Association for Information Systems
AIS Electronic Library (AISeL)

AMCIS 2012 Proceedings | Proceedings

Identifying Business Process Activity Mappings by Optimizing Behavioral Similarity

Jörg Becker
European Research Center for Information Systems (ERCIS), University of Münster, Münster, Germany., joerg.becker@ercis.uni-muenster.de

Dominic Breuker
European Research Center for Information Systems (ERCIS), University of Münster, Münster, Germany., dominic.breuker@ercis.uni-muenster.de

Patrick Delfmann
European Research Center for Information Systems (ERCIS), University of Münster, Münster, Germany., delfmann@ercis.de

Hanns-Alexander Dietrich
European Research Center for Information Systems (ERCIS), University of Münster, Münster, Germany., hanns-alexander.dietrich@ercis.uni-muenster.de

Matthias Steinhorst
European Research Center for Information Systems (ERCIS), University of Münster, Münster, Germany., matthias.steinhorst@ercis.uni-muenster.de

Identifying Business Process Activity Mappings by Optimizing Behavioral Similarity

Jörg Becker
University of Münster, ERCIS
joerg.becker@ercis.uni-muenster.de

Dominic Breuker
University of Münster, ERCIS
dominic.breuker@ercis.uni-muenster.de

Patrick Delfmann
University of Münster, ERCIS
patrick.delfmann@ercis.uni-muenster.de

Hanns-Alexander Dietrich
University of Münster, ERCIS
hanns-alexander.dietrich@ercis.uni-muenster.de

Matthias Steinhorst
University of Münster, ERCIS
matthias.steinhorst@ercis.uni-muenster.de

ABSTRACT

This paper describes an approach designed to create a mapping between corresponding activities from two business processes that is geared towards handling noisy similarity values for the labels describing these activities. This is achieved by formulating an optimization problem – maximize the behavioral similarity of the processes as a whole – whose target value depends on the mapping. Thereby, the mapping is created not only with respect to label similarities but also with respect to the overall control flow structure, which avoids some mistakes resulting from erroneous label similarities. A preliminary evaluation demonstrates the improvement.

Keywords

Business Process Model Analysis, Behavioral Similarity, Bisimulation

INTRODUCTION

Business Process Diagrams (BPDs) have become an important tool for process oriented management of organizations.Over time, BPD repositories grow to considerable sizes, containing thousands of BPDs (Dumas, Garcia-Banuelos, and Dijkman, 2009). Consequently, the maintenance of such repositories becomes more and more important. Numerous approaches have been proposed in the past to (partly) automate BPD management tasks that would otherwise require vast manual effort.Most of the suggested functionality requires some form of comparison between different models. Such functionality could be searching the repository for BPDs similar to a given pattern model (Dumas, Garcia-Banuelos, and Dijkman, 2009), retrieving BPD fragments that occur often throughout the repository (Uba, Dumas, Garcia-Banuelos, La Rosa, 2011) or merging multiple BPDs into a single one (Mendling and Simon, 2006). A commonality of them all is that certain correspondences between elements from different models must be established. Most importantly, one must know which activities of the one process can be associated with activities of the other. Usually, this is done by analyzing the labels of modeling elements representing these activities.

A remarkable property of many techniques establishing a mapping of elements between two BPDs is that they heavily rely on local aspects of the models. Individual modeling elements are compared pairwise with each other based on their labels. To account for their context, some approaches also include information drawn from preceding or succeeding elements. However, global aspects such as the control flow structure of the entire BPDs are neglected, despite the fact that, for techniques using such mappings, these global aspects usually are an important point of interest. Consider for example business process similarity measurement, in which not only the activities performed in the process determine similarity, but also the way in which they are performed.

Therefore, we present in this paper an alternative to approaches from literature by proposing a procedure integrating both local and global information into the mapping process. This is achieved by defining a similarity measure for BPDs depending simultaneously on label similarities of modeling elements (local) as well as the similarity of the control flow structure as a whole (global). Rather than requiring a mapping as input, it is implicitly constructed by solving an optimization problem. Label similarities incorporated in the constraints ensure that associated activities exhibit a high degree of similarity. At the

same time, noisy and erroneous label similarities do not lead to unreasonable mappings, since dissimilarities of the control flow also drive down the target value. Thus, an optimal embedding of one process into the other is found.

The paper proceeds as follows. We first discuss related work. We then illustrate in detail potential problems occurring when only local information are used to map elements. We subsequently introduce the formal foundation of our approach, which is rooted in automata theory. The following sectionthen formulates the optimization problem and demonstrates how to identify the mapping by inspecting its solution. We then evaluate the approach with respect to different levels of noise in label similarities of activity names. Finally, we give an outlook on future research.

RELATED WORK

Natural Language Processing in Business Process Models

A simple approach to determine similarities of labels is to apply measures like string edit distance to them (Navarro 2011). To deal with synonyms and homonyms, lexical systems like WordNet can be used to identify relationships between words and to compute more accurate similarities (Fellbaum 1998). However, modelers often describe processes by using terms specific to the particular domain of application. To avoid ambiguities, one can define a dictionary of domain terms and enforce its usage during process modeling (Delfmann, Herwig, Lis, 2009). As a consequence, abovementioned natural language processing techniques become applicable again. Going one step further one can also formalize domain knowledge in form of ontologies (Höfferer 2007). By applying techniques for semantic standardization, one can clearly improve the quality of label similarities. However, as their application comes along with significant efforts for creation and maintenance of dictionaries or ontologies, one cannot expect the results to be perfect. Therefore, algorithms for creating element mappings must be robust with respect to noisy similarities.

Business Process Element Mapping

A popular procedure to create a mapping of elements is to first apply several similarity measures to pairs of elements and to subsequently compute a weighted combination of them. The mapping is then chosen such that the sum of combined element similarities is maximal (Ehrig, Koschmider, Oberweis, 2007). Dijkman, Dumas, van Dongen, Käärik, and Mendling (2010) propose a related approach and add two other similarity measures. To avoid mapping elements that are very dissimilar, van Dongen, Dijkman and Mendling (2008) use a cutoff value below which element similarities are treated as zero.A framework able to detect not only 1:1 mappings but also complex (i.e. 1:n) has been developed by Weidlich, Dijkman, and Mendling (2010).

Business Process Similarity Measurement

Given a mapping of activities, there are several ways to define a similarity between processes. For instance, one can create an abstraction of the process behavior to accomplish the comparison. One such abstraction is a causal footprint (Dijkman et al. 2010). Embedding this causal footprint into a vector space allows computing similarities using the well-known cosine similarity. Another measure is the metric proposed by Kunze, Weidlich, and Weske (2011), which uses a behavioral profile containing relations between activities performed in a process. Also based on an abstract representation of behavior is the approach of Zha, Wang, Wen, Wang, Sun (2010), which computes the similarity of matrices containing so called transition adjacency relations specifying if certain activities can follow directly on each other or not.

As BPDs can be interpreted as graphs, methods from graph theory can be applied as well. Based on edit distance, which is the number of certain transformation steps required to transform one graph into another, two BPDs can be compared to each other (Dijkman, Dumas, García-Bañuelos, 2009). Related approaches are proposed by Minor, Tartakovski, and Bergmann (2007) and Li, Reichert, and Wombacher (2008).

Another possibility is to extract so-called features from BPDs and to define a similarity measure based on them. Madhusudan, Zhao and Marshall (2004) apply a similarity flooding algorithm to retrieve workflow cases. As such they develop not a measure of model but of instance similarity, which is accomplished by using a graph structured instance representation. De Medeiros and van der Aalst (2008) use the fitness function of a genetic mining algorithm to measure similarities.

LOCAL AND GLOBAL ELEMENT MAPPING

The last section demonstrated that activity mappings usually are created based on local information, i.e., similarities of pairs of activities. It has also been discussed that the reliability of label similarities strongly depends on the degree of semantic standardization. Avoiding errors completely is almost impossible.

To see a potential problem, consider the example in Figure 1. Assume there are BPDs of two versions of the same human resources candidate selection process. Figure 1 represents an excerpt of each version (two activities each). The first pair of

activities, occurring early in the process, describes an initial review of the candidate's resume or curriculum vitae (CV). Later on in the process, the resume or CV is investigated more intensively (if the candidate is still under consideration), which is described by the second pair of activities. The goal is to correctly associate the activities of both versions of the process with each other.

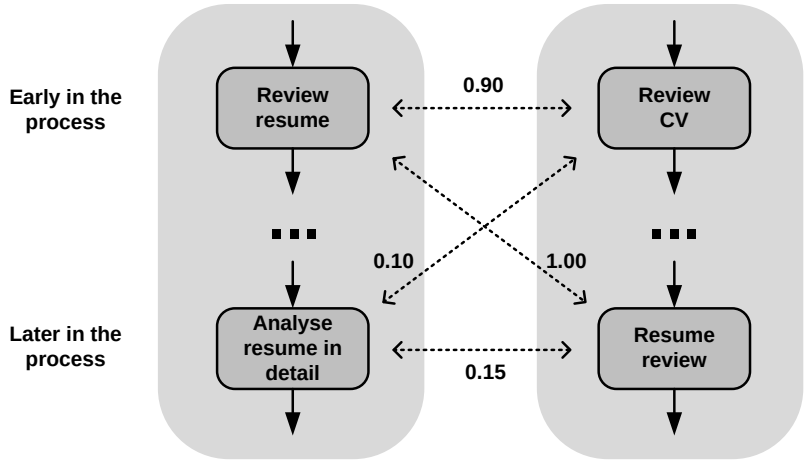


Figure 1: excerpts from two versions of a human resources candidate selection process

Dashed lines represent similarities of activities from different versions as they could result from a label similarity measure. Activities *Review resume*and *Review CV* could be 90% similar as the term *Review* is identical in both cases and the terms *resume* and *CV* are at least synonymous, which could be determined by means of a suitable ontology. Activities *Analyze resume in detail* and *Resume review* do only have the term *resume* in common and thus might be only 10% similar. However, due to the ambiguity of the terms *review* and *resume*, the similarity score of activities *Review resume* and *Resume Review* could equal 100% even though different activities are described.

Matching activities only with respect to their label similarity score would result in associating *Review resume* and *Resume review* with each other. If the mapping would afterwards be used for further analysis, biased or wrong results would occur. The erroneous mapping would let the two BPDs appear as if different processes were represented. To avoid such unreasonable mappings, we propose taking into account not only similarity scores of labels, but at the same time also the control flow structure as a whole. Noticing that *Review resume* and *Resume review*, while being (locally) 100% similar, occur in different phases of the process indicates that they should not be mapped. Associating them drives down the (global) similarity of the two BPDs. Therefore, the idea is to maximize the similarity of two BPDs with the mapping of activities being the "variable" determining this similarity. In the above example, the lower label similarity of *Review resume* and *Review CV* (as compared to *Review resume* and *Resume review*) can then be compensated by a better fit with respect to the control flow. Thereby, the mapping procedure becomes more robust with respect to noisy label similarities.

In the following chapters we will adapt a suitable process similarity measure from literature combining both local and global aspects. Rather than requiring an existing mapping on which the calculation of BPD similarity is based, it requires only label similarities as an input and directly incorporates them into the optimization problem delivering the similarity. By transforming the problem into a linear program, it can be solved efficiently with standard software, even if the problem at hand becomes large. Inspecting the constraints of the linear program allows reconstructing the associations between activities that have been chosen implicitly during optimization.

PRELIMINARY DEFINITIONS

As the similarity measure we adapt is defined on processes represented by labeled transition systems (LTS), we will use this notation throughout the paper. To apply the approach to ordinary BPDs, they must be transformed into LTS form first. In case of a petri net this corresponds to reachability graph generation. An overview on how to transform BPDs created with other notations can be found in Lohmann, Verbeek and Dijkman (2009). An LTS (Cortadella, Kishinevsky, Lavagno, Yakovlev, 1998) is defined as a directed graph $LTS = (S, O, T, s_{in})$ with

- S being a non-empty set of states,
- O being a set of observation symbols or labels,
- $T \subseteq S \times O \times S$ being a set of transitions between states, emitting labels
- $s_{in} \in S$ being an initial state.

We denote a transition $s \xrightarrow{o} s'$ instead of (s, o, s'). If no transition leaving state $s$and labeled by o exists, we say$s \overset{o}{\nrightarrow}$.In the opposite case, we say$s \xrightarrow{o}$.

A theoretical notion to compare LTSs is called bisimulation. Two LTSs are called bisimilar if each one can simulate the other, i.e., if their behavior could not be distinguished by an observer. Inspired by this notion of equivalence, it is possible to define similarity measures quantifying the degree to which two LTSs are bisimilar.

In this paper, we adopt the measure of Sokolski and Kannan(2006). First, they define the degree to which a state of one LTS simulates a state of another. Let $LTS_1 = (S_1, O_1, T_1, s_{in_1})$ and $LTS_2 = (S_2, O_2, T_2, s_{in_2})$ be two LTSs and let $s_i \in S_1$ and $s_j \in S_2$. The degree to which s_i is simulated by s_j (quantitative simulation) is defined by

$$Q(s_i, s_j) = \begin{cases} 1 & if\ \forall o, s_i \overset{o}{\nrightarrow} \\ \max(W_1, W_2) & otherwise \end{cases} \quad (1)$$

where

$$W_1 = \max_{s_j \xrightarrow{b} s_m} p \cdot Q(s_i, s_m) \quad (2)$$

$$W_2 = \frac{1}{n} \cdot \sum_{s_i \xrightarrow{a} s_k} \max\left(\max_{s_j \xrightarrow{b} s_m} L(a,b) \cdot Q(s_k, s_m), p \cdot Q(s_k, s_j) \right) \quad (3)$$

and n being the number of outgoing transitions of s_1. In this definition, $L : O_1 \times O_2 \to [0,1]$ is a function assigning a similarity score to each pair of labels and $p \in]0,1[$ is a parameter, called the gap penalty. The degree of bisimulation of two LTSs is defined as the minimum degree of simulation between their initial states:

$$B(LTS_1, LTS_2) = \min\left(Q(s_{in_1}, s_{in_2}), Q(s_{in_2}, s_{in_1}) \right). \quad (4)$$

The intuition behind these recursive definitions is explained as follows. First, any state s_j can simulate a state s_i perfectly if s_i does not have any outgoing transitions (upper case of Equation 1). The rationale is that if the process to be simulated has reached an end, the simulating process can mimic it by performing no action at all. In all other cases s_i has outgoing transitions which must be simulated by s_j. The main idea of the measure is to let both processes perform transitions either together or separately. The degree of simulation prior to performing a transition is recursively defined as the degree of simulation after it has been performed, reduced by a certain factor. In case two transitions are performed together, this factor is the similarity of the corresponding labels. If only one process transitions to a new state, the reducing factor is the gap penalty$p$.

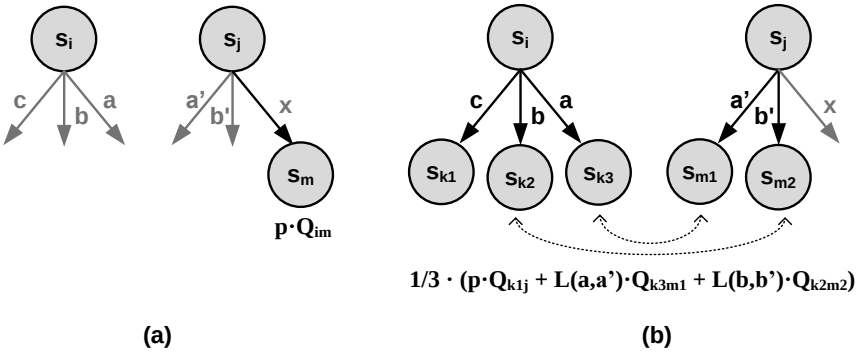


Figure 2: Exemplary illustration of recursive alternatives for computing the quantitative simulation of one state by another. (a) represents skipping ins_i, (b) the opposite

Equation 2 represents the case in which the simulating process "skips a step", i.e., it performs a transition labeled b moving from s_j to s_m which is not meant to mimic one of s_i. Instead, it is meant to pass the responsibility of simulating s_i on to state

s_m. The symbol $b$emitted on the way causes dissimilarity since there is no counterpart for it. This is accounted for by multiplying $Q(s_i, s_m)$ with p.

Equation 3 represents the case thats_j is used to simulate at least some of the behavior observable in s_i. This means one iteratively inspects all transitions leaving s_i. For each of them, one either finds a suitable transition leaving the simulating state s_j or one lets the simulated process "skip a step" by letting it enter a new state s_k without simulating this transition. Again, this is punished multiplicatively by p. The overall degree $Q(s_i, s_j)$ in this case is the arithmetic mean of the best of the above described choices for each of the transitions leaving s_i.

Figure 2 illustrates the different alternatives. In Figure 2 (a), s_i has three outgoing transitions labeled by a, b and c. State s_j has three outgoing transitions as well, labeled a', b' and x. Assuming that none of the labels is similar enough to simulate another one, the second LTS moves to s_m, which might have more suitable transitions, while the first LTS does nothing. If we assume labels a and a' as well as b and b' are sufficiently similar, we might encounter a situation depicted in Figure 2 (b) instead. Then, the corresponding transitions are used for simulation. However, there is no transition leaving s_j equipped with a suitable label to simulate the transition labeled c. Consequently, transition x remains unused, while, for the transition labeled c, the second LTS remains in state s_j.

BUSINESS PROCESS SIMILARITY AND ACTIVITY MAPPING

Computation of Quantitative Bisimulation

To compute quantitative simulation one could follow equations 1-3 recursively. However, if there are loops in the processes, this approach is rendered infeasible. Fortunately, the problem can be formulated as a linear program simultaneously computing the degree of quantitative bisimulation between any pair of states (Sokolski and Kannan 2006).

$$\min \sum_{s_i \in S_1, s_j \in S_2} Q_{i,j} \qquad \textbf{(5)}$$

$$Q_{i,j} \geq p \cdot Q_{i,m} \quad \text{for each } s_i \in S_1, s_j \in S_2 \text{ and } s_j \xrightarrow{b} s_m \in T_2 \qquad \textbf{(6)}$$

$$Q_{i,j} \geq \frac{1}{n} \sum_{e = s_i \xrightarrow{a} s_k \in T_1} X_{e,j} \quad \text{for each } s_i \in S_1 : \exists\, a \in O_1 : s_i \xrightarrow{a} \text{ and } s_j \in S_2 \qquad \textbf{(7)}$$

$$X_{e,j} \geq p \cdot Q_{k,j} \quad \text{for each } e = s_i \xrightarrow{a} s_k \in T_1 \text{ and } s_j \in S_2 \qquad \textbf{(8)}$$

$$X_{e,j} \geq L(a,b) \cdot Q_{k,m} \quad \text{for each } e = s_i \xrightarrow{a} s_k \in T_1 \text{ and } s_j \xrightarrow{b} s_m \in T_2 \qquad \textbf{(9)}$$

$$0 \leq Q_{i,j} \leq 1 \quad \text{for each } s_i \in S_1 \text{ and } s_j \in S_2 \qquad \textbf{(10)}$$

$$0 \leq X_{e,j} \leq 1 \quad \text{for each } e = s_i \xrightarrow{a} s_k \in T_1 \text{ and } s_j \in S_2 \qquad \textbf{(11)}$$

$$Q_{i,j} = 1 \quad \text{for each } s_i \in S_1 : \forall\, a \in O_1 : s_i \overset{a}{\nrightarrow} \text{ and } s_j \in S_2\,. \qquad \textbf{(12)}$$

Once solved, we can associate $Q(s_i, s_j)$ with $Q_{i,j}$. The equality constraints defined by Equation 12 ensure that $Q(s_i, s_j) = 1$ whenever a state s_i has no outgoing transitions. All inequality constraints defined by Equations 6-9 correspond to choices that are made during the calculation of quantitative simulation. For instance, the constraints defined by Equation 6 ensure that $Q(s_i, s_j)$ is at least as large as $Q(s_i, s_m)$, for the best state s_m the second LTS could move to while the first LTS does nothing. Equation 7 ensures that $Q(s_i, s_j)$ is also at least as large as the average of all $X_{e,j}$, where e is an outgoing transition of s_i. $X_{e,j}$ are artificial variables representing components of the sum defined by equation 3 (compare constraints defined by equations 8 and 9 to see that). Thus, equations 6 and 7 ensure that $Q(s_i, s_j)$ amounts at least to the degree to which s_j simulates s_i. As $Q(s_i, s_j)$ is being minimized, it amounts to exactly the degree of simulation. See Sokolski and Kannan (2006) for a proof that such a linear program has exactly one solution.

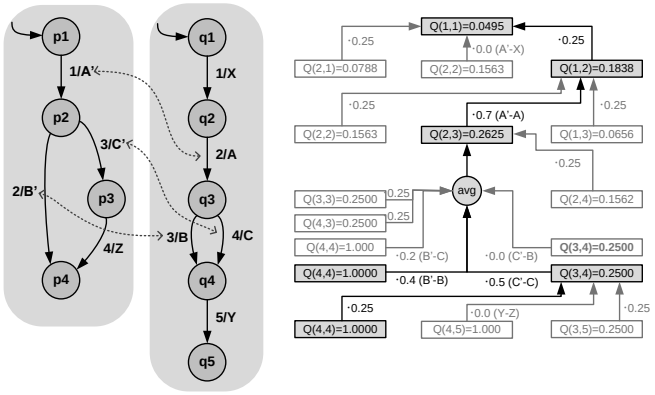
Activity Mapping

The aim of this work is not to measure the similarity of business processes but map the activities. The main idea is to exploit the fact that the Lagrangian multipliers, computed while solving the linear program, can be interpreted as decisions that are made when simulating one process by another. A Lagrangian multiplier corresponds to exactly one inequality constraint of a linear program and is not equal to zero if and only if this constraint is active. By inspection of the Lagrangian multipliers corresponding to constraints defined in Equation 9, we can identify if for a certain combination of states s_i and s_j a transition $s_j \xrightarrow{b} s_m$ has been used at some point to simulate another transition $s_i \xrightarrow{a} s_k$. This can be interpreted as evidence that the labels of these two transitions are related.

The following algorithmconstructs a mapping of activities based on the simulation of LTS_1 by LTS_2. If one LTS skips a step or if two transitions are used together is determined by inspecting the Lagrangian multiplier of the corresponding constraint of the linear programming solution (Equations 6-9).

```
trace(map, Q(i,j))
visitedQs.add(Q(I,j));
  if Q(i,j)=1 then return mapping;
  if visitedQs.contains(Q(i,j)) then return mapping;
  if LTS_1 skips then call trace(map, Q(i,m));
  else
    for each transition t leaving state i
      if LTS_2 skips then call trace(map, Q(k,j));
      else
map.add(a,b);
        call trace(map, Q(k,m));
      end if
    end for
  end if
end.
```

Given two LTSs, now called p and q, one can solve the linear program and call $trace\left(emptyMap, Q\left(s_{in_p}, s_{in_q}\right)\right)$ with s_{in_p} being the initial state of p and s_{in_q} that of q. Using the algorithm, one can see how process q simulates p and which activities are associated. This algorithm is now used to illustrate how the mapping is constructed in two simple examples.



Label similarities

L	X	A	B	C	Y
A'	0.0	**0.7**	0.0	0.2	0.1
B'	0.0	0.1	**0.4**	0.2	0.0
C'	0.1	0.0	0.0	**0.5**	0.1
Z	0.0	0.0	0.1	0.1	0.0

Simulation of p by q **p=0.25**

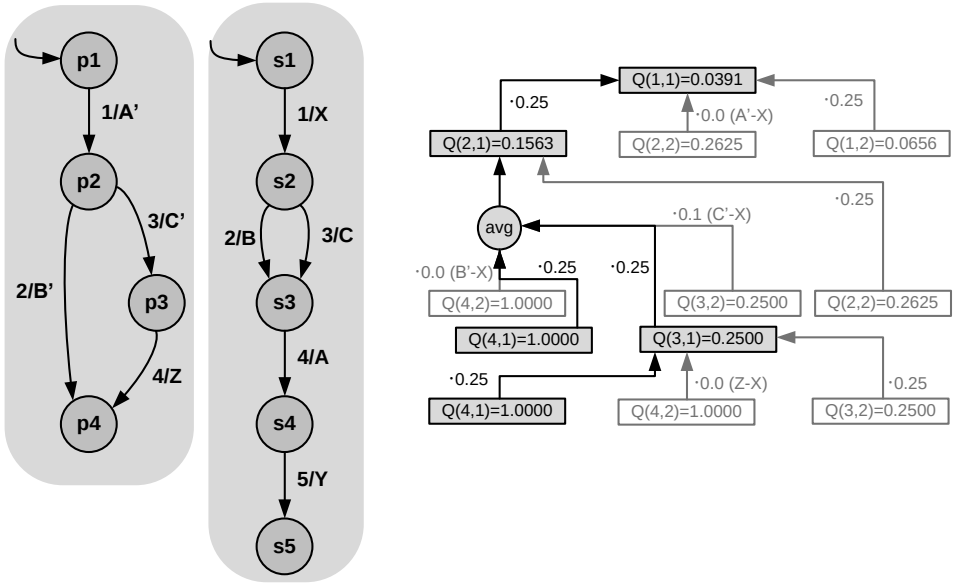
Q	q1	q2	q3	q4	q5
p1	**0.0495**	0.1838	0.0656	0.0391	0.0391
p2	0.1563	0.1563	0.2625	0.1562	0.1562
p3	0.2500	0.2500	0.2500	0.2500	0.2500
p4	1.0000	1.0000	1.0000	1.0000	1.0000

Figure 3: Exemplary illustration of simulating process p by process q.

First, consider the two LTSs p and $q$in Figure 3. Label similarities are depicted in the lower left corner, *Q(i,j)* values delivered by the linear program in the lower right. Note that all similarities except those of A and A', B and B' as well as C and C' are rather low. We now call $trace\left(emptyMap, Q(p1, q1)\right)$. The course of this calculation is represented by the tree in the upper right corner of Figure 3, with nodes representing pairs of states from$p$ and q together with the degree to which

the state of q simulates that of p. Highlighted nodes represent pairs of states for which the algorithm is called. Shaded nodes represent choices that are ruled out.

Initially, we start at the node corresponding to Q(1,1), which is the node assuming a value equal to the degree to which q1 simulates p1. We now have three possible paths to pursue. First, process p might do nothing, in which case process q would move to q2. As the value of Q(1,2), representing the degree to which q2 simulates q1, is equal to 0.1838, which, multiplied by a gap penalty of $p = 0.25$, amounts exactly to 0.0495, we already found the active constraint and can proceed with Q(1,2). If instead p would have moved to p2 and q would have done nothing, we would end up with a value of $0.0788 \cdot 0.25 = 0.0197$, which is smaller than 0.0495. Similarly, if both processes had moved to another state, the result would have been $0.1563 \cdot 0 = 0 < 0.0495$, since labels A' and X are 0% similar. This demonstrates that the constraints corresponding to these decisions (Equations 8 and 9 respectively) are not active and their Lagrangian multipliers are equal to zero.



Label similarities

L	X	A	B	C	Y
A'	0.0	**0.7**	0.0	0.2	0.1
B'	0.0	0.1	**0.4**	0.2	0.0
C'	0.1	0.0	0.0	**0.5**	0.1
Z	0.0	0.0	0.1	0.1	0.0

Simulation of p by q **p=0.25**

Q	q1	q2	q3	q4	q5
p1	**0.0391**	0.0656	0.1094	0.0391	0.0391
p2	0.1563	0.2625	0.1563	0.1562	0.1563
p3	0.2500	0.2500	0.2500	0.2500	0.2500
p4	1.0000	1.0000	1.0000	1.0000	1.0000

Figure 4: Exemplary illustration of simulating process p by process s.

Being at Q(1,2), we have to check the case that p skips a step again, which would lead to Q(1,3). This time however, $0.0656 \cdot 0.25 = 0.0164$ is smaller than 0.1838. Thus, the corresponding constraint is not active. The two options left are to let process q skip while p moves to p2 or to associate transition 1/A' with 2/A. For the first case, we have $0.1563 \cdot 0.25 = 0.039075 < 0.1838$ and the constraint is inactive. This leaves associating 1/A' and 2/A as the only option. In fact, we see that $0.2625 \cdot 0.7 = 0.18375$ equals the value of Q(1,2) (up to a rounding error). As we have associated the transitions, we also associate the labels A and A', indicatedby a dashed line. Proceeding analogously, transitions 2/B' and 3/B as well as 3/C' and 4/C are being associated. We now have two different nodes to check. One corresponds to state p4, which has no outgoing transitions. Thus, the computation ends at this node. For node Q(3,4), we have to let process q skip a last time and in turn arrive at a node corresponding to p4. The algorithm terminates and delivers, as expected, a mapping between equivalent parts of both processes with X, Y and Z left unmapped.

A similar situation is illustrated in Figure 4, with the only difference being the fact that in the second process, now called s, the decision between activities B and C is done prior to the execution of A. While the labels of these transitions still exhibit a high similarity, the control flow structure does not fit anymore. In this case, the algorithm does not map any activities.

EVALUATION

Our goal was improving the quality of activity mappings between BPDs in presence of noisy label similarities. To demonstrate and quantify the improvement, we conducted a simulation study. It involved creating an activity mapping between two processes using two different methods. We compare our approach to a simple greedy matcher, working in the following way. At any time, consider only those activities that have not been matched with any other activity yet. Search for the pair with the highest label similarity and match these two. Proceed until no candidate pairs are left.

The two processes to which these matchers were applied are processes p and q of the example shown in Figure 3. A matcher correctly identified a mapping if and only if it associated activities 1/A' with 2/A, 2B' with 3B, 3C' with 4C and no other activities. To simulate noise, we randomly generated label similarities according to the following scheme. For each pair of labels, we sampled a random number x from a beta distribution with parameters $\alpha = 1$ fixed and $\beta \in \{20,18,16,14,12,10,8,6\}$ varying. If the pair of labels was either A' and A, B' and B or C' and C, the similarity was set to $1 - x$ while in all other cases it was set to x. Like in many traditional approaches (c.f. section 2.2), we considered all label similarities below a certain cutoff value (0.3) as zero. As gap penalty, we used $p = 0.25$. The probability density functions for $\beta = 20$ and $\beta = 6$ are illustrated in Figure 5 (a). One can see that higher values become increasingly probable as β decreases while for all$\beta's$ lower values are more probable than higher values. This way, the similarity of labels that belong to each other is high with occasional drops, while all other label similarities are low with occasional peaks. The lower β becomes, the stronger is this noise.

We generated 1000 label similarity matrices for each $\beta \in \{20,18,16,14,12,10,8,6\}$, applied both matchers, and counted the mistakes they made. The total number of errors is plotted in Figure 5 (b). For low levels of noise (high values of β), both techniques deliver good results. As the noise increases (β decreases) both matchers tend to make mistakes, but the number of mistakes is considerably smaller for the bisimulation approach. For $\beta = 6$, the greedy matcher produced 239 erroneous mappings while the bisimulation approach did only make 136 mistakes, which is a reduction to approx. 57%.

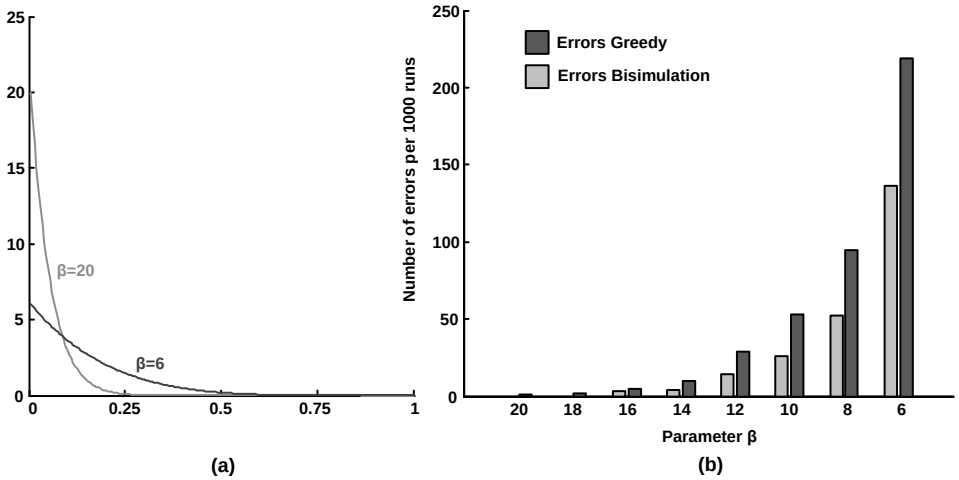


Figure 5: (a) illustrates the probability density function of the beta distribution for different parameters, (b) represents the errors made in the simulation runs

CONCLUSION

In this paper, we proposed an approach to map corresponding activities of two BPDs. It accounts for both local label similarities as well as the global control flow structure simultaneously. This was achieved by defining an optimization problem delivering a similarity of these two BPDs which, as a byproduct, also delivers the mapping. A linear programming formulation of this optimization problem allows the usage of sophisticated solvers that are able to handle large scale problems efficiently.

The motivation for our approach was the fact that label similarities often exhibit high levels of noise as the short phrases used to label activities in BPDs can produce ambiguities that are hard to resolve. Relying only on label similarities for creating a mapping between activities means that viable information codified in the control flow structure of the processes is wasted. Thus, we incorporated this information into our approach, with the goal of making activity mapping creation more robust to noise. A preliminary evaluation demonstrated that it is in fact superior to a simple, greedy procedure.

Future research will focus on a more comprehensive evaluation of the approach. It will be applied to a large collection of real world BPDs to evaluate its applicability in practice. An empirical investigation of label similarities will be conducted to gain further insights into their actual quality. This will allow a more realistic modeling of noise thus increasing the external validity of future simulation studies.

REFERENCES

1. Cortadella, J.; Kishinevsky, M.; Lavagno, L.; Yakovlev, A.: Deriving Petri Nets from Finite Transition Systems. In: IEEE Transactions on Computers, vol. 47, no. 8, 1998, pp. 859-882.
2. Dijkman, R.; Dumas, M.; García-Bañuelos, L.: Graph Matching Algorithms for Business Process Model Similarity Search. In: Proc. of the 7th Int. Conf. on Business Process Management 2009, 2009, pp. 48-63.
3. van Dongen, B.; Dijkman, R.; Mendling, J.: Measuring Similarity between Business Process Models. In: Proc. of the 20th Int. Conference on Advanced Information Systems Engineering, 2008, pp. 450–464.
4. Dumas, M.; Garcia-Bañuelos, L.; Dijkman, R: Similarity search of business process models. In: Bulletin of the IEEE Computer Society Technical Committee on Data Engineering, vol. 32, no. 3, 2009, pp. 23-28.
5. Delfmann, P.; Herwig, S.; Lis, L.: Unified enterprise knowledge representation with conceptual models - Capturing corporate language in naming conventions. In: Proc. of the 30th Int. Conf. on Information Systems, 2009.
6. Dijkman, R..; Dumas, M.; van Dongen, B.; Käärik, R.; Mendling, J.: Similarity of business process models: Metrics and evaluation. In: Information Systems, vol. 36, no. 2, 2010, pp. 498-516.
7. Ehrig, M.; Koschmider, A.; Oberweis, A.: Measuring similarity between semantic business process models. In: Proc. of the 4th Asia-Pacific Conference on Conceptual Modelling (APCCM '07), 2007, pp. 71-80.
8. Fellbaum, C.: WordNet: An Electronic Lexical Database. MIT Press, Massachusetts, 1998.
9. Höfferer, P.: Achieving business process model interoperability using metamodels and ontologies. In: Proc. of the 15th European Conference on Information Systems (ECIS 2007), vol. 7, no. 09.06, 2007, pp. 1620-1631.
10. Kunze, M.; Weidlich, M.; Weske, M.: Behavioral Similarity - A Proper Metric. In: Proc. of the 9th Int. Conf. on Business Process Management, 2011, pp. 166-181.
11. Li, C.; Reichert, M.; Wombacher, A.: On measuring process model similarity based on high-level change operations. In: Proc. of the 27th Int. Conf. on Conceptual Modeling (ER'08), 2008, pp. 248-264.
12. Lohmann, N.; Verbeek, E.; Dijkman, R.: Petri Net Transformations for Business Processes – A Survey. In: Lecture Notes in Computer Science, vol. 5460/2009, pp. 46-63.
13. de Medeiros, A. K.; van der Aalst, W. M. P.: Quantifying process equivalence based on observed behaviour. In: Data & Knowledge Engineering, vol. 64, no. 1, 2008, pp. 55-74.
14. Mendling, J.; Simon, C.: Business Process Design by View Integration. In: Lecture Notes in Computer Science, vol. 4103, 2006, pp. 55-64.
15. Minor, M.; Tartakovski, A.; Bergmann, R.: Representation and structure-based similarity assessment for agile workflows. In: Proc. of the 7th Int. Conf. on Case-Based Reasoning, ICCBR, 2007, pp. 224-238.
16. Madhusudan, T.; Zhao, J. L.; Marshall, B.: A case-based reasoning framework for workflow model management. In: Data & Knowledge Engineering, vol. 50, no. 1, 2004, pp. 87–115.
17. Navarro, G.: A Guided Tour to Approximate String Matching. In: ACM Computing Surveys, vol. 33, no. 1, 2001, pp. 31–88.
18. Sokolsky, O.; Kannan, S.; Lee, I.: Simulation-based graph similarity. In: Tools and Algorithms for the Construction and Analysis of Systems, 2006, pp. 426-440.
19. Uba, R.; Dumas, M.; Garcia-Bañuelos, L.; La Rosa, M.: Clone Detection in Repositories of Business Process Models. In: Proc. of the 9th Int. Conf. on Business Process Management, 2011.
20. Weidlich, M.; Dijkman, R.; Mendling, J.: The ICoP Framework: Identification of Correspondences between Process Models. In: Proc. of the 22th Int. Conf. on Advanced Information Systems Engineering CAiSE, 2010, pp. 483–498.
21. Zha, H.; Wang, J.; Wen, L.; Wang, C.; Sun, J.: A workflow net similarity measure based on transition adjacency relations. In: Computers in Industry, vol. 61, no. 5, 2010, pp. 463-471.