences in the development datasets. In total, the gold standard of BR contains 57 sets, and UA has 53. Next, we intersected these sets with the fragments from the models' RPSTs. For BR, this intersection contains eight fragments; for UA only one. This shows that solely relying on fragment hierarchies will rule out a large number of actual complex correspondences. Next, we analyzed how many sets are connected sub-graphs. Two activities were seen as connected, if there is a path in the undirected version of the process model that connects both activities and contains no other activity. Note that these sub-graph sets are supersets of the RPST fragments. The intersections with the gold standard sets contain 50 sub-graphs for BR, and 35 for UA. While this is a clear improvement, 12% of the corresponding activity sets on BR and even 34% on UA can still not be derived from the connected sub-graphs.

We also investigated how precise the fragment detection methods are. That is, we determined the total number of RPST fragments as well as connected and arbitrary sub-graphs as shown in Table 2. On BR 4% of the RPST fragments are actually part of a complex correspondence and on UA

Table 2: The number of potential candidates and their overlap with the gold standards

	BR		*UA*	
Fragment Type	*Overlap*	*Potential*	*Overlap*	*Potential*
RPST fragments	8	211	1	229
Connected sub-graphs	50	125,321	35	5,535,807,993
Arbitrary sub-graphs	57	52,969,801	53	281,760,613,146,367

.4%. Considering connected sub-graphs deteriorates the situation: less than .004% of the connected sub-graphs participate in complex correspondences. Lastly, for all sub-graphs the number of potential candidates explodes.

These results indicate that imposing control flow restrictions on complex correspondences yields unreliable results. While not all potential candidates can be found, a large number of irrelevant candidates needs to be considered.

3.3. Check Consistency

The third use case focuses on checking, if control flow dependencies between activities in a model reflect those of the corresponding activities in the other model. A common strategy is to measure the consistency in terms of a graph edit distance [5, 7]. Such a distance measures the number of operations needed to transform one model into the other by inserting, deleting, or substituting nodes and edges. For example, in order to transform process A into process B in Figure 1, an activity "archiving of documents" needs to be inserted between the end node and the inclusive block in process A. As graph edit distances account for unmatched nodes, they focus the consistency of process models, but not of alignments.

Similar to [6], we thus define consistency based on the ordering of the activities in the alignments and consequently focus on the position properties: π_{sta}, π_{end}, π_{dep}, and $\pi_{\rightsquigarrow}$, for which we define the *order relationship*

score δ_y. Given a property π_y and one alignment per model pair, we first compute the order relationship score for each alignment: the relative frequency of distinct correspondence pairs $((a_1, a_1'), (a_2, a_2'))$ in an alignment for which the activity positions are consistent, i.e., for which we observe that $\pi_y(a_1) - \pi_y(a_2)$ and $\pi_y(a_1') - \pi_y(a_2')$ have the same sign. Then, the overall score is the average of the scores over all alignments in the set.

Definition 3 (Order relationship score). Given a set of alignments $\mathscr{A}^*$ and a position property $\pi : A \to [0, 1]$ the *order relationship score* δ is defined as:

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

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

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

To analyze the validity of δ, we computed the scores for the gold standard alignments of the development datasets. Per dataset and π_y function this results in one value, δ_y^{GS} in Table 3. On UA, where 83% of the model pairs contain models that are not sound, we refrained from evaluating $\delta_{\rightsquigarrow}$. The overall high values, especially for $\delta_{\rightsquigarrow}$ and δ_{sta}, give evidence in favor of δ.

Next, we refined our analysis and examined if high values for δ_y are a distinctive feature of the true alignments. To this end, we simulated a diverse range of alignments and assessed the correlation between their difference to the true alignments in terms of the micro f-measure and their order relationship scores. For both development datasets we randomly generated 1,000 sets of alignments, each with one alignment per model pair. To simulate a

diverse range of alignments, we controlled the generation such that their micro f-measures were uniformly distributed over the interval $[0, 1]$. For each set of alignments we computed δ_y as well as Spearman's rank correlation coefficient ρ for all combinations of variables as presented in Table 3.

The correlation coefficients show a strong positive correlation between all variables on both datasets. As the findings are significant for all variable pairs ($p \ll .001$), we conclude that objectively true alignments tend to preserve the control flow relationships between correspondences. Because the scores are also strongly correlated among themselves, only one of them should be considered as a consistency measure. $\delta_{\rightsquigarrow}$ and δ_{sta} achieve the highest scores and the strongest correlation to F_μ. However, the applicability of $\delta_{\rightsquigarrow}$ is limited. Hence, we propose to rely on δ_{sta}; it has the strongest discriminative power among the three remaining scores as illustrated by the scatter plots in Figure 3. That is, δ_{sta} has the largest range of observed score values. Thus, it best separates alignments with a low score and low F_μ from alignments with higher values for the score and the f-measure.

It is important to note that this result does not invalidate the findings from Section 3.1 where activities are compared with regard to their position in isolation. In contrast, we here examined activities in the con-

Table 3: Analysis results for the order relationship scores on the development datasets

	BR						*UA*					
	Correlation Coefficients (ρ)						*Correlation Coefficients (ρ)*					
	δ^{GS}	F_μ	δ_{sta}	δ_{end}	δ_{dep}	$\delta_{\rightsquigarrow}$	δ^{GS}	F_μ	δ_{sta}	δ_{end}	δ_{dep}	$\delta_{\rightsquigarrow}$
F_μ	-	-	.97	.95	.95	.97	-	-	.97	.97	.88	-
δ_{sta}	.92	.97	-	.96	.96	1.0	.93	.97	-	.98	.91	-
δ_{end}	.81	.95	.96	-	.94	.96	.89	.97	.98	-	.89	-
δ_{dep}	.85	.95	.96	.94	-	.96	.81	.88	.91	.89	-	-
$\delta_{\rightsquigarrow}$	.93	.97	1.0	.96	.96	-	-	-	-	-	-	-

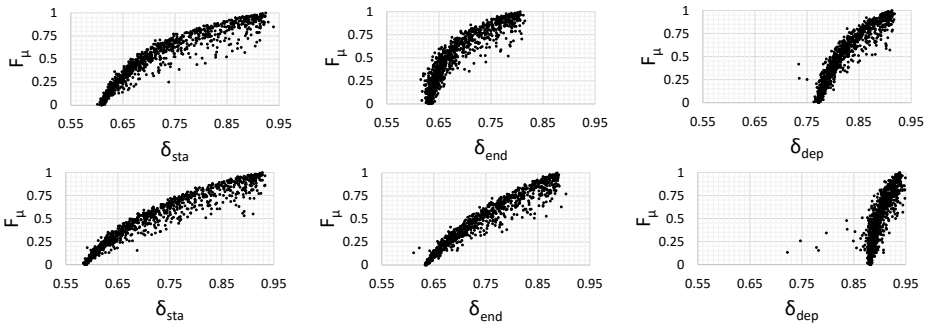

Figure 3: Scatter plots for δ vs $F_\mu$on BR (upper row) and UA (lower row)

text of other activity pairs. To illustrate the difference, we refer to the two sequential process models $\langle a, b, c\rangle$ and $\langle d', a', c'\rangle$ with the alignment $\mathscr{A} = \{(a, a'), (c, c')\}$. Here, the start-based similarity value of the correspondence (a, a') is equal to that of the non-corresponding pair (b, d'), but different from that of the other correspondence $(c, c)'$, i.e., $\sigma.\pi_{sta}(a, a') = \sigma.\pi_{sta}(b, d') = .5 \neq \sigma.\pi_{sta}(c, c') = 1$. Yet, (a, a') and (c, c') are consistent with regard to their position $(\pi_{sta}(a) < \pi_{sta}(c) \wedge \pi_{sta}(a') < \pi_{sta}(c'))$, whereas (a, a') and (b, d') are not $(\pi_{sta}(a) < \pi_{sta}(b) \wedge \pi_{sta}(a') > \pi_{sta}(b'))$.

4. Automatic Matcher Configuration

In prior work, e.g., in [5, 6, 7], consistency checks are used to refine alignments. That is, an alignment is constructed by iteratively adding activity pairs to the alignment until the consistency drops below a satisfactory level. Yet, in our experiments we found that this strategy is prone to errors, because (i) the alignment will contain false positives that distort the consistency checks and (ii) not all correspondences are consistent ($\delta^{GS}_{sta} \approx .92$ in Table 3), but some non-corresponding activity pairs are. With that in mind, we instead use consistency checks for *automatic matcher configuration*, i.e., the selection of specific parameter values and resources. That means, we use the order relationship score as an oracle to guide our search for a good

configuration. We also aim to mitigate the risks of the above problems by measuring the consistency of sets of alignments rather than of individual activity pairs. The approach in [8] is similar in that it pursues the idea of selecting matchers based on their predicted performance. However, in [8] selection is based on the input, i.e., features of process models and activity pairs; in contrast, we select configurations of matchers based on their *output*, specifically the consistency of their results. Moreover, unlike our work, in [8] control flow is considered only at the process model level, not the activity level. Finally, we critically evaluate our technique on four datasets.

4.1. The Order Preserving Bag-of-Words Technique

At the heart of our self-configuring matcher is our *bag-of-words technique* (BOT) [2]. To match two process models $p = (N, E, \lambda, \tau, A)$, $p' = (N', E', \lambda', \tau', A')$, it iterates over the set of activity pairs $A \times A'$. If the label similarity $\sigma_\lambda : A^2 \to [0, 1]$ for an activity pair is higher than or equal to a predefined threshold $\theta \in [0, 1]$, the pair is suggested as a correspondence. As correspondences are bidirectional, see Definition 2, we consider a symmetric label similarity. In particular, it yields a value of 1 for activity pairs with equal labels ($\lambda(a) = \lambda'(a')$). Further, any activity having an equally labeled counterpart in the other process is considered totally dissimilar to all other activities and a value of 0 is assigned to the respective pairs. For all remaining activity pairs we compute the bag-of-words similarity $\sigma_b : A^2 \to [0, 1]$.

The bag-of-words similarity splits each activity label into the set of individual words and removes all stop words (like "the") which are function words with little semantic meaning. Given two activities a, a' with the word sets Ω, Ω', it determines a similarity score $\sigma_\omega : \mathcal{W}^2 \to [0, 1]$ for each word pair in $\Omega \times \Omega'$. In this regard, it reduces the words to their stem, using

Porter's algorithm [29], determines the maximum similarity score per word, and combines these maximum scores into a single similarity score for the activity pair following one of two options. First, the average of the maximum scores yielded for all words ($\Omega \cup \Omega'$) is returned. Second, *pruning* can be activated to unify the label specificity in cases where one label contains more words than the other ($|\Omega_l| > |\Omega_s|$). If pruning is enabled, Ω_l is reduced to Ω_l^r by selecting the $|\Omega_s|$ words with the highest maximum score in Ω_l. Then, the average of the maximum scores of the words in $\Omega_l^r \cup \Omega_s$ is returned.

BOT can be configured by enabling or disabling pruning, setting the threshold to a specific value, and choosing a specific word similarity. Whereas the possible values for the first two parameters result from their domain, we consider three word similarities. The *Levenshtein* similarity (LEV) [30] is a widely adapted syntactic similarity measure and *Lin*'s similarity (LIN) [31] is a semantic measure based on WordNet [32], a lexical database for English. For the CP dataset we rely on GermaNet [33] instead. Finally, we adapt the *cosine co-occurence* (CCO) similarity [34] based on model collection statistics. For two words, CCO is the cosine of the angle between their context vectors. To construct these vectors we count how often each word co-occurs with any other word in the labels from the model collection. For each word we select the two most frequently co-occurring words as its context words. For two words the context vectors contain the co-occurence counts for the words in the union of their context words and the according word.

The challenge now is to define a search strategy that automatically and reliably identifies configurations with a high effectiveness *without knowledge about true alignments*. To this end, we *could* simply compute δ_{sta} for all possible configurations and then select the result with the highest score. Besides being computationally expensive, this strategy is prone to select

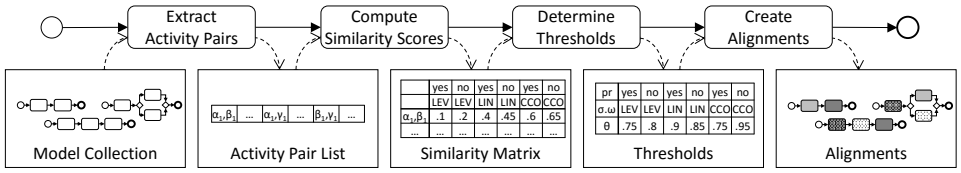

Figure 4: The OPBOT matching process

outliers. Consider, e.g., the scatter plot for δ_{sta} vs. F_μ on BR in Fig. 3 and the range of F_μ-values for $\delta_{sta} = .7$ and $\delta_{sta} = .75$. Here, both δ_{sta}-values yield overlapping f-measures intervals ($[0.44, 0.64]$ vs. $[0.48, 0.8]$). Thus, the configuration with the higher δ_{sta} score might actually have a lower effectiveness, e.g., (δ_{sta}, F_μ) of $(0.7, 0.64)$ vs. $(0.75, 0.48)$.

With respect to these observations, we have tested various search strategies and developed the *order preserving bag-of-words technique* (OPBOT). It consists of four sequential steps that are shown in Figure 4.

Extract activity pairs: First, we derive the set of all activity pairs from the pre-processed model collection where we expect labels to be normalized and the π_{sta} values as well as the word co-occurrence counts to be available.

Compute similarity scores: Next, a similarity matrix is determined. It contains six similarity scores for each activity pair, one score per combination of the three word similarities and the two pruning options.

Determine thresholds: For each of the six combinations we next search for the threshold $\theta \in [\theta_{min}, 1]$. We limit the possible values to at least θ_{min} to speed up the search and to lower the risk of selecting outliers. From BR and UA we determined that $\theta_{min} = 0.6$ for LEV and LIN as well as $\theta_{min} = 0.7$ for CCO could safely be set without risking to exclude the best configuration. Then, we compute δ_{sta} for each of the distinct similarity scores that the combination yielded within $[\theta_{min}, 1]$ and select the score with the highest value for δ_{sta} as the threshold. As the total number of threshold values

varies across the combinations and model collections, we in total obtained 874 configurations on BR, 2,683 on UA, 701 on AM, and 242 on CP.

Create alignments: Finally, we construct the alignments based on the two configurations that yielded the highest values for δ_{sta}. Again, we minimize the risk of favoring outliers by considering two configurations. In particular, we propose an activity pair as a correspondence, if for at least one of the two configurations the similarity score is equal to or higher than the threshold.

4.2. Evaluation

In the evaluation, we compare OPBOT with other matchers and a semi-manual configuration approach for BOT. Finally, we examine the general validity of δ_{sta} as well as its portability to matcher selection.

Effectiveness. To investigate OPBOT's effectiveness our primary interest is its relative performance, i.e., how close can it get to the maximum micro f-measure yielded by any possible BOT configuration. In this regard, Table 4 shows the effectiveness of OPBOT and contrasts it to BOT_{max}, the BOT configuration with the highest micro f-measure[6], determined through exhaustive search with knowledge of the gold standards. This comparison is done purely to assess the potential of OPBOT, since BOT_{max} is determined based on the unrealistic assumption that the true correspondences are known. On average OPBOT achieves 97.6% of the micro f-measure of BOT_{max}. On BR it is close to the maximum (.520 vs. .534 $\triangleq$ 97.4%) and on UA even better (.411 vs. .393 $\triangleq$ 104.6%) due to the combination of the top two matcher configurations. While OPBOT is also close to the maximum

[6]We chose the macro level measures for UA in Table 4, as there were no micro level measures reported in the matching contest for this dataset [9].

Table 4: Effectiveness of OPBOT, BOT, and the best purely label-based (LB) as well as control flow considering (CF) matchers from the matching contests [9, 10]

	BR			UA			AM			CP		
	P_μ	R_μ	F_μ	P_M	R_M	F_M	P_μ	R_μ	F_μ	P_μ	R_μ	F_μ
OPBOT	.61	.45	.52	.46	.41	.41	.60	.67	.63	.73	.34	.46
BOT_{max}	.65	.45	.53	.63	.33	.40	.83	.59	.69	.67	.37	.48
LB [9, 10]	.50	.42	.46	.37	.39	.38	.79	.60	.68	-	-	-
CF [9, 10]	.65	.31	.42	.36	.37	.36	.76	.56	.65	-	-	-

on CP (.463 vs. .479 $\triangleq$ 96.7%), its lowest relative performance (.630 vs. .686 $\triangleq$ 91.8%) is yielded for AM. This near-optimality gives evidence that the δ_{sta}-based search strategy is reliable in delivering effective configurations.

Finally, for broad comparison to the state-of-the-art, Table 4 shows the best matchers in terms of the f-measure from the matching contests [9, 10], excluding earlier versions of BOT and OPBOT. Here, we seperated purely label-based and control flow considering matchers. Overall, OPBOT yields slightly better f-measures on BR and UA, but performs marginally worse on AM. This comparable performance in combination with the fact that the six baseline values were yielded by five different matchers (the matcher from [8] was the best control flow considering technique on BR and AM) shows the advantage of the automated configuration. That is, without requiring efforts from users top-performing matcher configurations are identified.

Comparison to semi-manual configuration. Next, we compare OPBOT's performance to a semi-manual configuration approach: a part of the model collection is manually matched, then the best-performing configuration on the resulting alignments is automatically determined and used to match the remaining model pairs. To this end, for each dataset we randomly partitioned the 36 model pairs into $s = 36/k$ distinct sets of size

Table 5: Results of the semi-manual configuration approach

	BR			UA			AM			CP		
k	$\overline{F_\mu}$	E_c	E_p	$\overline{F_M}$	E_c	E_p	$\overline{F_\mu}$	E_c	E_p	$\overline{F_\mu}$	E_c	E_p
1	.44	16	371	.35	15	746	.44	6	126	.41	10	51
2	.47	32	742	.38	30	1492	.53	12	253	.42	21	104
3	.48	49	1113	.39	44	2238	.59	19	380	.44	31	156
6	.52	97	2226	.40	89	4476	.62	37	760	.45	63	311

$k \in \{1, 2, 3, 6\}$. For each k we determined 36 sets by generating $36/s$ partitions. Then, for each of the sets we determined the best BOT configuration and evaluated this configuration on the rest of the model pairs. Finally, per k we compute the average f-measure $\overline{F_\mu}$ (or $\overline{F_M}$) as an estimation of the effectiveness that can be achieved by training BOT. Moreover, the average number of correspondences E_c and activity pairs E_p in the training sets serve as estimates of the users' effort: the user needs to correctly identify E_c correspondences from a pool of E_p candidates.

Table 5 shows that on all datasets the average f-measure increases with a growing k and for $k = 6$ it is virtually equal to that of OPBOT. The comparison of the substantial amount of effort that users need to invest to get close to OPBOT, e.g., $E_c = 97$ and $E_p = 2226$ for $k = 6$ on BR, to OPBOT's execution time further demonstrates its utility. Here, we observed the following times on a laptop with an Intel i7 processor and 16GB of RAM running Java 1.8: $0.1s$ on CP, $0.2s$ on AM, $0.9s$ on BR, and $7.8s$ on UA. In light of the effort needed to semi-manually configure BOT, these times give further evidence towards the utility of OPBOT.

General Validity. To examine the general validity of the order relationship score we here repeat the analysis from Section 3.3 on AM and CP. For CP, the gold standard yields a slightly lower value than for the development

datasets ($\delta_{sta}^{GS} = .86$), but the correlation between δ_{sta} and F_μ is still very strong ($\rho = .965$ with $p \ll 0.001$). In contrast, on AM's gold standard the order relationship score is much lower ($\delta_{sta}^{GS} = .77$) and the correlation is only moderate ($\rho = .542$ with $p \ll 0.001$). Unlike the other datasets where all process models refer to the same higher-level process, AM contains model pairs where correspondences exist but appear in different contexts, and other pairs *without any* correspondences. The latter strongly impact δ_{sta}, which is 0 on model pairs without correspondences. To investigate the magnitude of this effect, we removed all six such pairs from the dataset and calculated the order relationship score for the gold standard and the correlation score again: both scores are improved strongly ($\delta_{sta}^{GS} = .93$ and $\rho = .807$ with $p \ll 0.001$), indicating a strong correlation between δ_{sta} and F_μ. These results further substantiate the general validity of δ_{sta}, but show that its applicability is limited to model pairs with at least some similarity.

Portability to Matcher Selection. We further analyzed the applicability of the order relationship score to matcher selection. To this end, we used δ_{sta} to rank the twelve matchers competing in the second contest [10], where all of the results on BR and AM are publicly available[7]. We then compared the top m matchers in this ranking to the m best performing matchers in terms of F_μ from the contest. Note that to avoid distortion, we excluded the model pairs without correspondences on AM. While on BR the best performing matcher ($m = 1$) is also the best with regard to the score, this is not the case on AM. However, for $m = 3$ the score finds two of the best performing matchers and three for $m = 5$ on both datasets. Although the best performing matcher on AM does not yield the highest score, we still

[7] https://ai.wu.ac.at/emisa2015/contest.php, accessed: 20/12/2016

identify matchers with a high effectiveness. That is, the top ranked matcher in terms of δ_{sta} achieves 89% of the micro f-measure of the best performing matcher. For $m = 3$ we yielded a maximum of 98% and for $m = 5$ of 100%. The analysis gives evidence towards the score's applicability to matcher selection. As the matchers were designed by other researchers, the results further substantiate the score's general validity.

5. Discussion

In this article, we used two development datasets to analyze how control flow information can contribute to the matching process. Firstly, we investigated options for a pairwise comparison of activities. While the goal was to extend the label-based detection of correspondences, our analysis revealed that the considered control flow similarities have a low discriminative power. Secondly, we assessed if control flow information can be used to detect complex correspondences. Here, our analysis suggested that deriving candidates for complex correspondences from the control flow yields unreliable results. Thirdly and most importantly, we studied alignment consistency based on the order relationship score δ and found that the consistency of alignments is correlated to their effectiveness. To further investigate this result, we developed OPBOT which uses δ_{sta} as an effectiveness oracle to detect promising matcher configurations. Relying on another two datasets, we demonstrated OPBOT's utility by comparing it to other matchers and a semi-manual configuration approach. We also confirmed the correlation between δ_{sta} and F_μ on the additional datasets and successfully applied δ_{sta} to matcher selection, thus substantiating the general validity of our findings.

Yet, there are threats that *limit the validity* of these findings. First,

concerns regarding the *construct validity* exist, i.e., how well our constructs reflect cause and effect. This most notably pertains the use of self-defined activity properties which led to the rejection of two use cases. While we oriented these properties towards existing approaches, they only reflect one particular view and we can thus not rule out that more suitable properties would actually allow for a successful implementation of the rejected use cases. In this regard, we support extensibility and replicability of our analyses by publishing the source code (see Section 2.2 for more details). Second, the *external validity* is restricted, i.e., the degree to which our results are generalizable. Considering that process model collections in industry can contain thousands of models, the use of 144 model pairs cannot be regarded as an exhaustive evaluation. The size also affects the *ecological validity*, i.e., the degree to which our data reflects real-life situations. In this regard, the development datasets partly contain models created under laboratory conditions by students in the context of business process management lectures [6, 9]. Moreover, three out of four datasets consist of process models that all refer to the same abstract process. Consequently, a broader evaluation including a larger variety of matching scenarios, e.g., comparisons within as well as across organizations, business units, and functional areas, would indeed be warranted, but are hampered by the unavailability of datasets. However, this is an issue for all works in this space, and we hope the additional dataset accompanying this work helps to improve the situation. Finally, the *internal validity* – the extent to which causal relationships hold – might be compromised. In a separate direction of our work [35] we found that opinions of experts regarding the ground truth are more diverse than the use of binary gold standards suggests. Thus, we might draw a somewhat distorted picture. To mitigate this threat we used four different

datasets with different gold standards created by different experts.

6. Related Work

Throughout the paper we referred to closely related works, e.g., [2, 3, 4, 5, 6, 7, 8, 9, 10, 20, 22]. Here we discuss the broader relationship to other areas. In recent years, *process similarity search* addressed the comparison of process models, not activities. Whereas some measures solely exploit textual information, e.g., [36], others refer to the control flow. For instance, in [23] the graph edit distance for alignment construction [5] 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 [37]. In contrast, the workflow similarity in [38] is based on the number of corresponding nodes and edges, like the edit distance [5]. The measure in [39] considers the depth of activities in process trees. An overview of similarity measures is provided in [40]. Similar to many of these approaches, the order relationship score compares the relative position of activities. Yet, in contrast to the similarity measures it disregards non-corresponding nodes, as it is not measuring the similarity of the models, but the consistency of alignments.

Another area of interest is schema and ontology matching where matcher configuration has been recognized as a central challenge [18], leading to the development of various configuration approaches [41]. Many methods rely on human intervention, e.g., a software tool that assists users in manually assembling and refining schema matchers is described in [42]. Few approaches address autonomous configuration. The matcher in [43] optimizes its configuration for two given schemas based on automatically derived versions of

these schemas and the correspondences between them and the original. The approach introduced in [44] views ontology matchers as individual agents that negotiate an alignment. Complementary to these works, OPBOT addresses the configuration of process model matchers and uses process specific control flow information to estimate their effectiveness.

7. Conclusion

This article complements prior research on process model matching which primarily focused on evaluating the effectiveness of matchers, but did not study the benefits and limitations of relying on control flow information. In this regard, our empirical analyses suggest that such information forms a slim basis for activity comparison and for the detection of complex correspondences. Yet, the analyses also reveal that it can in fact be used to check the consistency of alignments. In addition, our self-configuring matcher, OPBOT, demonstrates that high performing matcher configurations can be identified by assessing the consistency of the proposed alignments.

Regarding future work, we believe that more attention should be spent on matcher adaptation and configuration, particularly considering (i) the limitations, (ii) the fairly low effectiveness of process model matching techniques in general [9, 10], and (iii) the advances we achieved through matcher configuration in this work. More specifically, we aim to further advance our approaches and prepare them for practical application. First, we want to improve the configuration search for large collections where it can become computationally expensive, e.g., by separately optimizing the configurations for clusters of model pairs. Furthermore, we aim to integrate the automated configuration with our feedback based optimization [20] to maximize the

effectiveness and minimize user efforts. To orient our extensions towards realistic use cases, we also strive to achieve a broader coverage of matching scenarios in empirical data, and to consider non-binary gold standards that better reflect the diversity of experts' perceptions.

References

[1] C. C. Ekanayake, M. La Rosa, A. H. ter Hofstede, M.-C. Fauvet, Fragment-based version management for repositories of business process models, in: On the Move to Meaningful Internet Systems (OTM), Confederated Intl. Conferences: CoopIS, DOA-SVI, and ODBASE, Hersonissos, Greece, October 17-21, 2011, Proceedings Part I, Springer, 2011, pp. 20–37.

[2] C. Klinkmüller, I. Weber, J. Mendling, H. Leopold, A. Ludwig, Increasing recall of process model matching by improved activity label matching, in: Business Process Management (BPM): 11th International Conference, Beijing, China, August 26-30, 2013. Proceedings, Springer, 2013, pp. 211–218.

[3] J. Fengel, Semantic technologies for aligning heterogeneous business process models, Business Process Management Journal 20 (4) (2014) 549–570.

[4] M. C. Branco, J. Troya, K. Czarnecki, J. Küster, H. Völzer, Matching business process workflows across abstraction levels, in: Proceedings of the 15th International Conference on Model Driven Engineering Languages and Systems, Springer, 2012, pp. 626–641.

[5] R. Dijkman, M. Dumas, L. Garcia-Banuelos, R. Kaarik, Aligning business process models, in: 2009 IEEE International Enterprise Distributed Object Computing Conference, IEEE, 2009, pp. 45–53.

[6] H. Leopold, M. Niepert, M. Weidlich, J. Mendling, R. M. Dijkman, H. Stuckenschmidt, Probabilistic optimization of semantic process model matching, in: International Conference on Business Process Management, Springer, 2012, pp. 319–334.

[7] M. Weidlich, R. Dijkman, J. Mendling, The icop framework: Identification of correspondences between process models, in: Advanced Information Systems Engineering: 22nd International Conference, CAiSE 2010, Hammamet, Tunisia, June 7-9, 2010. Proceedings, Springer, 2010, pp. 483–498.

[8] M. Weidlich, E. Sheetrit, M. C. Branco, A. Gal, Matching business process models using positional passage-based language models, in: Conceptual Modeling: 32th International Conference, ER 2013, Hong-Kong, China, November 11-13, 2013. Proceedings, Springer, 2013, pp. 130–137.

[9] 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, M. Weidlich, Report: The process model matching contest 2013, in: Business Process Management Workshops, Beijing, China, August 26, 2013, Revised Paper, Springer, 2013, pp. 442–463.

[10] G. Antunes, M. Bakhshandeh, J. Borbinha, J. Cardoso, S. Dadashnia, C. Di Francescomarino, M. Dragoni, P. Fettke, A. Gal, C. Ghi-

dini, 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, M. Weidlich, The process model matching contest 2015, in: Proceedings of the 6th Int. Workshop on Enterprise Modelling and Information Systems Architectures, GI, 2015, pp. 127–155.

[11] C. D. Manning, H. Schütze, Foundations of Statistical Natural Language Processing, MIT Press, Cambridge, MA, USA, 1999.

[12] C. D. Manning, P. Raghavan, H. Schütze, Introduction to Information Retrieval, Cambridge University Press, New York, 2008.

[13] J. Euzenat, P. Shvaiko, Ontology Matching, Springer, Berlin, 2013.

[14] E. Rahm, P. A. Bernstein, A survey of approaches to automatic schema matching, VLDB J. 10 (4) (2001) 334–350.

[15] J. Ling, L. Zhang, Q. Feng, Business process model alignment: An approach to support fast discovering complex matches, in: K. Mertins, F. Bénaben, R. Poler, J.-P. Bourrières (Eds.), Enterprise Interoperability VI, Springer International, 2014, pp. 41–51.

[16] S. L. Salzberg, On comparing classifiers: Pitfalls to avoid and a recommended approach, Data Mining and Knowledge Discovery 1 (3) (1997) 317–328.

[17] J. Demšar, Statistical comparisons of classifiers over multiple data sets, J. Mach. Learn. Res. 7 (2006) 1–30.

[18] P. Shvaiko, J. Euzenat, Ten challenges for ontology matching, in: R. Meersman, Z. Tari (Eds.), On the Move to Meaningful Internet Sys-

tems: OTM 2008: OTM 2008 Confederated International Conferences, CoopIS, DOA, GADA, IS, and ODBASE 2008, Monterrey, Mexico, November 9-14, 2008, Proceedings, Part II, Springer, 2008, pp. 1164–1182.

[19] J. Zobel, Writing for Computer Science, Springer, Heidelberg, 2004.

[20] C. Klinkmüller, H. Leopold, I. Weber, J. Mendling, A. Ludwig, Listen to me: Improving process model matching through user feedback, in: Business Process Management (BPM): 12th International Conference, Haifa, Israel, September 7-11, 2014. Proceedings, Springer, 2014, pp. 84–100.

[21] M. Weidlich, T. Sagi, H. Leopold, A. Gal, J. Mendling, Predicting the quality of process model matching, in: Business Process Management (BPM): 11th International Conference, Beijing, China, August 26-30, 2013. Proceedings, Springer, 2013, pp. 203–210.

[22] S. Nejati, M. Sabetzadeh, M. Chechik, S. Easterbrook, P. Zave, Matching and merging of statecharts specifications, in: 29th International Conference on Software Engineering, IEEE, 2007, pp. 54–64.

[23] R. Dijkman, M. Dumas, B. van Dongen, R. Käärik, J. Mendling, Similarity of business process models: Metrics and evaluation, Inf. Syst. 36 (2) (2011) 498–516.

[24] J. Vanhatalo, H. Völzer, J. Koehler, The refined process structure tree, IEEE Trans. Knowl. Data Eng. 68 (9) (2009) 793–818.

[25] A. Polyvyanyy, L. García-Bañuelos, M. Dumas, Structuring acyclic process models, Information Systems 37 (6) (2012) 518–538.

[26] M. Weidlich, A. Polyvyanyy, J. Mendling, M. Weske, Causal behavioural profiles - efficient computation, applications, and evaluation, Fundamenta Informaticae 113 (3–4) (2010) 399–435.

[27] F. J. Massey Jr., The Kolmogorov-Smirnov test for goodness of fit, Journal of the American Statistical Association 46 (253) (1951) 68–78.

[28] P.-N. Tan, M. Steinbach, V. Kumar, Introduction to Data Mining, Pearson Education Limited, Harlow, 2014.

[29] M. F. Porter, An algorithm for suffix stripping, Program 14 (3) (1980) 130–137.

[30] V. I. Levenshtein, Binary codes capable of correcting deletions, insertions and reversals, Dokl. Phys. 10 (8) (1966) 707–710.

[31] D. Lin, An information-theoretic definition of similarity, in: Proceedings of the Fifteenth International Conference on Machine Learning, Morgan Kaufmann, 1998, pp. 296–304.

[32] G. A. Miller, Wordnet: A lexical database for english, Commun. ACM 38 (11) (1995) 39–41.

[33] B. Hamp, H. Feldweg, GermaNet – a lexical-semantic net for German, in: Proceedings of the ACL workshop Automatic Information Extraction and Building of Lexical Semantic Resources for NLP Applications, ACL, 1997, pp. 9–15.

[34] R. Navigli, Word sense disambiguation: A survey, ACM Comput. Surv. 41 (2) (2009) 10:1–10:69.

[35] C. Rodríguez, C. Klinkmüller, I. Weber, F. Daniel, F. Casati, Activity matching with human intelligence, in: Business Process Management (BPM) Forum, Rio de Janeiro, Brazil, September 18-22, 2016, Proceedings, Springer, 2016, pp. 124–140.

[36] A. Schoknecht, N. Fischer, A. Oberweis, Process model search using latent semantic analysis, in: 1st International Workshop on Process Querying, 2016.

[37] P. Schumacher, M. Minor, Towards a trace index based workflow similarity function, in: C. Lutz, M. Thielscher (Eds.), KI 2014: Advances in Artificial Intelligence: 37th Annual German Conference on AI, Stuttgart, Germany, September 22-26, 2014. Proceedings, Springer, 2014, pp. 225–230.

[38] R. Bergmann, Y. Gil, Similarity assessment and efficient retrieval of semantic workflows, Information Systems 40 (2014) 115–127.

[39] D. Sánchez-Charles, V. Muntés-Mulero, J. Carmona, M. Solé, Process model comparison based on cophenetic distance, in: Business Process Management (BPM) Forum, Rio de Janeiro, Brazil, September 18-22, 2016, Proceedings, Springer, 2016, pp. 141–158.

[40] M. Becker, R. Laue, A comparative survey of business process similarity measures, Comput. Ind. 63 (2) (2012) 148–167.

[41] P. Shvaiko, J. Euzenat, Ontology matching: State of the art and future challenges, IEEE Trans. Knowl. Data Eng. 25 (1) (2013) 158–176.

[42] E. Peukert, J. Eberius, E. Rahm, Amc - a framework for modelling and comparing matching systems as matching processes, in: 2011 IEEE

27th International Conference on Data Engineering, IEEE, 2011, pp. 1304–1307.

[43] Y. Lee, M. Sayyadian, A. Doan, A. S. Rosenthal, etuner: Tuning schema matching software using synthetic scenarios, VLDB J. 16 (1) (2007) 97–122.

[44] V. Spiliopoulos, G. Vouros, Synthesizing ontology alignment methods using the max-sum algorithm, IEEE Trans. Knowl. Data Eng. 24 (5) (2012) 940–951.