Overcoming Individual Process Model Matcher Weaknesses Using Ensemble Matching

Christian Meilicke[a], Henrik Leopold[b], Elena Kuss[a], Heiner Stuckenschmidt[a], Hajo A. Reijers[b]

[a] *Research Group Data and Web Science, University of Mannheim, Mannheim, Germany*
[b] *Department of Computer Science, Vrije Universiteit Amsterdam, Amsterdam, The Netherlands*

Abstract

In recent years, a considerable number of process model matching techniques have been proposed. The goal of these techniques is to identify correspondences between the activities of two process models. However, the results from the Process Model Matching Contest 2015 reveal that there is still no universally applicable matching technique and that each technique has particular strengths and weaknesses. It is hard or even impossible to choose the best technique for a given matching problem. We propose to cope with this problem by running an ensemble of matching techniques and automatically selecting a subset of the generated correspondences. To this end, we propose a Markov Logic based optimization approach that automatically selects the best correspondences. The approach builds on an adaption of a voting technique from the domain of schema matching and combines it with process model specific constraints. Our experiments show that our approach is capable of generating results that are significantly better than alternative approaches.

Keywords:

1. Introduction

The automatic identification of correspondences between the activities of process models using so-called *matchers* has received considerable attention in recent years. The outcome is a plethora of matching techniques that exploit different features to identify activity correspondences as accurately as possible (e.g., [1, 2, 3, 4, 5]). One reason for this large interest in matching techniques is the wide range of process model analysis techniques that rely on the correspondences generated by these matchers. Examples include process model search [6, 7, 8], clone detection [9, 10], and the harmonization of process model variants [11, 12].

Email addresses: christian@informatik.uni-mannheim.de (Christian Meilicke), h.leopold@vu.nl (Henrik Leopold), elena@informatik.uni-mannheim.de (Elena Kuss), heiner@informatik.uni-mannheim.de (Heiner Stuckenschmidt), h.a.reijers@vu.nl (Hajo A. Reijers)

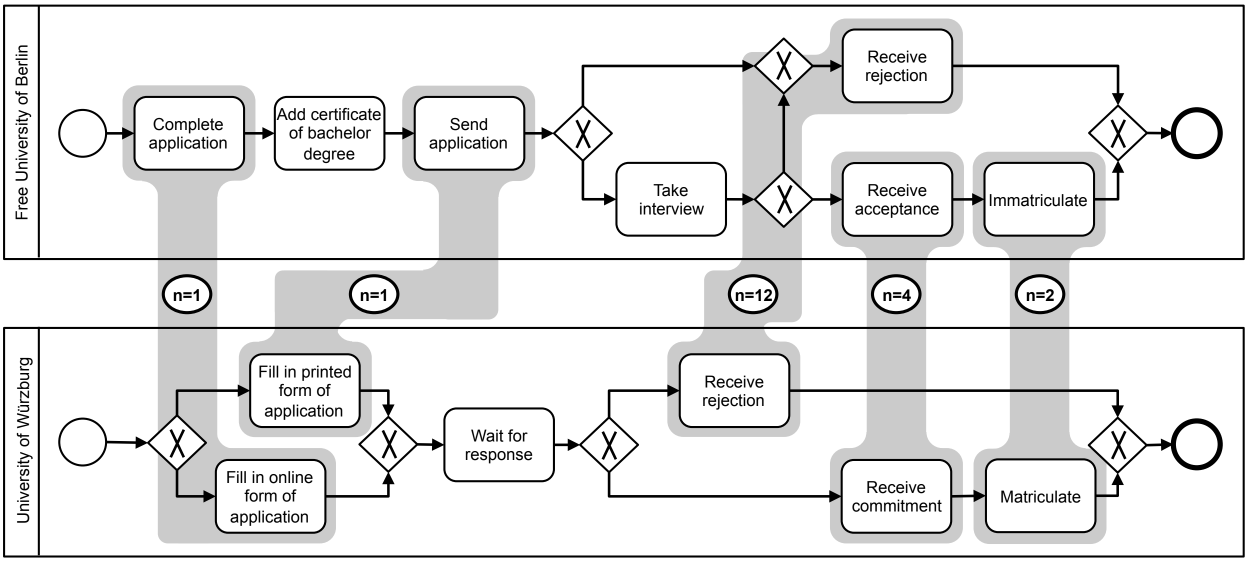
Despite the growing number of matching techniques, comparative evaluation experiments reveal that there is still no universally applicable matching technique available. The Process Model Matching Contest (PMMC) 2015 shows that a technique that performs well on dataset A can perform poorly on dataset B [13]. Apparently, each technique has particular strengths and weaknesses, which results in a considerably varying level of performance. From a practical perspective, this implies that users have to select the appropriate matching technique for a given problem. Taking the large number of available techniques and their complexity into account, this is often not feasible.

To cope with the problem that a non-optimal matching technique is selected, we introduce an approach that builds on an ensemble of matching techniques. More specifically, we propose a Markov Logic based optimization approach that automatically selects the best correspondences from the output of several matching techniques without any prior knowledge about these techniques. Our approach builds on an adapted form of a voting technique from the domain of schema matching which is combined with process model specific constraints. The specific contributions of this paper are:

- We apply the idea of majority voting, originally developed for ontology matching [14], to the field of business process model matching.
- We use several constraints to guide the selection of correspondences. In comparison to prior work [2], we introduce two novel constraints, i.e., a constraint that takes the resource perspective into account and a one-to-n cardinality constraint.
- We introduce a Markov Logic formalization that computes the alignment with the highest number of votes from the members of the matcher ensemble that satisfies the introduced constraints.
- We conduct comprehensive and systematic experiments to demonstrate the impact of the majority voting and the constraints.

The rest of the paper is organized as follows. Section 2 illustrates and motivates the problem we want to solve. Section 3 introduces the notion of majority voting in the context of a constrained based approach. Section 4 defines the constraints we employ. Section 5 describes the Markov Logic implementation of our approach. Section 6 presents the results of our experiments, which are evaluated and discussed in Section 7. Section 8 finally discusses related work before Section 9 concludes the paper.

Figure 1: Correspondences identified by the matchers from the PMMC 2015



2. Problem Statement

Given two process models $\mathcal{M}_1$ and $\mathcal{M}_2$ with their sets of activities $A(\mathcal{M}_1)$ and $A(\mathcal{M}_2)$, an activity correspondence between $\mathcal{M}_1$ and $\mathcal{M}_2$ is a pair $\langle a_1, a_2 \rangle$ with $a_1 \in A(\mathcal{M}_1)$ and $a_2 \in A(\mathcal{M}_2)$. An alignment $\mathcal{C}$ between $\mathcal{M}_1$ and $\mathcal{M}_2$ is a set of correspondences between $\mathcal{M}_1$ and $\mathcal{M}_2$. Alignments between process models can be created by human experts or by a matching algorithm. We will refer in the following to such an algorithm (also known as matching technique, matching tool, or matching system) as a matcher m, and we refer to a set of matchers that is applied to solve the same matching task as an ensemble of matchers $\mathcal{E}$. A matcher ensemble results in a comprehensive alignment $\mathcal{H}$ that is the union of all alignments $\mathcal{C}_m$, whereas each $\mathcal{C}_m$ is generated by some $m \in \mathcal{E}$. In the following, we assume that each correspondence $\langle a_1, a_2 \rangle \in \mathcal{H}$ is annotated with the set of matchers (more precisely, their IDs) that have generated this correspondence. We call such an annotated set $\mathcal{H}$ a set of matching hypotheses.

The task that we have to solve is to select a subset $\mathcal{H}^* \subseteq \mathcal{H}$ from the set of all matching hypotheses. Figure 1 illustrates this problem by showing a (simplified) model pair from the PMMC 2015 University Admission dataset [13]. It consists of two different process models describing the steps students have to take when applying for a graduate study program. The grey shades denote the correspondences that have been identified. The numbers in the ellipses show the number of matchers that have recognized a particular correspondence.

The numbers from Figure 1 highlight important points. First, we learn that trivial correspondences, such as the one between the identically labeled activities *Receive rejection*, can be expected to be found by a considerable number of matchers. Second, we see that the fact that a low number of matchers identified a certain correspondence does not necessarily mean that this correspondence is false. Analyzing the correspondences with a low value of *n* in more detail, we observe that the correspondences between *Complete application* and *Fill in online form of application* (n=1) as well as *Send application* and *Fill in printed form of application* (n=1) are indeed not meaningful. However, the correspondences between *Receive acceptance* and *Receive commitment* (n=4) as well as *Immatriculate* and *Matriculate* (n=2) are correct. The reason for the low values of n was the inability of many matchers to recognize the synonymous terms of these activities. This illustrates the complexity of selecting a suitable subset $\mathcal{H}^* \subseteq \mathcal{H}$.

We solve this task as an optimization problem taking into account the number of matchers that generated a correspondence, cardinality constraints, and process model specific constraints. We do neither leverage any external resources nor do we require any further knowledge about the matchers from the ensemble.

3. Voting

The main idea behind the voting principle is to count the generation of a correspondence $\langle a_1, a_2 \rangle$ by a matcher m as a vote for $\langle a_1, a_2 \rangle$. In a majority voting setting, we require that at least half of the matchers from an ensemble $\mathcal{E}$ generate a correspondence $\langle a_1, a_2 \rangle$. If this is the case, we accept $\langle a_1, a_2 \rangle$ as part of the final solution $\mathcal{H}^*$. In the field of ontology matching, this simple approach performs surprisingly well [14]. It almost met the performance of a supervised machine learning approach that tried to understand which matcher performs best for a particular matching problem given additional information about the testcases and the matcher performance on a subset. We reuse the approach proposed in [14] and integrate it with a constraint-based approach.

As a generalization of the approach, we introduce the #n-alignment, which denotes the subset of $\mathcal{H}$ that has been generated by at least n matchers. The #1-alignment will have high recall and low precision, while we expect decreasing recall and increasing precision with increasing n. The #1-alignment will probably be too noisy as input for our approach, because it contains each correspondence created by one (or several) matcher(s). Thus, it makes sense to filter out correspondences with few votes prior to solving the optimization problem. Contrary to this, the #n-alignments for $n > |\mathcal{E}|/2$ will probably be highly precise such that a constraint based approach will have a limited impact. The optimal input to our constraint-based approach can be expected for low n values between 2 and $|\mathcal{E}|/2$.

In Section 4, we introduce cardinality, resource and sequence constraints in details. Now we discuss the one-to-one constraint as an example to explain the connection between constraints and voting. An alignment is consistent against the one-to-one constraint if there exists no pair of correspondences $\langle a, a' \rangle$ and $\langle a, a^* \rangle$ in the alignment such that $a' \neq a^*$. It is also possible to define constraints such that more than two correspondences are required to cause a violation. We refer to such a constraint violating set of correspondences as a conflict set. A set of constraints marks a subset of the alignments power set as conflict sets. If we remove at least one correspondence from each conflict set, we construct an alignment that is consistent against the constraints.

Going back to our simple example of the one-to-one constraint, one can construct a consistent alignment by applying the following greedy algorithm as shown in [15]. We briefly explain this algorithm to highlight its difference against an algorithm that constructs a solution that is optimal with respect to the sum of votes. The algorithm starts by selecting correspondence m with the highest number of votes and removes any other correspondence from $\mathcal{H}$ that refers to one of the entities mapped in m. The algorithm continues in decreasing order of votes until all one-to-one conflicts are resolved. We propose an alternative to the state-of-the-art greedy algorithm by selecting a one-to-one subset $\mathcal{H}^*$ of $\mathcal{H}$ that is optimal with respect to the sum of votes.

For the alignment $\mathcal{H} = \{c_1, c_2, c_3\}$ the optimal solution is $\mathcal{H}^* = \{c_2, c_3\}$ with a total of 7+4=11 votes. The greedy approach results in $\mathcal{H}^* = \{c_1\}$.

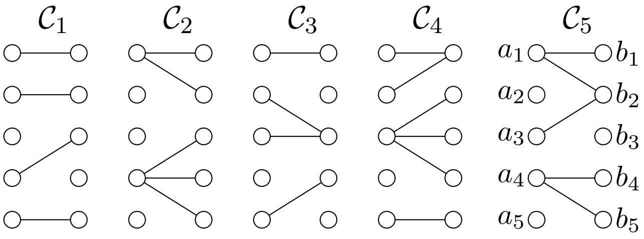
$$\begin{aligned} c_1 &= \langle \textit{Check online application, Send online application} \rangle, 8 \\ c_2 &= \langle \textit{Check online application, Check application} \rangle, 7 \\ c_3 &= \langle \textit{Apply online , Send online application} \rangle, 4 \end{aligned}$$

In our approach, we extract consistent alignments that are optimal with respect to the sum of votes. However, the constraints that guide the selection are more complex than the simple one-to-one constraint.

4. Constraints

In Section 4.1, we distinguish between the well-known one-to-one and the novel contribution of a one-to-n constraint. In Section 4.2, we introduce a novel resource-based constraint. In Section 4.3, we recall three sequence-based constraints originally introduced in [2].

Figure 2: Alignments with different cardinalities.

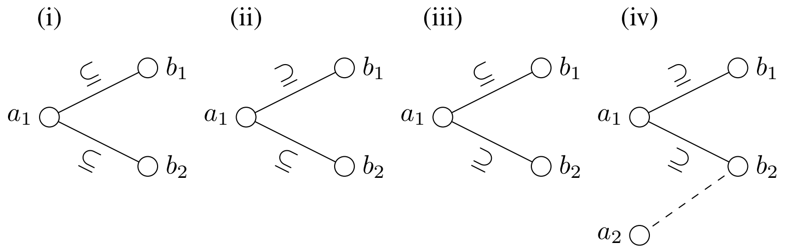


4.1. Cardinality Constraints

Figure 2 depicts five alignments $\mathcal{C}_1$ to $\mathcal{C}_5$. Circles within columns represent activities within the same model. Lines connecting the circles represent correspondences. $\mathcal{C}_1$ is a one-to-one alignment while the others are not one-to-one alignments. Suppose now that we want to select a subset $\mathcal{H}^*$ from $\mathcal{H}$ such that all correspondences in $\mathcal{H}^*$ are *equivalence* correspondences. This additional restriction on the type of semantic relation allows us to conclude that any subset from $\mathcal{H}$ that is not a one-to-one alignment contains at least one incorrect correspondence. Suppose we have $\langle a, a' \rangle$ and $\langle a, a^* \rangle$ violating the one-to-one constraint. Due to the transitivity of equivalence, we can conclude $a' = a^*$. However, it makes no sense that an activity is defined twice within a model. Obviously, a process model has to encode the re-occurrence of an activity via different paths that allow to reach this activity. Thus, we are entitled to use the one-to-one constraint, whenever we want to select a consistent set of equivalence correspondences.

Now we define the notion of a one-to-n alignment, which allows us to differentiate between alignments like $\mathcal{C}_2$, $\mathcal{C}_3$, or $\mathcal{C}_4$ on the one hand and alignments like $\mathcal{C}_5$ on the other hand. This requires to introduce the formal notion of a difunctional relation [16]: Let $\mathcal{R} \circ \mathcal{R}'$ denote the relation composition of two relations $\mathcal{R}$ and $\mathcal{R}'$ (in our case alignments) and let $\mathcal{R}^{-1}$ denote the inverse relation of $\mathcal{R}$. A relation $\mathcal{R}$ is difunctional if $\mathcal{R} \circ \mathcal{R}^{-1} \circ \mathcal{R} = \mathcal{R}$. Alternatively, a difunctional alignment relation can be characterized as follows. Let $x\mathcal{R} = \{y \mid x\mathcal{R}y\}$ be the set of entities to which an entity x is mapped by $\mathcal{R}$ and let $\mathcal{R}$ be a difunctional relation. It follows that $x_1\mathcal{R} \cap x_2\mathcal{R} \neq \emptyset$ implies $x_1\mathcal{R} = x_2\mathcal{R}$. Checking against this definition, we can see that $\mathcal{C}_5$ is not difunctional, because $a_1\mathcal{C}_5 = \{b_1, b_2\}$ and $a_3\mathcal{C}_5 = \{b_2\}$, i.e., $a_1\mathcal{C}_5 \cap a_3\mathcal{C}_5 = \{b_2\} \neq \emptyset$, however, we have $a_1\mathcal{C}_5 \neq a_3\mathcal{C}_5$. We define an alignment that is difunctional as a one-to-n alignment. We say that an alignment is a proper one-to-n alignment if it is a one-to-n alignment that is not a one-to-one alignment. Moreover, we define an n-to-m alignment as an alignment that does not conform to the one-to-n

Figure 3: Four combinations of subsumption correspondences. Only one is valid (iv), however, it cannot be extended to a valid alignment by adding a correspondence depicted by a dashed line.



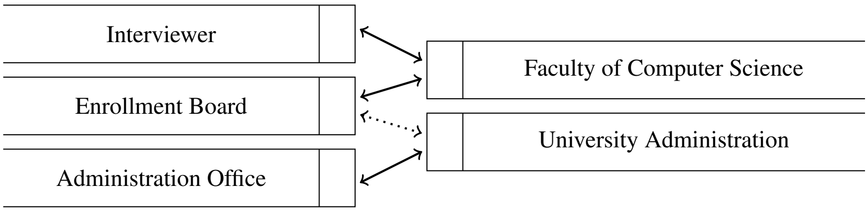
constraint.

Suppose now that we are interested in selecting correspondences that describe equivalence, *more-specific* or *more-general* relations. We call the latter two relations a subsumption relation and denote it with a symbol from set theory, i.e. $a \subseteq b$ (a is more specific than b) and $a \supseteq b$ (a is more general than b). Given that the selection $\mathcal{H}^*$ should contain equivalence or subsumption correspondences only, we can conclude that any subset $\mathcal{H}^*$ that is not a one-to-n alignment must contain incorrect correspondences. The prove for this claim is built on the distinction of four cases as depicted in Figure 3. Moreover, it requires the intuitive assumption that two activities within a workflow should not overlap. A related point to consider is that we exclude process models with subprocesses that correspond to an expanded activity or other constructs that introduce different levels of granularity within the same process model. Otherwise our assumption does not hold.

Now we try to construct the simplest case of an n-to-m alignment, using subsumption (or equivalence) correspondences, and show that such an alignment contradicts our assumption. First of all, we analyze all possible combinations of two correspondences resulting in a proper one-to-n alignment. In Case (i) a_1 is more specific than both b_1 and b_2. This implies that b_1 and b_2 are overlapping activities (or that a_1 is an "empty" activity), which contradicts our assumption. In Case (ii) we have $b_1 \subseteq a_1 \subseteq b_2$ and thus $b_1 \subseteq b_2$ which contradicts again our assumption. The same holds for Case (iii) by swapping b_1 and b_2. The only consistent combination is depicted in Case (iv), where a_1 is more general than both b_1 and b_2. Such an example is for instance $a_1 =$ *Send application*, $b_1 =$ *Send online application*, and $b_2 =$ *Send paper-pencil application*.

In order to extend this alignment to become non-difunctional, we have to add a correspondence that

Figure 4: A resource alignment between the resources of $\mathcal{M}_1$ (left side) and $\mathcal{M}_2$ (right side). Removing the correspondence represented by a dotted arrow resolves resource alignment inconsistency.



maps an activity from the first model, say a_2, to one of b_1 or b_2. This is indicated by the dashed line. If this correspondence expresses a more-general relation we are creating a combination of correspondences as in Case (i), where right and left side are swapped. If this correspondence expresses a more-specific relation we end up in Case (ii) (again, right and left side swapped). Thus, it is not possible to construct one-to-n alignment out of subsumption correspondences that does not contradict the assumption that two activities within a process model do not overlap. For that reason we can use the one-to-n constraint instead of using the strict one-to-one constraint if we want to select not only equivalence but also subsumption correspondences.

4.2. Resource Consistency

The activities of a process model are executed by one or more resources. Common notations such as Event-driven Process Chains (EPCs) and BPMN allow for the explicit definition of these resources. In the following, we introduce a novel constraint that guides the selection of correspondences using the resource perspective. We expect that our constraint has, in particular, a positive impact if two process models contain similar activities executed by different resources. Typical examples are activities like the receipt of a business object (e.g., an *Application*) that is passed trough different organizational units (e.g. *Administrative Office* → *Selection Committee*), where it is first received, in different ways processed, and forwarded to the next resource.

Let in the following $\mathcal{R}$ be a resource alignment that contains correspondences between resources of two process models $\mathcal{M}_1$ and $\mathcal{M}_2$. Obviously, $\mathcal{R}$ will not always be a one-to-one alignment because participants and their resources can be modeled at different levels of granularity. Thus, a single resource from one model might be mapped to several resources from the other.

Figure 4 depicts such a case using an example from a BPMN diagram, where resources are represented as swimlanes. The resource *Faculty of Computer Science* is mapped to both the resources *Enrollment Board* and *Interviewer*. This is correct for the specific matching problem, because the members of the *Faculty of Computer Science* do not only act as members of the *Enrollment Board*, but also interview applicants. Also, the resource *University Administration* is mapped to two other resources; to the *Enrollment Board* and to the *Administration Office*. It is obvious that this resource mapping constellation does not describe a clear hierarchy, since it implies partially overlapping responsibilities of the different resources. For example, the resource *Enrollment Board* is responsible for activities from the *Faculty of Computer Science* and the *University Administration*. At the same time, however, the *University Administration* is also responsible for activities from the *Administration Office* and the *Enrollment Board*. Thus, the example depicts an n-to-m resource alignment, which introduces mutually overlapping resources. This is inconsistent with the assumption that resources can be ordered in a hierarchy, which indicates an error in the resource alignment. For that reason, we define an n-to-m resource alignment as inconsistent.

The notion of resource alignment inconsistency itself does not help in filtering out problematic activity correspondences. For that reason, we define the notion of an *induced resource alignment*. We say that a correspondence between two activities induces a correspondence between the resources these activities are associated with. The resource alignment induced by an activity alignment is then defined as the set of resource correspondences induced by the activity correspondences of the activity alignment. We define an activity alignment as resource consistent if its induced resource alignment is consistent.

The idea of applying this constraint differs fundamentally from approaches (see for example [17]) where both models are annotated with the same resources (intra-organizational matching) or where an alignment between resources is created first. In our approach the activity alignment induces a resource alignment that is checked against the formal criteria of being a one-to-n alignment. If this criteria is not fulfilled, the reason for the problem is triggered back to the combination of activity correspondences that caused the problem. Our approach does not require any additional mechanism to establish a resource alignment.

4.3. Sequence Consistency

The sequence of activities defined in $\mathcal{M}_1$ and $\mathcal{M}_2$ can be used to enforce a set of a sequential consistency constraints on $\mathcal{H}$. These constraints have already been proposed in [2]. For our application scenario we select three of the proposed constraints that have been shown to achieve good results within a limited evaluation scenario. These constraints are based on the notion of strict order between activities and the exclusiveness of activities. Two activities a_1 and a_2 are in *strict order*, we say a_1 strictly precedes a_2, if a_1 is always executed before a_2. Two activities are *exclusive* iff there exists no execution order where both a_1

and a_2 are executed. An example for strict order are the activities *Send Application* and *Receive Rejection* from the *Free University Berlin* process in Figure 1; two exclusive activities are the activities *Immatriculate* and *Receive Rejection*.

Given two correspondences $\langle a, a' \rangle$ and $\langle b, b' \rangle$ we will apply the following constraints. We use the symbol $\succ$ to refer to the strict order relation and $\perp$ to refer to the relation of exclusiveness.

Strict Order Consistency (Strong) If $a \succ b$ in $\mathcal{M}_1$ then $a' \succ b'$ in $\mathcal{M}_2$.

Strict Order Consistency (Weak) If $a \succ b$ in $\mathcal{M}_1$ then $\neg(b' \succ a')$ in $\mathcal{M}_2$.

Exclusiveness Consistency If $a \perp b$ in $\mathcal{M}_1$ then $a' \perp b'$ in $\mathcal{M}_2$.

The strong form of strict order consistency specifies that the order relation among two activities and their counterparts should be identical. The weak form requires that the order relation should not be reversed. For more details we refer the reader to [2].

5. A Markov Logic Implementation

To solve the problem of selecting the most probable subset from $\mathcal{H}$ fulfilling the constraints introduced in the previous section, we define a Markov Logic-based optimization approach. Markov Logic (ML) [18] offers a framework for combining probabilities and first-order logic. A Markov Logic formalization is a set of weighted and unweighted first-order logic formulae. Under a set of constants, a Markov Logic formalization of a concrete problem instantiates into a ground Markov network where every node is a binary random variable called a *ground atom*. Bayesian Networks are a special case of Markov networks, where the network is a directed acyclic graph. Markov networks are undirected graphs where edges represent correlations instead of causality. Undirected graphs are required for our case, because the nodes are in mutual dependencies.

Given such a network, it is possible to apply a-posteriori inference, in short MAP inference. MAP inference computes the most probable assignment of truth values to all nodes in the network such that the sum of weights attached to these nodes is maximized and assignments violating unweighted formulae are singled out. According to [18], the probability of a world x is defined as

$$P(X = x) = \frac{1}{Z} exp(\sum_i w_i n_i(x))$$

where i in an index that refers to one of the given formulae, w_i is the weight attached to this formulae, $n_i(x)$ is the number of true groundings of the formulae in x, and Z is the normalizing factor.

The MAP state corresponds to the most probable world, that is, the world for which the sum of all weights associated to the true formulae is maximal. We can leverage this for the ensemble matching problem by weighing all matching hypotheses from $\mathcal{H}$ by the number of votes that the hypothesis received from the ensemble. In doing so, we can use a MAP inference engine for Markov logic in order to compute the consistent sub set from $\mathcal{H}$ that has the highest number of votes. In the following, we describe the different types of formulae of our Markov Logic formalization step by step by referring to the matching example from Figure 1.

We first add all hypotheses from $\mathcal{H}$ as weighted formulae, where the weights attached to the correspondences are the votes for that correspondence. Here are two examples from Figure 1. The subscript indicates the origin ($\mathcal{M}_1$ vs. $\mathcal{M}_2$) of the activity.

$$\texttt{map}(\mathit{ReceiveReject}_1, \mathit{ReceiveReject}_2), 14$$
$$\texttt{map}(\mathit{ReceiveAccept}_1, \mathit{ReceiveCommitment}_2), 4$$

The formulae using the `map` predicate are weighted formulae. We describe the strict order relation and exclusiveness with the help of the predicates `pre` (precedes) and `ex` (exclusive). The following formulae represent examples.

$$\texttt{pre}(\mathit{SendApplication}_1, \mathit{TakeInterview}_1)$$
$$\texttt{ex}(\mathit{WaitForResponse}_2, \mathit{ReceiveRejection}_2)$$

In a similar way, we state which activities are executed by a particular resource. If a resource `r` performs an activity `a`, we add the formulae `re(r,a)`.

So far we have shown how to formalize the observations related to the correspondences in $\mathcal{H}$ and the structural relations in $\mathcal{M}_1$ and $\mathcal{M}_2$. Now we have to present the formulae to model cardinality and consistency constraints that we described in the previous section. We start with the one-to-one constraint.

$$\texttt{map}(x, y) \wedge \texttt{map}(x, y') \rightarrow y = y'$$
$$\texttt{map}(x, y) \wedge \texttt{map}(x', y) \rightarrow x = x'$$

The one-to-n constraint corresponds to a rule that forbids a combination of correspondences as depicted in

Case (iv) of Figure 3.

$$\texttt{map}(x,y) \wedge \texttt{map}(x',y) \ \wedge \texttt{map}(x',y') \wedge x \neq x' \rightarrow y = y'$$

We depicted the variant for ensuring the one-to-n constraint for activity correspondences. Using the predicate `mapr` to express a correspondence between resources, one can apply the constraint also to resource alignments by replacing `map` by `mapr` in the formula given above. The modified rule ensures that a resource alignment is consistent as defined in Section 4.2. However, it is still required to add the following rule that describes the induction of a resource correspondence from an activity correspondence

$$\texttt{map}(x,y) \wedge \texttt{re}(rx,x) \wedge \texttt{re}(ry,y) \rightarrow \texttt{mapr}(rx,ry)$$

The strict order constraint in its weak (first formulae) and strong form (second formulae) can be expressed as follows.

$$\texttt{map}(x,y) \wedge \texttt{map}(x',y') \wedge \texttt{pre}(x,x') \rightarrow \neg\texttt{pre}(y',y)$$

$$\texttt{map}(x,y) \wedge \texttt{map}(x',y') \wedge \texttt{pre}(x,x') \rightarrow \texttt{pre}(y,y')$$

The formula for the exclusiveness constraints is similar to the second formula. Due to the lack of space, we omit its presentation. In our experiments we will first choose the cardinality constraint that corresponds to the task that needs to be solved (equivalence vs. subsumption correspondences). We will also experiments with settings activating/deactivating some of the other constraints. For each setting we will then compute the MAP state, which is the subset of $\mathcal{H}$ with the maximal number of votes that is consistent against the activated constraints. With one exception we apply the constraints in our experiments as hard constraints, which means that they cannot be violated. In one setting we will also report about results for a weighted variant of the weak strict order constraint. Attaching positive weights to general formulae can be understood as injecting votes to the optimization problems for every combination of correspondences that fulfills the constraint.

For each matching problem, we add the general formulae and generate the problem specific input. Then we use the system RockIt [19], which has been proved to be the most efficient Markov logic solver, to compute the MAP state.

6. Experimental Results

In Section 6.1 we explain the experimental setup and the datasets we use in our experiments. In Section 6.2 we present the results of our experiments. A detailed analysis can be found in Section 7.

University Admission (UA)				Birth Registration (BR)				Asset Management (AM)			
Matcher	P	R	F	Matcher	P	R	F	Matcher	P	R	F
RMM-NHCM	.691	.655	.673	OPBOT	.713	.468	.565	AML-PM	.786	.595	.677
RMM-NLM	.768	.543	.636	pPalm-DS	.502	.422	.459	RMM-NHCM	.957	.505	.661
Match-SSS	.807	.487	.608	RMM-NHCM	.727	.333	.456	RMM-NLM	.991	.486	.653
OPBOT	.603	.608	.605	RMM-VM2	.474	.400	.433	BPLangMatch	.758	.563	.646
KMatch-SSS	.513	.578	.544	BPLangMatch	.645	.309	.418	OPBOT	.662	.617	.639
RMM-SMSL	.511	.578	.543	AML-PM	.423	.365	.392	Match-SSS	.897	.473	.619
TripleS	.487	.483	.485	KMatch-SSS	.800	.254	.385	RMM-VM2	.676	.545	.603
BPLangMatch	.368	.44	.401	RMM-SMSL	.508	.309	.384	KMatch-SSS	.643	.527	.579
KnoMa-Proc	.337	.474	.394	TripleS	.613	.280	.384	TripleS	.614	.545	.578
AML-PM	.269	.672	.385	Match-SSS	.922	.202	.332	pPalm-DS	.394	.595	.474
RMM-VM2	.216	.47	.296	RMM-NLM	.859	.189	.309	KnoMa-Proc	.271	.514	.355
pPalm-DS	.162	.578	.253	KnoMa-Proc	.234	.297	.262	RMM-SMSL	.722	.234	.354

Table 1: Participants of the Matching Contest 2015 and their results for the UA, BR, and AM dataset.

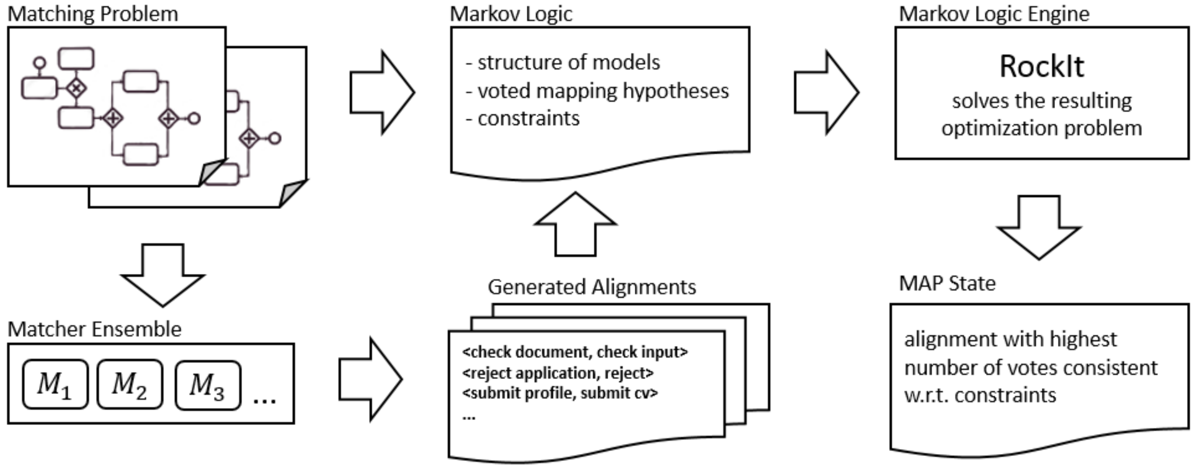
6.1. Experimental Setup

For our experiments we used the University Admission (UA), the Birth Registration (BR), and the Asset Management (AM) dataset from the Process Model Matching Contest (PMMC) 2015 [13]. Since the datasets have been used in the PMMC, we have access to a large set of mappings generated by state-of-the-art process model matching systems. Moreover, we had access to the corresponding gold standard. It is based on three independent assessments of matching experts. In case of disagreement, the experts discussed each specific case to decide if it is included to the gold standard. Therefore, the gold standard can be assumed to be of adequate quality.

The UA dataset consists of nine BPMN models with varying sizes of 10 to 44 activities, which cover the admission process from different German universities. The dataset includes a resource perspective in terms of pools and lanes.[1] The BR dataset consists of 36 model pairs that were derived from 9 models representing the birth registration processes of Germany, Russia, South Africa, and the Netherlands. The

[1] When working with the UA dataset, we detected incorrect correspondence in the reference alignments. We corrected this mistake and reported it to the organizers of the contest. The mistake was based on a semi-automated translation from a label-based-representation of the gold standard (originally created by the people who set up the gold standard manually) to an id-based-representation. In the label based representation there was an entry *Receive Application = Receive Application*. However, *Receive Application* appeared twice as label for different activities in one of the models. During the automated conversion to id-based correspondences sometimes the wrong one was chosen.

Figure 5: Architecture


Matching Problem
Markov Logic
- structure of models
- voted mapping hypotheses
- constraints
Markov Logic Engine
RockIt
solves the resulting optimization problem
Matcher Ensemble
M_1
M_2
M_3
...
Generated Alignments
<check document, check input>
<reject application, reject>
<submit profile, submit cv>
...
MAP State
alignment with highest number of votes consistent w.r.t. constraints

models are available as Petri Nets (PNML). They do not include a resource perspective. The AM dataset is modeled as Event-driven Process Chains (EPC) describing different processes from the area of finance and accounting. It also misses a resource perspective. All datasets as well as the alignments generated by the participants of the contest have been made available by the organizers on the web page of the contest. Note also that the gold standards of the first two datasets were available to the participants, i.e., these evaluation tests were performed as non-blind tests.[2]

The datasets differ with respect to the types of correspondences that are specified in the reference alignments. The reference alignment of the UA dataset contains only equivalence correspondences. The main task with respect to this dataset is to generate correspondences between equivalent activities. This means that we have to activate the one-to-one constraint for the UA dataset. The BR and the AM dataset, however, do not have this restriction. Therefore, both datasets contain correspondences which might express equivalence or subsumption. As a consequence, we have to use the one-to-n constraint. It is worth mentioning that this is not a characteristic of the dataset itself, but a decision that was made when creating the gold standards.

Twelve matchers participated in the 2015 edition of the contest. We computed precision, recall, and f-measure for each participating system. The results are shown in Table 1.

The overall workflow of our approach and the architecture we use for our experiments are depicted in Figure 5. Note that the execution of the matchers from the ensemble is not part of our experiments. We

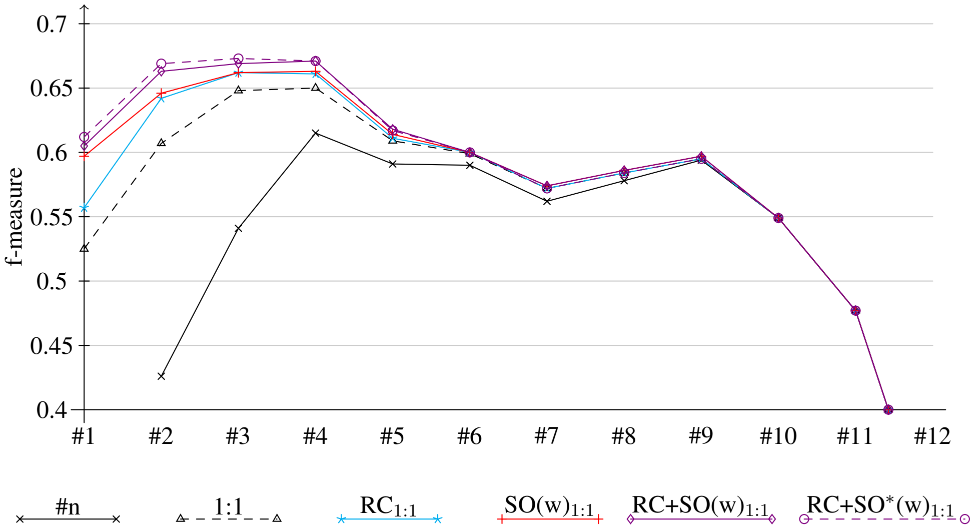
[2] https://ai.wu.ac.at/emisa2015/contest.php

reuse the alignments generated in the context of the PMMC 2015. All other steps of the workflow are completely automated.

6.2. *Results*

In the following we report about our experimental results. We summarize the most important results in Figure 6 (UA), Figure 7 (BR), and Figure 8 (AM). These figures show the results for voting as well as the impact of cardinality, resource constraints (if applicable on the dataset) and the most effective sequential constraint. On the x-axis we distinguish between the #1 to #12-alignment. Each line in a figure refers to a combination of constraints and visualizes their impact on the f-measure (y-axis) for varying n# values. In Figure 6, for example, the red line marked with a + refers to activating the 1:1 constraint together with the weak variant of Strict Order Consistency.

Figure 6: Results for the UA dataset.



Voting. The f-measure of the voting based approach, which is in all figures depicted as a solid black line marked with an x, shows a similar behavior over all datasets. For very low n# values the alignments have a high recall and a low precision, while the opposite holds for high n#, where most of the matchers have to agree on the correctness of a correspondence. For the UA and AM dataset we observed the best values for n#=4 or n#=5. The BR dataset marks an exception. The curve is shifted to the left and the top f-measures are reached at n#=2 and n#=3. The dataset seems to contain more correspondences that are less trivial to detect.

Figure 7: Results for the BR dataset.

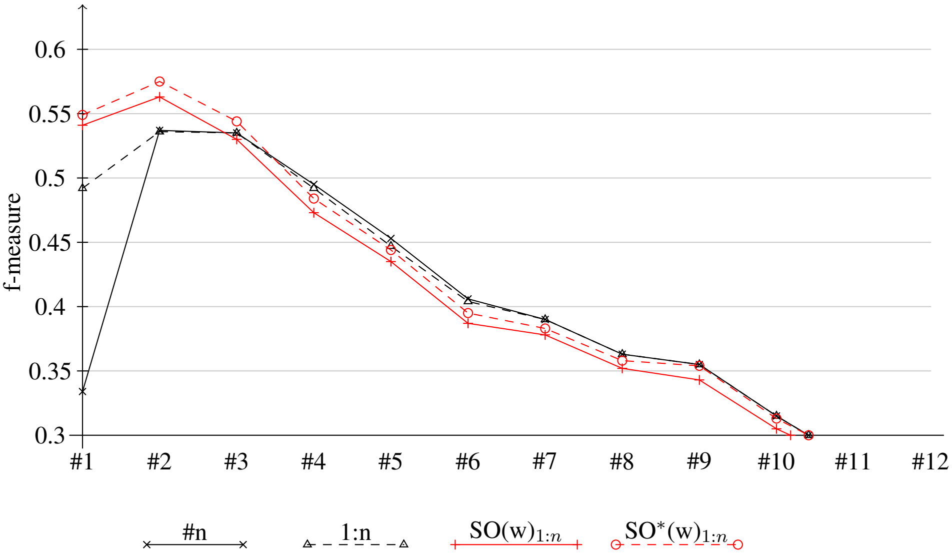
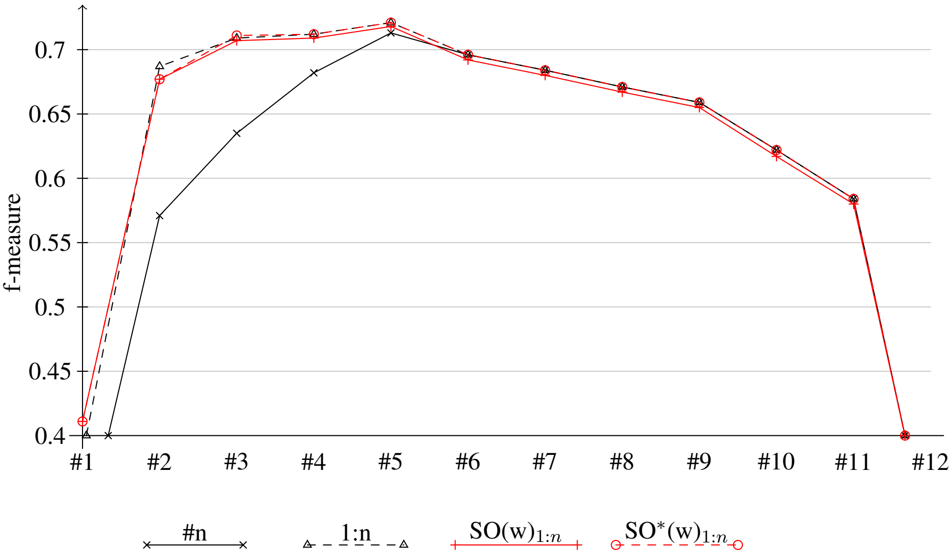


Figure 8: Results for the AM dataset.



Cardinality Constraint. The UA dataset has a gold standard that consists of equivalence correspondences. For this dataset we activate the 1:1 constraint. The BR and AM datasets include also subsumptions. For these datasets we used the 1:n cardinality constraint. The corresponding results are depicted as a dashed black line marked with a triangle. The cardinality constraint has a similar effect for the UA and AM dataset. For low n# values we observe a significant improvement of the resulting f-measure, which is based on

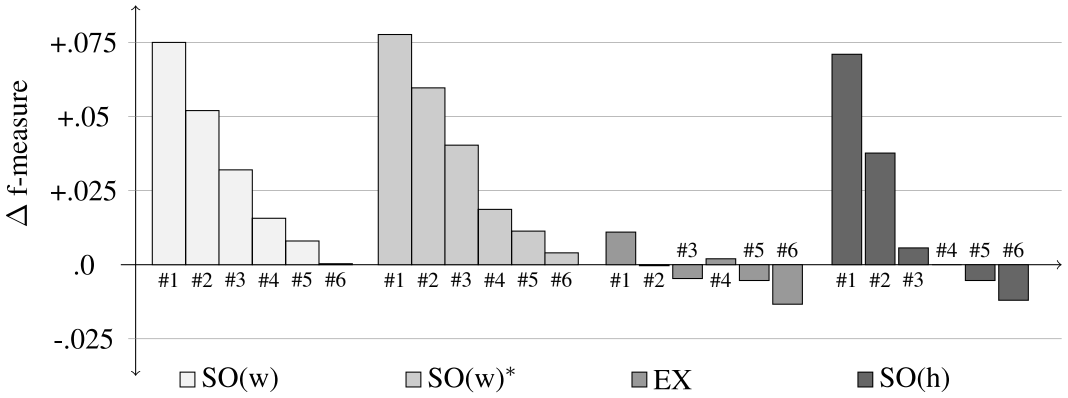


Figure 9: Comparing the impact of sequential constraints for n=#1 to n=#6.

a relatively small loss of recall and a strong improvement of precision. With increasing n# this effect is reduced and for values around n#=6 we cannot observe any impact. The BR dataset has a slightly different characteristic. Improvements can only be observed for n#=1.

Resource Consistency. Resource Consistency (RC) can only be applied to the UA dataset, because the other datasets do not have an associated resource perspective. Enforcing resource consistency results in an improvement of up to 3.5 percentage points (cyan line marked with a star).

Sequence Consistency. In Figures 6, 7, and 8 we have depicted results related to the weak variant of Strict Order Consistency, referred to as SO(w). Showing results for all sequence-based constraints within these figures would make it hard to distinguish between the plotted lines. For that reason we present aggregated results in Figure 9. We have conducted experiments using Strict Order Consistency SO(w=weak), SO(h=hard) and Exclusiveness (EX). Moreover, we have also used Strict Order Consistency as a weighted soft constraint referred to as SO(w)*. We have chosen the weight 2.5, which means that a correspondence from a pair of conflicting correspondences is only removed, if it has less than 3 votes. Note that there are more complex cases with several correspondences, which are involved in overlapping conflicts. In that case it might also happen that a correspondence with more than 2 votes is removed. Figure 9 shows the impact in terms of F-measure by comparing the results of activating the cardinality constraint only with the results of additionally activating one of the four sequential constraints.

For all variants of Strict Order Consistency we observe a similar trend. The positive influence is rather high for low #n values, and decreases down to #n=6. The hard variant of the constraint has even a negative impact for alignments that are already relatively precise, i.e., for high #n values. The weighted version of the weak variants achieves the best results, however, these results are only slightly better than the non-weighted version. The impact on the F-measure is always based on an increased precision and a decreased recall. The results for Exclusiveness are mixed. Overall, the constraint is too aggressive and cannot improve the results. We have also conducted experiments with the weighted variant of SO(h) and EX that are not depicted here. However, the results improve only slightly compared to using the hard constraints.

Combining Constraints. For the UA dataset we combined the most promising sequential constraints with resource consistency. The results are depicted as the violet lines marked with a diamond (RC+SO) and a circle (RC+SO*). Using the combination of different constraints results mostly in summing up their positive impact.

7. Results Analysis

In the following we analyze the results and present some lessons learned. We start with a global evaluation (Section 7.1), followed by a fine-grained analysis to better understand the impact of the particular constraints (Section 7.2).

7.1. Overall Performance

In Table 2 we compare our results with two baselines. For the first baseline we assume that we have no knowledge about the matching systems within our ensemble. In such a setting we will randomly choose one of the matchers. We computed the average of all matchers, denoted as *Random Choice*, to measure the outcome of this approach. Given additional knowledge about the performance on another dataset, we would obviously choose the system that performs best on this dataset instead of making a random choice. For our second baseline we assume that we know the results for the UA dataset and we apply the best UA matcher to the BR and to the AM dataset. We do the same for the other two combinations averaging over the results. We refer to this baseline as *Best in Test*.

Table 2 depicts also the average values for RMM-NHCM, which is the best matcher from the ensemble averaging over all datasets. We refer to these scores as *Best in Average*. However, before running all matchers and comparing their results to the gold standard, we do not know which of them will generate the best average results. Furthermore, we present results of the best matcher for each dataset, referred to as *Best per Dataset*. Note that it is highly unrealistic that we are able to select the best matcher for each

Baseline / Setting		Precision	Recall	F-measure
Random Choice		.597	.461	.484
Best in Test		.606	.516	.523
Best in Average *(a posteriori)*		.0.792	.498	.597
Best per Dataset *(a posteriori)*		.73	.573	.638
$\text{RC+SO}^*(\text{w})_{1:1/1:n}$	#1	.461	.66	.519
	#2	.676	.632	.634
	#3	.744	.588	.635
	#4	.786	.538	.618
	#5	.811	.485	.59

Table 2: Comparing the ensemble matching approach to baselines and best matcher

dataset a priori. Thus, both *Best in Average* and *Best per Dataset* are not viable alternatives, but inform about characteristic scores.

First of all, our motivating claim, that it is hard or even impossible to make an a priori choice of the best or at least a good matching system, is supported by the observation that we have only a +3.9% improvement from *Random Choice* to *Best in Test*, while there are more than 10% missing to achieve the *Best per Dataset* scores. This is also substantiated by the fact that the best matcher for a certain dataset is never the best for any other dataset (compare with Table 1).

In the last five rows of Table 2 we are presenting the results of applying our approach, using Resource Consistency (RC) and weighted Strict Order Consistency in its weak variant (SO(w)*) together with the cardinality constraint, on the n#=1 to n#=5 alignments. The results show that we are clearly outperforming the *Random Choice* baseline. With the exception of the n#=1 alignments, we are also outperforming the *Best in Test* baseline by 6.7% to 11.2%. This shows also that the approach is robust and that it is not important to find the optimal parameter setting. One might object that we are using a weighted version of the SO(w) constraint. However, we experimented with different weights and for none of the weights we observed a loss of more than 1.5% compared to the presented values.

For the values n#=2 to n#=4 we even outperform the *Best in Average* scores by 2.1% to 3.8% percentage points. Furthermore, for n#=2 and n#=3 there are only 0.4% missing to reach the *Best per Dataset* scores. This means that our method is almost as good as an oracle that will always select the best matcher for a given matching task. The extent of the gap (more than 10%) between *Best in Test* scores and *Best per Dataset* scores illustrates the benefit of such an oracle. We are not proposing a method that selects a matcher from the ensemble but a subset of all correspondences generated by the matchers in the ensemble. However, our method is nearly as good as a method that would offer a perfect solution for the related matcher selection problem.

7.2. Impact per Constraint

Voting and Cardinality Constraints. For all datasets we measured very good results for the voting-based approach combined with cardinality constraints. For UA and AM for #n=2 to #n=6 the f-measure is more than 9 percentage points higher than the *Random Choice* baseline. The best results are obtained for #n=4 for these datasets. With respect to the BR dataset the results are different. Improvements can only be observed for #n=1. This might be related to the fact that the BR dataset contains only few matching alternatives for a given activity. Overall, a large portion of the good results reported in the previous section are caused by the positive impact of the voting-based approach combined with cardinality constraints.

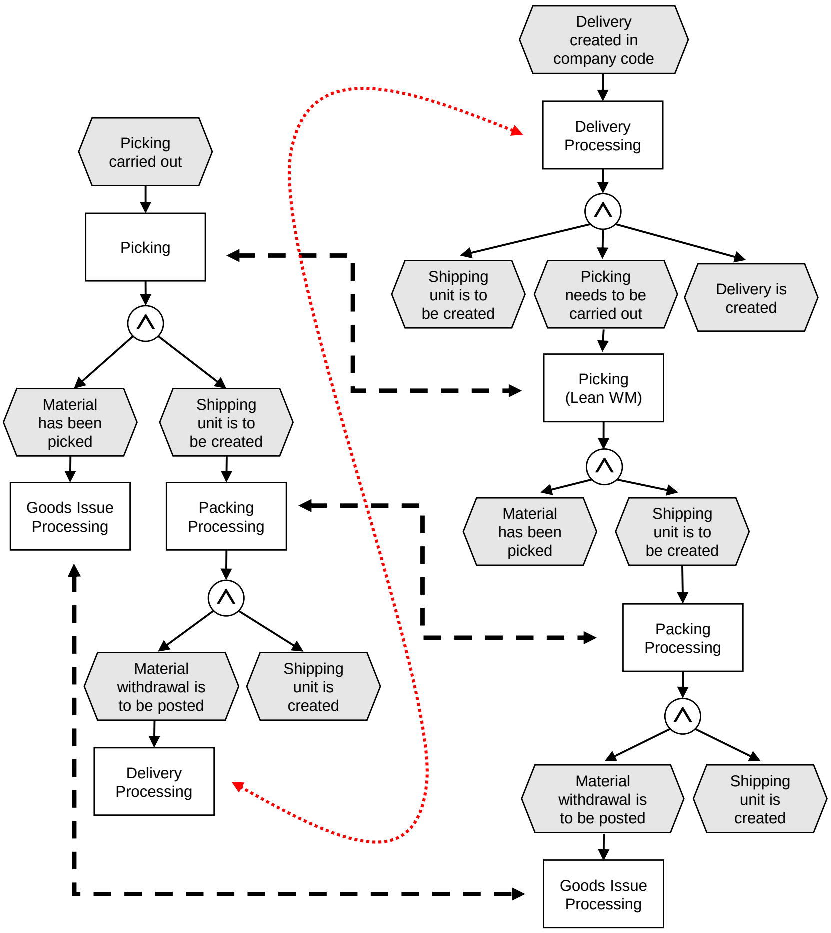
Resource Constraints. The notion of resource consistency can only be applied to business process models that make use of a resource perspective. This is the case for the models from the UA dataset. When applying the corresponding constraint we observed a positive impact resulting in an improved f-measure of up to 3.5 percentage points. Moreover, for none of the testcases the constraint had a negative effect on the results. We conclude that our proposal to leverage the resource perspective is beneficial for the overall results. The corresponding constraint should be activated if such a perspective is available.

Weak Variant of Strict Order. The results for the constraints derived from the sequence flow are less clear. The weak form of the strict order constraint has a positive impact on the UA and BR datasets and a minor negative impact on the AM dataset (a relatively constant loss of around 0.003). We have carefully analyzed this minor deterioration and detected that it is based on one testcase shown in Figure 10.

The problem of our approach is the position of the activity *Delivery Processing*. In one of the models it is an activity at the end of the process, while in the other model it refers to an activity at the beginning. This is, due to the existence of the other three trivial correspondences, a violation of the constraint. Thus, our approach removes the correspondence linking the activities labeled as *Delivery Processing*, because this removal is sufficient to resolve the conflict.

By taking a closer look, it becomes obvious that *Delivery Processing* in the model on the left side refers to a concrete activity, while the activity in the model on the right side refers to an activity that starts an overall process that will finally end with a delivery and this process is labeled *Delivery Processing*. If this is the appropriate interpretation, it means that *Delivery Processing* is an activity that includes the other

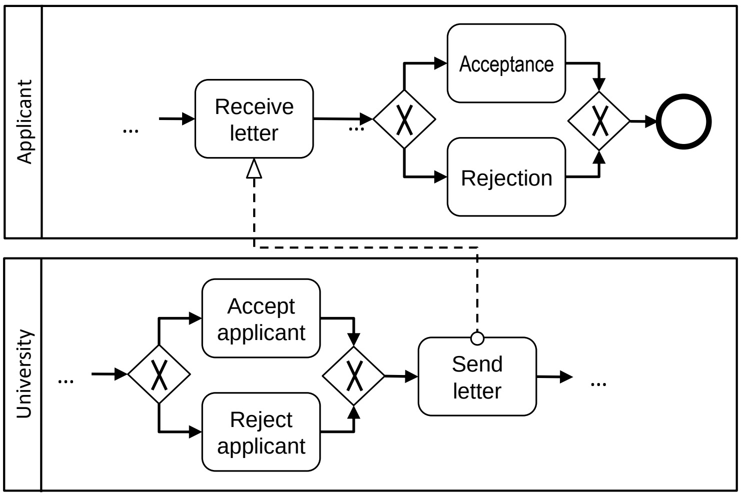
Figure 10: One of the problematic testcases from the AM dataset.



activities of the model as sub activities, which conflicts with the sequential order defined in the model. In an alternative interpretation, the activity might also refer to an interaction with the ERP system that results in the creation of a delivery transaction. However, in both interpretations it does not refer to the concrete delivery, which makes the *Delivery Processing* correspondence doubtful. Our observations are partially supported by the error analysis presented in [20], where the authors applied a structural analysis on the models of the SAP reference model. The AM dataset is a subset of this collection. The authors reported about a surprisingly high number of errors. Such errors are also a problem of our approach, because it builds on the correctness of the matched models.

In general, we recommend to activate the weak variant of the strict order constraint. If we aggregate the results for all datasets, we have a positive impact for low #n values, which still helps to improve the overall quality of results for the most reasonable choice of #n from #2 to #4 (see Figure 9 and Table 2). Moreover,

Figure 11: Example for (factual) exclusive activities (e.g., *Reject applicant* and *Acceptance*) not modeled explicitly as exclusive activities.



the constraint can be used to automatically detect interesting differences in the matched models. We have illustrated this with the help of the previous example, which might be useful in application scenarios where matching is used to detect differences in business processes.

Exclusiveness and Strong Variant of Strict Order. Exclusiveness and the strong variant of strict order have a mixed impact. To better understand these results we have to analyze the models and alignments of the concrete testcases. This reveals that problems related to the exclusiveness constraint are sometimes caused by an imprecise modeling. In particular, it can happen that actually exclusive activities are modeled as non-exclusive activities in the process model. Figure 11 depicts a (simplified) example from the UA dataset, where the activities *Reject applicant* and *Acceptance* are factually exclusive. However, they are not defined as exclusive activities. Matching this model to a model in which the corresponding activities are correctly modeled as exclusive activities, results in a violation of the exclusiveness constraint.

The strong variant of the strict order consistency achieves only slightly worse results than the weak variant (compare with Figure 9). By inspecting the results in detail, we detected that weak and strong variant have the same impact for many testcases. This is not a surprise, because the weak variant is logically entailed by the strong variant. In those testcases where we observed differences, this was often a deterioration caused by a more aggressive behavior of the strong variant. Such a deterioration occurred especially if sequential order is specified where a parallel execution of activities would be more appropriate. We conclude that exclusiveness and the strong form of the strict order constraint should not be applied in general, because we cannot expect a high degree of modeling precision in many application domains.

Previous Results. At first glimpse our results differ from the results reported in [2], where the authors measured a general positive impact for all sequence based constraints. They report about results around ≈0.318 in terms of f-measure, which is an improvement compared to the input. The positive impact is mostly based on improving the precision up to 0.421. Comparing these numbers with our results, it becomes obvious that the input to the experiments has been of limited quality. For such a setting the negative impact caused by different sequential structures is outnumbered by the positive impact of filtering out incorrect correspondences. Similar results can be seen for the #n=1 alignments in our setting, where almost all sequential constraints have a positive impact. However, in our experiments we observed also the negative impact for the exclusiveness constraint and the strong variant of strict order when applied to better input alignments (starting from #n=3). This is the more relevant assessment given the quality of current state-of-the-art matching techniques.

8. Related Work

Research in the field of process model matching has been long concerned with the definition of individual matchers. Most of them combine structural or behavioral properties with a textual similarity measure. Some matchers rely on simplistic techniques such as the Levenshtein distance [21], while others build on the lexical database WordNet [22] to also take semantic relationships between words into account [2, 4]. More recent techniques also consider the identification of correspondences between process models and other process-related artifacts. For instance, there are techniques identifying correspondences between process models and activity taxonomies [23], textual process descriptions [24], and process performance indicators [25].

Since our approach focuses on selecting a suitable subset from the correspondences generated by a matcher ensemble, in particular the existing process model matching techniques represent an important ingredient for our technique. The approach that first introduced the notion of combining different components and then selecting a subset of correspondences is the ICoP Framework [1]. However, the ICoP Framework does not consider the correspondences generated by several matchers. Also it does not perform better than more recently proposed matching systems [26]. Recognizing the limitations of many matchers in terms of performance, researchers recently started to explore alternative venues for solving the matching task. For instance, Klinkmüller et al. [27] proposed a semi-automated technique that incorporates user feedback to improve the matching performance. Weidlich et al. [28] defined a technique that aims at predicting the performance of a matcher based on the similarity values it produces for a given matching problem. In this way, the approach can help to select the most suitable matching technique for the problem at hand. Our approach pursues a different strategy by running all available matchers and then selecting the best correspondences. This strategy has the advantage of combining the strengths of multiple matching systems.

The foundations of our approach can be found in the fields of ontology and schema matching [29, 30]. In both domains ensemble matching has been recognized as a strategy to define robust matching systems. For instance, Eckert et al. propose a technique that builds on the majority vote to identify matching elements from ontologies [14]. Gal and Sagi propose an approach for schema matching that chooses among available matchers and automatically tunes their individual importance [31]. We build on the concepts defined in these approaches and extend them with process-model-specific selection mechanisms. Our evaluation experiments demonstrated that these constraint-based mechanisms significantly contribute to the performance of our ensemble matching approach.

9. Conclusion and Future Work

In this paper, we addressed the problem of the varying performance of individual process model matching techniques. To this end, we introduced an ensemble matching approach that uses the correspondences generated by a set of matchers as input. The core of our technique is formed by a Markov Logic based optimization approach that automatically selects the best correspondences. Our technique combines a generalized form of a voting mechanism and a number of process-model-specific constraints. The results of our evaluation experiments illustrate that our approach outperforms alternative approaches significantly. Moreover, the results are even close to the results of the best matcher per dataset, which is not known prior to comparing the results with the gold standard.

Within our work we have been using cardinality constraints, a constraint which is leveraging the resource perspective, and several sequential constraints. The applicability of the resource-based constraint relies on the availability of the resource perspective, which is missing in two of the datasets we used for our experiments. However, common modeling notations used in industry, such as EPCs and BPMN, do provide this perspective. Therefore, we believe that a selection mechanism based on the resource perspective is a viable choice. The results we obtained for the sequence-based constraints were mixed and we argued that this is on the one hand related to factual differences in the processes and on the other hand related to modeling inaccuracies.

In particular the combination of the voting-based approach with the appropriate cardinality constraint works surprisingly well. Aside from the straight-forward one-to-one constraint, we also applied a one-to-n constraint that is defined in terms of a difunctional relation. The results illustrate that this constraint is the appropriate counterpart to the one-to-n constraint if the given task is to generate both equivalence and subsumption correspondences. This is an important contribution of this work. However, sometimes more complex semantic relations are required to express interesting links between activities. Such relations might result in n-to-m alignments, which are not yet supported by our approach. As part of our future work we plan to develop a formalism that supports different semantic relations with relation-specific cardinality constraints.

10. References

[1] M. Weidlich, R. Dijkman, J. Mendling, The ICoP framework: Identification of correspondences between process models, in: Advanced Information Systems Engineering, Springer, 2010, pp. 483–498.

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

[3] M. Weidlich, E. Sheetrit, M. C. Branco, A. Gal, Matching business process models using positional passage-based language models, in: Conceptual Modeling, Springer, 2013, pp. 130–137.

[4] 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, Springer, 2013, pp. 211–218.

[5] U. Cayoglu, A. Oberweis, A. Schoknecht, M. Ullrich, Triple-s: A matching approach for Petri nets on syntactic, semantic and structural level, Tech. rep., Karlsruhe Institute of Technology (KIT) (2013).

[6] M. Dumas, L. García-Bañuelos, R. M. Dijkman, Similarity search of business process models., IEEE Data Eng. Bull. 32 (3) (2009) 23–28.

[7] M. Kunze, M. Weidlich, M. Weske, Behavioral similarity–a proper metric, in: Business Process Management, Springer, 2011, pp. 166–181.

[8] T. Jin, J. Wang, M. La Rosa, A. Ter Hofstede, L. Wen, Efficient querying of large process model repositories, Computers in Industry 64 (1) (2013) 41–49.

[9] C. C. Ekanayake, M. Dumas, L. García-Bañuelos, M. La Rosa, A. H. ter Hofstede, Approximate clone detection in repositories of business process models, in: Business Process Management, Springer, 2012, pp. 302–318.

[10] R. Uba, M. Dumas, L. García-Bañuelos, M. La Rosa, Clone detection in repositories of business process models, in: Business Process Management, Springer, 2011, pp. 248–264.

[11] M. La Rosa, M. Dumas, R. Uba, R. Dijkman, Business process model merging: An approach to business process consolidation, ACM Transactions on Software Engineering and Methodology (TOSEM) 22 (2) (2013) 11.

[12] M. Weidlich, J. Mendling, M. Weske, A foundational approach for managing process variability, in: International Conference on Advanced Information Systems Engineering, Springer, 2011, pp. 267–282.

[13] G. Antunes, M. Bakhshandeh, J. Borbinha, J. Cardoso, S. Dadashnia, C. D. 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. Péus, A. Schoknecht, E. Sheetrit, A. Sonntag, H. Stuckenschmidt, T. Thaler, I. Weber, M. Weidlich, The process model matching contest 2015, in: 6th International Workshop on Enterprise Modelling and Information Systems Architectures, 2015.

[14] K. Eckert, C. Meilicke, H. Stuckenschmidt, Improving ontology matching using meta-level learning, in: The Semantic Web: Research and Applications, Springer, 2009, pp. 158–172.

[15] C. Meilicke, H. Stuckenschmidt, Analyzing mapping extraction approaches, in: Proceedings of the 2nd International Workshop on Ontology Matching (OM-2007), 2007.

[16] R. Khchérif, M. M. Gammoudi, A. Jaoua, Using difunctional relations in information organization, Information Sciences 125 (1) (2000) 153–166.

[17] M. Baumann, M. H. Baumann, S. Schönig, S. Jablonski, Resource-aware process model similarity matching, in: Service-Oriented Computing-ICSOC 2014 Workshops, Springer, 2015, pp. 96–107.

[18] M. Richardson, P. Domingos, Markov logic networks, Machine Learning 62 (1-2). doi:10.1007/s10994-006-5833-1.

[19] J. Noessner, M. Niepert, H. Stuckenschmidt, Rockit: Exploiting parallelism and symmetry for MAP inference in statistical relational models, in: Proceedings of the Twenty-Seventh Conference on Artificial Intelligence, 2013.

[20] J. Mendling, H. Verbeek, B. F. van Dongen, W. van der Aalst, G. Neumann, Detection and prediction of errors in epcs of the sap reference model, Data & Knowledge Engineering 64 (1) (2008) 312–329.

[21] R. Dijkman, M. Dumas, L. García-Bañuelos, Graph matching algorithms for business process model similarity search, in: International Conference on Business Process Management, Springer, 2009, pp. 48–63.

[22] G. A. Miller, WordNet: a lexical database for english, Communications of the ACM 38 (11) (1995) 39–41.

[23] H. Leopold, C. Meilicke, M. Fellmann, F. Pittke, H. Stuckenschmidt, J. Mendling, Towards the automated annotation of process models, in: International Conference on Advanced Information Systems Engineering, Springer, 2015, pp. 401–416.

[24] H. van der Aa, H. Leopold, H. A. Reijers, Detecting inconsistencies between process models and textual descriptions, in: International Conference on Business Process Management, Springer, 2015, pp. 90–105.

[25] H. van der Aa, A. del Río-Ortega, M. Resinas, H. Leopold, A. Ruiz-Cortés, J. Mendling, H. A. Reijers, Narrowing the business-it gap in process performance measurement, in: International Conference on Advanced Information Systems Engineering, Springer, 2016, pp. 543–557.

[26] U. Cayoglu, R. Dijkman, M. Dumas, P. Fettke, L. Garcıa-Banuelos, P. Hake, C. Klinkmüller, H. Leopold, A. Ludwig, P. Loos, et al., The process model matching contest 2013, in: 4th International Workshop on Process Model Collections: Management and Reuse (PMC-MR'13), 2013.

[27] C. Klinkmüller, H. Leopold, I. Weber, J. Mendling, A. Ludwig, Listen to me: Improving process model matching through user feedback, in: Business Process Management, Springer, 2014, pp. 84–100.

[28] M. Weidlich, T. Sagi, H. Leopold, A. Gal, J. Mendling, Predicting the quality of process model matching, in: Business Process Management, Springer, 2013, pp. 203–210.

[29] A. Gal, Uncertain schema matching, Synthesis Lectures on Data Management 3 (1) (2011) 1–97.

[30] P. Shvaiko, J. Euzenat, Ontology matching: state of the art and future challenges, Knowledge and Data Engineering, IEEE

Transactions on 25 (1) (2013) 158–176.

[31] A. Gal, T. Sagi, Tuning the ensemble selection process of schema matchers, Information Systems 35 (8) (2010) 845 – 859.