On Behavioral Process Model Similarity Matching: A Centroid-based Approach

Michaela Baumann✉, Michael Heinrich Baumann*, and Stefan Jablonski

Universität Bayreuth,
Universitätsstraße 30, 95447 Bayreuth
{michaela.baumann,michael.baumann,stefan.jablonski}@uni-bayreuth.de
http://www.ai4.uni-bayreuth.de

Abstract. As business process models have a broad scope of applications, e.g., in science or in business administration the problem of handling large amounts of process models arises. One helpful tool for dealing with this amount of models is to reduce it by using similarity measures in order to detect similar models that can be merged. A set of similar models may be replaced by one model. As a pure similarity of labels is often not enough to compare process models other process perspectives are involved for calculating similarities. The current paper works on the process model's behavior which is one such perspective. A problem that arises when comparing two models and that is covered in this paper is that one of a differing granularity of process steps. Due to this granularity problem M-to-N-mappings are considered.
The present paper provides a centroid-based and so easily computable method for calculating behavioral similarity values which is constructed for M-to-N-mappings and an evaluation of it.

Keywords: Business process model, Behavioral process model similarity, M:N-Matching

1 Introduction

Not only for documentation purposes, business process models have been established in a large amount of organizations. They also serve as supportal means for communication, for training employees and redesigning actual workflows [1]. These widely spread applications lead to vast process model repositories in enterprises that have to be managed somehow [2]. One of these management purposes is to find similar models in order to reduce the tremendous amount of repository elements by detecting and merging similar models. The authors of [3] worked out a total of nine categories for application fields of similarity measures, amongst them process merging, facilitating reuse of models [4] and service

* The work of Michael Heinrich Baumann is supported by a scholarship of "Hanns-Seidel-Stiftung (HSS)" which is funded by "Bundesministerium für Bildung und Forschung (BMBF)".

discovery. But usually, the models are developed by different persons and thus have different levels of granularity which means that process steps in different models are modeled with a different fineness. Furthermore, the terminology, i.e., the way of defining names, labels, etc. varies from model to model and hence a comparison of these models is challenging [5]. These two issues often lead to the fact that actually very similar or even equal models are not recognized as such. Because of this and due to the wide variety of modeling languages and notations, like Event-driven Process Chains (EPCs), Petri Nets, UML (Unified Modeling Language) Activity Diagrams, Workflow Nets, the Business Process Model and Notation (BPMN) or the Business Process Execution Language (BPEL), perfect matches, i.e., a true/false answer to the question if two models are the same, cannot be expected. Instead, a degree of similarity, a value between 0 and 1 where 0 means completely different and 1 is an indication for (virtually) identical models, depending on the definition of the respective similarity measure, is desired. These measures can be defined on different disjoint aspects of process models: on node information, on process structure and on execution semantics [2]. Node information is attached to each process model element, especially activities, and can again be split up into the description of process model elements, assigned roles or agents, ingoing and outcoming data objects, and operational means. Process structure refers graph structure when taking a process model as a graph and execution semantics refers to the question, how, i.e., in which order and under which circumstances (parallel, inclusive, exclusive, loop, etc.) process model elements may be executed. A behavioral similarity usually relies on the execution semantics of a process model. In order to take into account all of this information about process models and to allow for a wide range of modeling notations, we define a process model according to the following, general form:

Definition 1 (Process Model). *A process model is a tuple* $G = (N, E, \lambda, \delta)$ *where*

- N *is a finite, non-empty set of model nodes with*

$$N = A \cup \{start, end\} \cup S_{AND} \cup S_{XOR} \cup S_{OR},$$

  where $S_{AND} = \bigcup_{i=1}^{k_A} \{AND_i, \overline{AND_i}\}$, $S_{XOR} = \bigcup_{i=1}^{k_X} \{XOR_i, \overline{XOR_i}\}$, $S_{OR} = \bigcup_{i=1}^{k_O} \{OR_i, \overline{OR_i}\}$, $A = \{1, \ldots, n_A\}$ *is the set of activity nodes;* S_{AND} *is the set of all parallel gateways where each split gateway* AND_i *has a corresponding merge gateway* $\overline{AND_i}$. S_{XOR} *is the set of all exclusive gateways and* S_{OR} *the set of all inclusive ones (pairwise split and merge).*
- $E \subseteq N^2$ *ist a set of directed edges between model nodes*
- $\lambda : A \to \mathfrak{B} \times \mathcal{P}(\mathfrak{D}) \times \mathcal{P}(\mathfrak{A}) \times \mathcal{P}(\mathfrak{W})$ *is a function that assigns descriptions (*$\mathfrak{B}$ *is a set of strings), data sets (*$\mathcal{P}(\mathfrak{D})$, $\mathfrak{D} = \{d_1, \ldots, d_{n_\mathfrak{D}}\}$*), agent sets (*$\mathcal{P}(\mathfrak{A})$, $\mathfrak{A} = \{a_1, \ldots, a_{n_\mathfrak{A}}\}$*) and sets of non-human resources (*$\mathcal{P}(\mathfrak{W})$, $\mathfrak{W} = \{w_1, \ldots, w_{n_\mathfrak{W}}\}$*) to the respective activities*
- $\delta : \{e \in E \mid e = (a, a) \vee \exists l : (e)_1 = OR_l \vee (e)_1 = XOR_l\} \to \mathfrak{B}$ *assigns certain edges (that ones coming out from inclusive or exclusive gateways) a constraint description.*

This definition is flexible in a way that certain parts of it may be omitted if they are not available or not needed. E.g., if information about non-human resources is not available, λ is defined without $\mathcal{P}(\mathfrak{W})$ in the cartesian product, or if there are no inclusive gateways, N is defined without set S_{OR}.
The focus of the work at hand lies on the behavioral aspect of process models, i.e., on control flow and how two models can be compared with respect to this aspect. During the matching process – this is what we call the process of finding a similarity value between two models – the activity nodes of two process models are not compared one-to-one (activity node compared to acitivity node) but they will be grouped into sets to encounter the problem of differing granularity, e.g., when one activity in the first process model is split up into three process steps in the second model. For these sets of activities centroids, i.e., average positions (see Definition 6), average repeatability and average optionality are calculated to determine behavioral similarity. As far as the authors know, this distinction of behavior into position, repeatability and optionality has not yet been done explicitely in previous work. In [6], process model elements are classified into, among other things, alternative and loop fragments, that resemble optional and repeatable elements. Furthermore, these centroids will be able to punish sets of activities that are widely spread over the whole process model or that have strongly differing manner. The resulting behavioral similarity value can then be combined with other similarity values, e.g., description similarity or data similarity, to get a better matching score that is more independent of local errors, i.e., that is more robust against errors in certain process model aspects [7]. To put it together, the method presented in this paper provides two main results: A normalized similarity value for two process models based on their behavior and a mapping that indicates the resembling parts of them which will be called M-to-N-mapping and is defined as follows (cf. [8]):

Definition 2 (M-to-N-mapping). *Let $G_i = (N_i, E_i, \lambda_i, \delta_i)$, $i = 1, 2$ be two process models with $A_i \subset N_i$ the set of activities of each process model and $P_i \subset \mathcal{P}(A_i) \ni \emptyset$ a complete and disjoint partition of A_i (i.e., $\bigcup_{p \in P_i} p = A_i$ & $\forall p, p' \in P_i : p \cap p' = \emptyset, p \neq p'$), $i = 1, 2$. A mapping between G_1 and G_2 is defined as a bijective function $M : P_1 \to P_2$. In particular, $\emptyset \mapsto p_2$ and $p_1 \mapsto \emptyset$ means, that p_2 and p_1 are deleted, respectively, where $p_1 \in P_1$, $p_2 \in P_2$, and $\neg(\emptyset \mapsto \emptyset)$.*

As Definition 2 shows, sets of activities are mapped rather than single activities which induces the term M-to-N-mapping. These sets of activities are achieved by establishing a partition of set A, i.e., all activities. In specific cases, e.g., when process models that should be compared strongly differ in granularity which is not an unusual task, this method, especially because of considering the M-to-N-mapping, may provide better results than methods presented in previous work. Furthermore, no complex calculations are needed.
The remainder of this paper is organized as follows: The next section gives a rough overview of existing similarity measures and process model matching methods. In Section 3 some conditions for process models are listed to secure soundness of them. Afterwards, the behavioral similarity measure in its three dimensions is introduced step by step. An extension for penalized similarity mea-

sures is given, too. Thereafter, in Section 5, a short example and an evaluation is performed. Section 6 revises the paper and gives ideas for future work.

2 Background and Related Work

In the literature, many techniques and methods for calculating the similarity or, on contrast, the distance of process models are presented. The authors of [3] provide a comparing overview of some of these techniques. Another collection of several matching techniques can be found in [9]. One way of measuring the similarity between a pair of process models is to first define a mapping between these two models. This mapping can either assign one element of the first model to one element of the second model, which often leads to a partial injective function [10], or map a set of elements of one model to a set of elements of the second model [7]. Thereby, we will refer to the latter defined mapping as M-to-N-mapping which is defined according to Definition 2 and is just a generalization of the former 1-to-1-mapping definition, i.e., an extension to powersets. As the authors of [11] suggest, for many scenarios, e.g., when processes have been developed independent of each other, a M-to-N-mapping is preferred to a simple 1-to-1-mapping. M-to-N-mappings are capable to overcome problems of granularity levels which is one of the future tasks stated in [12]. In [5] a method for establishing N-to-1-mappings is presented, but not extended to a M-to-N-mapping due to the applied matching techniques.

Label-based and structural similarity values: After having established a mapping between the elements of process models, similarity values between these elements can be computed. Depending on the given models, various information is used for this computation step. A similarity value based on the activity labels of a process model is, e.g., presented in [10] [13] and [14]. It makes use of the so-called string-edit-distance and other string-modifying techniques like stemming [10] or replacing certain words through synonyms [15]. This similarity is often referred to as syntactic or, when applying synonym dictionaries, semantic/linguistic similarity. Another information that can be used for comparing process models is information about authorized agents and assigned input and output data of each activity, which is available for example in BPMN process models (cf. [16]). This additional information can be analyzed lexically [5], like the activity descriptions, or when applying M-to-N-mappings through set-based methods performed on the subjects' or objects' identifiers, as it is done in [7] and [8]. Another important aspect of process models that is not considered in the techniques mentioned so far is the arrangement of the process elements. Basically, this arrangement can be categorized into two different similarity metrics: structural/contextual similarity and behavioral similarity [11], [17]. The contextual similarity, as it is defined in [11] or [15], is applicable especially for EPCs and Petri Nets due to their predetermined structure of events and functions (EPCs) and places and transitions (Petri Nets) [11]. Here, similarity is computed with respect to predecessor and successor model elements. Structural similarity makes use of the topology of process models, e.g., the concept of graph-edit-distance,

where the process model is considered to be a labeled graph consisting of nodes and edges. Similarity is then computed by deriving the minimum number of graph edit operations that are necessary to transform the first graph into the second one [10], [11]. By constrast, behavioral matching techniques are based on the execution semantics of a process model [10], which means, that, e.g., parallelism or exclusiveness of model elements as well as their possible execution order is respected.

Various Definitions of Behavioral Similarity: There are a lot of different approaches in the literature when it comes to behavioral similarity of process models. In [7] a computing method for M-to-N-mappings is suggested that makes use of partial order relations of the activity elements but is limited to serialized process models without any gateways. Behavioral profiles, a set of valid relations between every two process model elements, are introduced in [17] and [18] to define different behavioral similarity values, based on concrete relations, that may be strict, exclusive or interleaving. This approach is, however, applicable if the process models are mapped 1-to-1. Another common method is to look at the traces, or state transition system in the case of petri nets, of the process models to be compared [2]. Even if there is only a finite set of traces which is the case if there are no loops within the process models the problem of computing the trace-based behavior of a model is NP-hard [11]. An explicit discussion of trace-based methods is presented in [3]. Regarding partial traces is a variant of this trace-based approach and discussed in [19]. To overcome the computational complexity of traces, an approximation via casual footprints can be performed [2]. Casual footprints define the so-called look-back and look-ahead links of process model elements [11], [15]. However, this similarity value takes sequential, parallel or exclusive behavior of the model elements, which is important information about a process model's behavior, only unsufficiently into account (cf. [11]). A further approach of determining similarity between two process models is given in [20] where process models are compared with respect to some typical behavior that is gained from process event logs. However, as we want to compare two process models and check for their similarity, a third component serving as reference like an event log is not needed in the method we present in the further course of this paper. The aim of the paper is to derive a behavioral similarity function whose calculation is not too difficult which would be the case for, e.g., casual footprints [3] and that is suitable for M-to-N-mappings. All in all, an approach like ours makes use of already existing concepts like embeddability into graph-edit similarity but is adjusted to situations where previous approaches are not able to detect similarity or to take into account all given information.

3 Conditions for Process Models

For our work, we rely on process models given according to Definition 1 and that are structured in a block-like way, that means that gateway blocks may not overlap [21], [22]. A gateway block is a part of a process model that begins with a split gateway that has one incoming edge and ends with a join gateway

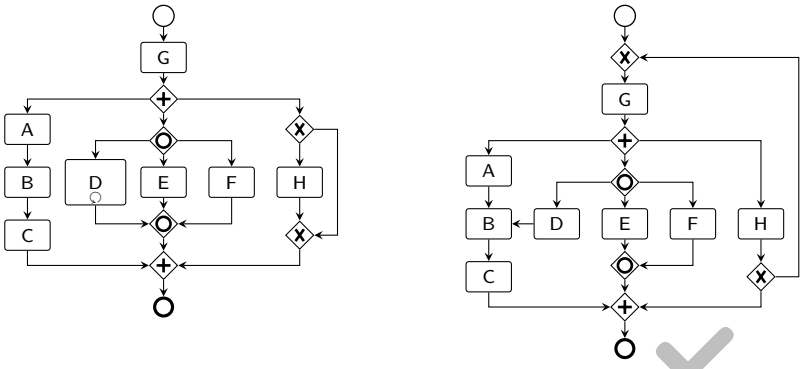


Fig. 1. Left: block-structured process model in BPMN 2.0 notation. Right: non-block-structured process model

of the same kind with one outgoing edge. That is, we always have pairs of nodes $(AND_i, \overline{AND_i})$, $(XOR_j, \overline{XOR_j})$ or $(OR_k, \overline{OR_k})$ so that deadlocks or multiple termination is prevented [23]. Loops are allowed but only with XOR-gateways with two decision edges. In this case, the XOR-block starts with the join gateway with two incoming edges and one outgoing edge and ends with the split gateway with one incoming edge and two outgoing edges, so we have $(\overline{XOR_j}, XOR_j)$. For transforming graph-oriented process models into block-structured ones see [21]. The left side of Fig. 1 shows a process model that is well-structured in a block-like way, the right one is not well-structured. To define the conditions gateway nodes must fulfill in order to get a structured process model the terms *chain* and *subchain* are introduced:

Definition 3 (Chain, Subchain). *A chain of length* $l \in \mathbb{N} \cup \infty$ *from* n *to* n', $n, n' \in N$, *is* $K \in E^l$ *with*

$$(K_1)_1 = n \wedge (K_l)_2 = n' \wedge \forall i \in \{1, \ldots, l-1\} : (K_i)_2 = (K_{i+1})_1.$$

A subchain K' *of lenth* l' *from* m *to* m', $m, m' \in N$, *of chain* $K = (K_1, \ldots, K_l)$ *is* $K' = (K_s, K_{s+1}, \ldots, K_t)$ *with* $1 \leq s \leq t \leq l$, $l' = t - s + 1$, *and* $(K_s)_1 = m$, $(K_t)_2 = m'$.

According to this definition a chain from model element n to model element n' exists if there is a sequence flow passing arbitrary other model elements between n and n'. Chains are defined using edges, i.e., via the sequence flow arrows. Subchains are chains themselves.

With this, we now state a few, plausible rules proper process models should follow before a behavioral matching score can be computed. Following these rules provides block-structured process models whereby this assumption is no real limitation of generality as every sound process model can be transformed to a block-structured process model [21]. Most of the rules listed in the following can be found in [21] as structural properties, except that we have formulated them with the help of edges and chains (see Definition 3). First, some conditions for start and end events are stated. They say that there may exist only one start event or end event, respectively, with exactly one outgoing or incoming edge.

(e1) $\exists! e \in E : (e)_1 = start \wedge \nexists e \in E : (e)_2 = start$
(e2) $\exists! e \in E : (e)_2 = end \wedge \nexists e \in E : (e)_1 = end$
(e3) $\forall n \in N \setminus \{start\} : \exists$ chain K from $start$ to n
(e4) $\forall n \in N \setminus \{end\} : \exists$ chain K from n to end

Activities must fulfill some conditions, too, namely those that they have exactly one incoming and one outgoing edge. Splits and joins are always done at the respective gateway types.

(a1) $\forall a \in A : \exists! e \in E \setminus \{(a,a)\} : (e)_2 = a$ (incoming edge)
(a2) $\forall a \in A : \exists! e \in E \setminus \{(a,a)\} : (e)_1 = a$ (outgoing edge)

Edge (a, a) is skipped because it is possible to have loop tasks, e.g. in BPMN (see [16]). These tasks can be repeated and thus, if this repeatability is indicated with a sequence flow directing from the activity to itself, this activity has two incoming and outgoing edges. Another possibility would be to introduce a XOR-loop that reflects the same functionality. Next, we state the rules for gateways so that a process model following these rules is block-structured. This is according to the definition for reducible flow graphs stated in [24]. Structered (process) graphs are reducible flow graphs as shown in [21]. Basically, if there are no loops, the following statements can be applied for AND-, OR- and XOR-gateways, so, variables GW_k and $\overline{GW_k}$ can be replaced by AND_k and $\overline{AND_k}$, OR_k and $\overline{OR_k}$ or XOR_k and $\overline{XOR_k}$. Conditions for loops are listed afterwards.

(g1) $\forall GW_k : \exists! e \in E : (e)_2 = GW_k$
(g2) $\forall \overline{GW_k} : \exists! e \in E = (e)_1 = \overline{GW_k}$
(g3) $|\{e \in E : (e)_1 = GW_k\}| = |\{e \in E : (e)_2 = \overline{GW_k}\}| \geq 2$

(g1) and (g2) state that split gateways have exactly one incoming edge and join gateways exactly one outgoing edge. Additionally, (g3) says that split gateways have at least two outgoing edges and join gateways the same number of incoming edges. Block-structure is provided by the following statements:

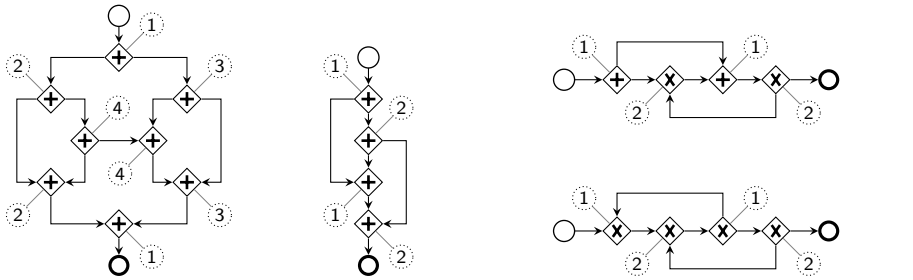
(g4) $\forall GW_k, \forall$chains K from GW_k to $end : \exists$ subchain of K from $\overline{GW_k}$ to end
(g5) $\forall \overline{GW_k}, \forall$chains K from $start$ to $\overline{GW_k} : \exists$ subchain of K from $start$ to GW_k

(g4) and (g5) ensure that every split is joined and vice versa and that no paths get lost after the split gateway. Situations as shown in Fig. 2(a) are prevented. The parallel gateways can be substituted by arbitrary other gateway types as long as every split gateway has a suitable join gateway. Activity nodes are skipped for better readability but can be inserted on every sequence flow. Further examples for erroneous patterns are shown in [23].

When permitting loops in process models, some modified and additional conditions must hold. Note, loops are only allowed for XOR-gateways with exactly one backward jump and one edge forward [21]. OR-gateways may not induce loops [22]. The backward jump is the direct chain from the XOR-split to the corresponding XOR-join and characterized in condition (l0):

(l0) $\forall XOR_k : \exists$chain from XOR_k to $\overline{XOR_k}$

The following conditions (l4) and (l5) apply for *XOR-loops* instead of (g4) and (g5). Split and join gateways simply change roles. (g1), (g2) and (g3) also hold for loops, so (g1)=(l1), (g2)=(l2) and (g3)=(l3) with "= 2" instead of "≥ 2".



(a) Situations prevented if (g4) and (g5) are fulfilled.

(b) Situations prevented if (g7) and (l5) are fulfilled.

Fig. 2. Non-block-structured process models; pins mark related split and join gateways.

(l4) $\forall XOR_k, \forall$chains K from *start* to $XOR_k : \exists$subchain of K from *start* to $\overline{XOR_k}$

(l5) $\forall \overline{XOR_k}, \forall$chains K from $\overline{XOR_k}$ to *end* : $\exists$subchain of K from XOR_i to *end*

Additionally, to prevent backward jumps from leaving other blocks they are embedded in, for all gateway pairs $(GW_k, \overline{GW_k})$, including XOR-loops, it must hold that

(g6) $\forall GW_k, \forall$chains K from GW_k to $\overline{GW_k} : \exists$subchain of K from GW_i to $\overline{GW_k}$

(g7) $\forall \overline{GW_k}, \forall$chains K from $\overline{GW_k}$ to $\overline{GW_k}$: $\exists$subchain of K from $\overline{GW_k}$ to GW_k

and thus (l6)=(g6) and (l7)=(g7). This means, that a XOR-gateway either fulfills (g1)–(g7) or it meets the loop requirements (l0)–(l7). An OR- or an AND-gateway has to fulfill (g1)–(g7). The situations as shown in Fig. 2(b) do not fulfill conditions (g7) or (l6). Note, that some of the conditions stated in the current section may be redundant. It would be another task to check and to prove which conditions are actually needed, but in the way they are stated above they are easy to check and provide correct, i.e., block-structured process models.

4 The Centroid-based Behavioral Similarity Measure

For process models satisfying all of the conditions listed in Section 3 that imply no heavy restrictions but rather provide sound, executable process models, a similarity value to compare process models on their behavior can be defined. This similarity value assumes a M-to-N-mapping between the two models and makes use of the centroids of the resulting sets of nodes. In particular, a M-to-N-mapping M is given according to Definition 2, i.e., a partition P_1 of activities A_1 of the first process model G_1 is mapped bijectively to a partition P_2 of activities A_2 of the second process model G_2 and every element of a partition $p \in P_i$ is a set of activities of the underlying process model, i.e., $p \subseteq A_i$. We want to remark that one task of the ongoing matching process is to find the best mapping, that means the mapping providing the highest similarity value that demonstrates the correspondences between the two process models G_1 and G_2 best, cf. [7] and [10]. The targeted behavioral similarity measure considers the order of nodes

given by the sequence flow but also takes into account mandatory and optional activities as well as repeatable ones which are the three dimensions of behavior as already mentioned in the introduction. A penalty score is added to neglect sets of heterogenous activities, e.g., widely spread sets of activities.

4.1 Positional Similarity

The first behavioral dimension reflects the location of nodes in a process model. The (positional) centroid of a set of nodes is computed with respect to the nodes' positions in the process model. This position is given through the length of the shortest chain from the *start* event to the node divided by the length of the shortest chain going from *start* to *end* while passing the node. The next two definitions formulate this precisely.

Definition 4 (Minimal length of a chain). *The minimal length $m(n, n')$ of a chain from $n \in N$ to $n' \in N$ (if such a chain exists) is given as:*

$$\min_l : \exists \textit{chain of length } l \textit{ from } n \textit{ to } n'$$

with $m(n, n) = 0 \ \forall n \in N$. If there does not exist a chain from n to $n' \neq n$ we have $m(n, n') =$ `NA`.

Definition 5 (Node position). *The position $\pi(n)$ of node $n \in N$ is:*

$$\pi(n) = \frac{m(start, n)}{m(start, n) + m(n, end)}.$$

This definition of a node's position denotes some kind of relative position in a process model. Particularly, we have $\pi(start) = 0$ and $\pi(end) = 1$. By using the minimal length in the definition of node position instead of some average chain length the problem of infinitely long chains resulting from loops is avoided. For all $p \in P$, P being a partition of process model G that means being a partition of the set of activities A of G, centroids $\pi(p)$ are computed as the average position of all node elements of p:

Definition 6 (Centroid of a node set). *The centroid $\pi(p)$ of $p \in P$ is given through $\pi(\emptyset) =$ `NULL` and*

$$\pi(p) = \frac{1}{|p|} \sum_{n \in p} \pi(n), \ p \neq \emptyset. \tag{1}$$

As $\pi(n) \in [0, 1] \ \forall n \in p$, $\pi(p)$ is also between 0 and 1. All `NULL`-values occurring in this paper are ignored, i.e., if $p = \emptyset$ or $M(p) = \emptyset$ all values for p and $M(p)$ are ignored in the behavioral context but they lead to a high overall distance between the compared models (cf. graph-edit-distance). The behavioral similarity of two partitions P_1 and P_2 of two process models combines the differences of the centroids of the mapped sets of nodes for all elements $(p, M(p))$ of the mapping M, $p \in P_1$, where $M : P_1 \to P_2$ bijective; $p \mapsto M(p)$.

Definition 7 (Behavioral similarity 1). *For two partitions P_1, P_2 of process models G_1, G_2 induced by a mapping M the first dimension of behavioral similarity, the position-based similarity, is given through*

$$VSim_M^{\pi}(P_1, P_2) = \frac{1}{|P_1|} \sum_{p \in P_1} \left(1 - |\pi(p) - \pi(M(p))|\right). \tag{2}$$

Formula (2) depending on the partitions P_1 and P_2 of two process models G_1 and G_2 induced by a mapping M can also be formulated with the process models themselves, so $VSim_M^{\pi}(G_1, G_2) = VSim_M^{\pi}(P_1, P_2)$.

4.2 Repeatability and Optionality

Besides the position value π we can also assign a repeatability value ϱ and an optionality value o (*omikron*) to each node. These other two dimensions of the behavior of process models display the execution with regard to the different gateway types. The approach for these two is similar to that of Section 4.1.

Definition 8 (Node repeatability). *The repeatability $\varrho(n)$ of node $n \in N$ is:*

$$\varrho(n) = 1,\ \exists \textit{chain from } n \textit{ to } n;\ 0,\ \nexists \textit{chain from } n \textit{ to } n.$$

The repeatability value provides information if a node can be executed more than once in one process instance, i.e., if it is involved in a XOR-loop. In BPMN, it is possible to mark activities as loop tasks which are also treated as repeatable nodes as mentioned after conditions (a1) and (a2) in Section 3. Another property of nodes is their optionality, i.e., if a node has to be executed in one process instance or if the process can finish without having executed this particular node. Optionality can be given if XOR- or OR-gateways appear.

Definition 9 (Node optionality). *Let $OGW_k \in \{XOR_k, OR_k\}$ including XOR-loops. The optionality $o(n)$ of node $n \in N$ is:*

$$o(n) = \begin{cases} 1 & \exists OGW_k :\ \exists \textit{chain } K \textit{ from } OGW_k \textit{ to } \overline{OGW_k} : \\ & \exists e \in K : (e)_2 = n \wedge \nexists i \neq j : e_i = e_j\ (e_i, e_j \in K), \\ 0 & \textit{otherwise} \end{cases}$$

The condition in the definition of $o(n)$ simply checks if a node lies on a loopless chain from a split (X)OR-gateway to the corresponding join gateway. As we do not assume any process log information about executed instances as e.g. shown in [20], there is no statement if an optional node is more or less likely to be executed. For future work, one can think of also assigning optionality values $\in (0, 1)$, e.g., by using execution probabilities or relative frequencies obtained from process execution logs. Analog to Definition 6 repeatability and optionality is extended to sets of node as shown in the following definition.

Definition 10 (Repeatability and optionality of node sets). *For $p \in P$, P a partition of G (i.e., of A) repeatability $\varrho(p)$ and optionality $o(p)$ of a node set p is given through 1 by replacing π through ϱ or o, respectively.*

With this preparatory work behavioral similarity for the two remaining behavior dimensions can be formulated:

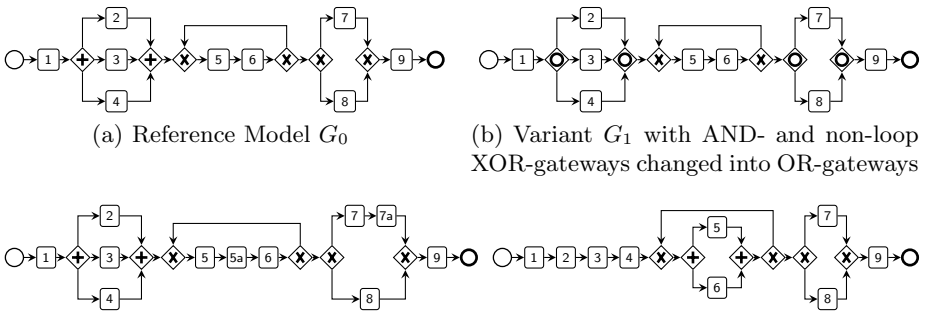
Definition 11 (Behavioral similarity 2 and 3). *For two partitions P_1, P_2 of G_1, G_2 induced by a mapping M the behavior similarities based on repeatability and optionality is given through equation 2 by replacing π through ϱ or o, respectively.*

4.3 Penalty Functions

The positional centroids of the activity set consisting of "the first" and "the last" activity and of a activity set p with $p = \{a\}$, $\pi(p) = \pi(a) = 0.5$ would be the same when calculated according to formula (2). But it is quite obvious that these two sets of nodes are unlikely to match together. This is why we introduce penalty terms for every dimension of behavioral similarity that lower the similarity value if one or both partition elements p and $M(p)$ are heterogenous activity sets. These penalty functions depend on the underlying mapping M and are denoted with $pen^{\pi}_M, pen^{\varrho}_M, pen^{o}_M \geq 0$. They have to be computed for each partition separately. The resulting penalized similarity is of the form $penVSim^{\xi}_M(P_1, P_2) = \left(VSim^{\xi}_M(P_1, P_2) - pen^{\xi}_M(P_1) - pen^{\xi}_M(P_2)\right)^+$, where $\xi \in \{\pi, \varrho, o\}$ and P_1 and P_2 are the partitions induced by M on the two process models G_1 and G_2. We set $penVSim^{\xi}_M(G_1, G_2) := penVSim^{\xi}_M(P_1, P_2)$. As $VSim \in [0, 1]$ it is reasonable to demand for penalty functions $pen \in [0, 0.5]$. A function that meets this requirement and that somehow measures the spread of a set of objects is the variance, in this case the sample variance that uses the centroids as (sample) means. Therefore, if we apply the unbiased sample variance, we get $pen^{\xi}_M(p) = \frac{1}{|p|-1} \sum_{a \in p} (\xi(a) - \xi(p))^2$ with $\xi \in \{\pi, \varrho, o\}$ as penalty value for one partition element $p \in P$ with $|p| \geq 2$. For $|p| = 1$ the penalty value is 0 and for $p = \emptyset$ it is not available, i.e., set to `NULL`. The penalty value for a whole partition P is computed as the average over the single penalty values: $pen^{\xi}_M(P) = \frac{1}{|P|} \sum_{p \in P} pen^{\xi}_M(p)$.

4.4 (Penalized) Behavioral Similarity

To get one value for behavioral similarity one has to combine the three dimensions of behavior and their corresponding similarity values $VSim^{\pi}$, $VSim^{\varrho}$ and $VSim^{o}$ or, analog, the penalized similarity values $penVSim^{\pi}$, $penVSim^{\varrho}$ and $penVSim^{o}$. This combination can take place with help of a weighted sum of the three values where the weights can be chosen according to one's own impression of suitability or, which would be worth futher studies, according to statistical findings including model training and parameter estimation, e.g.,



(a) Reference Model G_0

(b) Variant G_1 with AND- and non-loop XOR-gateways changed into OR-gateways

(c) Variant G_2 with refined granularity of activities

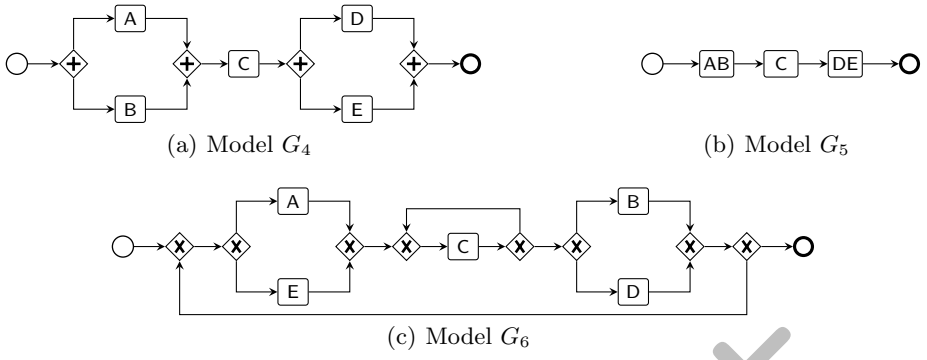
(d) Variant G_3 with change of parallelly and sequentially executable activities

Fig. 3. Initial model G_0 and variants G_1, G_2 and G_3

maximum likelihood methods. With non-negative weights w^π, w^ϱ and w^o with $w^\pi + w^\varrho + w^o = 1$ the weighted sum, i.e., the behavioral similarity value for two process models G_1 and G_2 under mapping M, would be of the following form: $VSim_M(G_1, G_2) := \sum_{\xi \in \{\pi, \varrho, o\}} \omega^\xi VSim^\xi_M(G_1, G_2)$.
For the penalized behavioral similarity $penVSim_M(G_1, G_2)$ the similarity values for the three behavioral dimensions are replaced by the penalized similarity values of the three dimensions. Both $VSim$ and $penVSim$ always take values between 0 and 1 where 0 means no similarity and 1 full similarity. The (penalized) behavioral similarity can then again be used for calculating the similarity value including all process model perspectives, i.e., activity description, data objects, human and non-human resources and the behavior [7], [8].

5 Example and Evaluation

This section gives computation examples for $VSim$ and $penVSim$ and a comparison of casual footprints, "smallest" casual footprints, and the centroid-based approach presented in the work at hand, cf. Fig. 3, Fig. 4, and [3].
Example: At first, a short example is given in this section with one process model G_0 as shown in Fig. 3(a) and three variants (Fig. 3(b)–3(d)) where some change operations as suggested and performed in [3] are applied. Mapping M which induces partition P_0 and all other partitions P_1, P_2 and P_3 is indicated below. Let partition $P_0 = \{p, q, r, s, t\}$ with $p = \{1, 2, 3, 4\}, q = \{5, 6\}, r = \{7\}, s = \{8\}$ and $t = \{9\}$. The centroids for the three behavioral dimensions for partition P_0 are as follows: position: $\pi(p) = \frac{5}{26}$, $\pi(q) = \frac{1}{2}$, $\pi(r) = \pi(s) = \frac{10}{13}$, $\pi(t) = \frac{12}{13}$; repeatability: $\varrho(q) = 1$, $\varrho(\cdot) = 0$, else; optionality: $o(r) = o(s) = 1$, $o(\cdot) = 0$, else. The first model variant G_1 in Fig. 3(b) can be seen as a generalization of the original model G_0. Partition P_1 is chosen to be the same as partition P_0. Weighting all three dimensions equally we get a behavioral similarity value for P_0 and P_1 of $VSim_M(P_0, P_1) = 0.95$. Variant G_2 is obtained by refining: P_2 is as P_0 but with $q = \{5, 5a, 6\}$ and $r = \{7, 7a\}$. We get $VSim(P_0, P_2) \approx 0.998$. For G_3 sequential executions and parallel ones are mixed up. With P_3 like



(a) Model G_4 (b) Model G_5

(c) Model G_6

Fig. 4. Behavioral similarity values are computed for models G_4–G_5 and G_4–G_6

P_0 it is $VSim(P_0, P_3) \approx 0.997$. Computing the penalized behavioral similarity measures leads to in the following listed results: $penVSim_M(P_0, P_1) \approx 0.932$, $penVSim_M(P_0, P_2) \approx 0.997$, and $penVSim_M(P_0, P_3) \approx 0.995$. Besides, modeling experts were asked to assess behavioral differences of G_0 to the three variants based on their modeling experience. Common opinion was that G_1 is the most differing variant. A comparative evaluation of the centroid-based behavioral similarity measure and casual footprints is carried out in the following paragraph.

Evaluation: For the comparative evaluation performed in the following, three process models G_4, G_5 and G_6, shown in Figure 4 are considered. Models G_4 and G_5 describe the same process but were modeled by different persons. The original label descriptions have been removed and substituted by letters "A to E" to provide better readability as the focus lies only on the models' behavior. Resembling letters indicate resembling descriptions. Model G_6 describes a different process including similar tasks but with a differing activity order. Besides, not all activities have to be executed and some may be executed several times. Models G_4 and G_5 always have activities "A to E" executed exactly once.

For calculating the similarity between Models G_4 and G_5 the *p*artial *i*njective 1-to-1-mapping M_1^{pi} is established with $\{(A, AB), (C, C), (D, DE)\} = M_1^{pi}$. Activities B and E from G_4 are not mapped but results would not be different if B and E instead of A and D would have been mapped. The *b*ijective M-to-N-mapping M_1^b according to Definition 2 ist established with $\{(\{A, B\}, \{AB\}), (\{C\}, \{C\}), (\{D, E\}, \{DE\})\} = M_1^b$. These mappings provide the highest similarity, respectively, when taking into account the activities' descriptions. The mappings for the second comparison (G_4 and G_6) are the same, namely $M_2^{pi} = \{(\cdot, \cdot) \mid \cdot \in \{A, B, C, D, E\}\}$ and $M_2^b = \{(\{\cdot\}, \{\cdot\}) \mid \cdot \in \{A, B, C, D, E\}\}$.

For evaluation, three behavioral similarity values are computed for every comparison. One with help of casual footprints (cf. [11]), one with smallest casual footprints as suggested in [3], Section 6.3, IV *discussion* and one with the penalized centroid-based approach. The results are listed in Table 1. The similarity of two activities needed for calculating the (smallest) casual footprints, sometimes also known as correlation, was chosen label-based, i.e., for simplicity it was set $Sim(A, AB) = Sim(D, DE) = 0.5$ and $Sim(\cdot, \cdot) = 1 \; \forall \cdot \in \{A, B, C, D, E\}$. Obviously, all three methods state that models G_4 and G_5 are more similar than

Sim. (#*values*)	CF	smallest CF	centroid-based
$Sim(G_4, G_5)$	0.799 (294)	0.885 (90)	1.000 (30)
$Sim(G_4, G_6)$	0.640 (414)	0.632 (108)	0.333 (30)

Table 1. Similarity values are computed between models G_4, G_5 and between models G_4, G_6. CF stands for casual footprint and smallest CF is a modified casual footprint approach were some "useless" elements in look-back and look-ahead links are omitted; the numbers in brackets count the number of computed intermediate values

models G_4 and G_6 which is as desired. But differences between the assigned similarity values are substantial. Where the centroid-based approach states full behavioral similarity between models G_4 and G_5, which could be discussed if this result fits reality, because the first model's parallel gateways seem to be ignored, the casual footprint method says similarity is only about 80%, although, as stated above, when finished all activities A to E are executed exactly once. In contrast, the casual footprint method assigns a similarity of about 64% to G_4 and G_6, although these models describe completely different processes, concerning their behavior. The centroid-based approach assigns a relatively low similarity value of about 33%. For both comparisons, the smallest casual footprint approach gives values in between the two other methods.
It should be pointed out again that the centroid-based similarity value gives information about only one aspect of the compared process models. Information about labels, data and resources is not used for calculating this value. Similarity values concerning these aspects can be calculated separately and then be combined altogether. Instead, the casual footprint method needs a similarity value assigned to each pair of activities which is element of the underlying mapping. Thus, the casual footprint method does not completely separate the different process perspectives orthogonally from each other.
Another even more remarkable difference gets apparent when considering the number of calculated intermediate values (not the elementary arithmetical operations). They are shown for all three methods and for both comparisons in brackets in Table 1. It is apparent that between the common casual footprint method and the smallest casual footprint approach there is a huge difference in the number of calculated intermediate values even if the resulting similarity values do not differ that much. For the casual footprint method the number of intermediate values rises exponentially with the number of model nodes. For the smallest casual footprint approach this number is only increasing quadratic, whereas for the centroid-based approach the number of calculated intermediate values rises linearly with the number of model activity nodes.

6 Conclusion and Future Work

As shown in Section 2 there already exists a variety of techniques for calculating the behavioral similarity of process models. The methods presented in the work

at hand should not be seen as strictly better but should rather help in computing *behavioral similarity also for M-to-N-mappings*, a task that has not been examined in the related work thoroughly. A big advantage of the *centroid-based approach* is that average values can *easily be computed* and thus the presented method is suitable even for large practical applications. Furthermore, the idea of splitting process models into several perspectives like label description, data objects, etc. as already pursued in multiple similarity matching papers, is continued in this work by dividing model behavior into the three dimensions *(relative) position, repeatability and optionality.*

Some approaches for future work are already stated in the main part of the paper. Concerning execution specific features of process models, e.g., the optionality value of a node it is conceivable to use process log information to improve similarity values. Additionally, parameters, weights and maybe even formulae might be improved by applying machine learning methods on already matched process models. Another big topic for future work would be to implement M-to-N-matching methods for all process model perspectives and to run a big evaluation that combines all similarity values for the different perspectives. Thereby, a nice feature would be to bring the similarity values of all of the perspectives into a penalized form like $(Sim - pen)^+$. For the organizational perspective, i.e., for human resources it could result in a similarity of shape $(\sum_{p \in P} |A_p \cap A_{M(p)}|/|A_p \cup A_{M(p)}|)/|P|$ and penalty terms of shape $(\sum_{p \in P} pen^A(p))/|P|$ with $pen^A(p) = |\bigcup_{n \in p} A_n \setminus \bigcap_{n \in p} A_n|/(2 \cdot |\bigcup_{n \in p} A_n|) \in [0, 0.5]$ with $A_p = \bigcap_{n \in p} A_n$ (cf. [8]) and A_n being the set of agents having the competence for node n with $n \in p$. Another task could be to check and maybe adjust similarity measures to get metrics that fulfill the corresponding conditions of symmetry, non-negativity, identity and the triangle inequality. With such metrics it is possible to search huge repositories even faster for similarity with the use of metric trees [25]. Another task for future work could be to relax proces model conditions, e.g., to be feasible a process model does not need the requirement that XOR-gateways are block-structured. Further on, it might be challenging to construct/generate a "best" model using a set of similar models.

References

1. Dumas, M., La Rosa, M., Mendling, J., Reijers, H. A.: Fundamentals of business Process Management. Springer-Verlag Berlin Heidelberg (2013)
2. Dumas, M., García-Bañuelos, L., Dijkman, R.: Similarity Search of Business Process Models. IEEE2009, Bulletin of the IEEE Computer Society Technical Committee on Data Engineering (2009)
3. Becker, M., Laue, R.: A Comparative Survey of Business Process Similarity Measures. Computers in Industry 63, Issue 2, 148–167 (2012)
4. Pittke, F., Leopold, H., Mendling, J., Tamm, G.: Enabling Reuse of Process Models through the Detection of Similar Process Parts. BPM 12, LNBIP 132, 586-597 (2013)
5. Weidlich, M., Dijkman, R. Mendling, J.: The ICoP Framework: Identification of Correspondences between Process Models. CAiSE 2010, LNCS 6051, 483-498 (2010)

6. Gerth, C., Luckey, M., Küster, J. M., Engels, G.: Detection of Semantically Equivalent Fragments for Business Process Model Change Management. Services computing (SCC), IEEE International Conference, 57-64 (2010)
7. Baumann, M. H., Baumann, M., Schönig, S., Jablonski, S.: Towards Multi-perspective Process Model Similarity Matching. EOMAS, LNBIP 191, 21-37 (2014)
8. Baumann, M., Baumann, M. H., Schönig, S., Jablonski, S.: Resource-Aware Process Model Similarity Matching. *in-press (RMSOC)* (2014)
9. Cayoglu, U., Dijkman, R., Dumas, M., Fettke, P., García-Bañuelos, L., Hake, P., Klinkmüller, C., Leopold, H., Ludwig, A., Loos, P., Mendling, J., Oberweis, A., Schoknecht, A., Sheetrit, E., Thaler, T., Ullrich, M., Weber, I., Weidlich, M.: The Process Model Matching Contest 2013. BPM Workshops. Springer International Publishing, 442-463 (2014)
10. Dijkman, R., Dumas, M, García-Bañuelos, L, Käärik, R.: Aligning Business Process Models. EDOC'09, IEEE International, 45-53 (2009)
11. Dijkman, R., Dumas, M., van Dongen, B., Käärik, R., Mendling, J.: Similarity of Business Process Models: Metrics and Evaluation. Information Systems 36, Issue 2, 498-516 (2011)
12. Dijkman, R., van Dongen, B. F., Dumas, M., García-Bañuelos, L., Kunze, M. Leopold, H., Mendling, J., Uba, R., Weidlich, M., Weske, M., Yan, Z.: A Short Survey on Process Model Similarity. 25 Years of CAiSE, Seminal Contributions to Inforamtion Systems Engineering, 421-427 (2013)
13. Dijkman, R., Dumas, M., García-Bañuelos, L.: Graph Matching Algorithms for Business Process Model Similarity Search. BPM 09, LNCS 5701, 48-63 (2009)
14. Klinkmüller, C., Weber, I., Mendling, J., Leopold, H., Ludwig, A.: Increasing Recall of Process Model Matching by Improved Activity Label Matching. BPM 13, LNCS 8094, 211-218 (2013)
15. van Dongen, B., Dijkman, R., Mendling, J.: Measuring Similarity between Business Process Models. CAiSE 2008, LNCS 5074, 450-464 (2008)
16. `http://www.bpmb.de/images/BPMN2_0_Poster_DE.pdf` February 9, 2015
17. Kunze, M., Weske, M.: Metric Trees for Efficient Similarity Search in Large Process Model Repositories. BPM 2010, LNBIP 66, 535-546 (2011)
18. Kunze, M., Weidlich, M., Weske, M.: m3 - A Behavioral Similarity Metric for Business Processes. ZEUS'11, CEUR-WS, 89-95 (2011)
19. Wombacher, A.: Evaluation of Technical Measures for Workflow Similarity Based on a Pilot Study. CoopIS 2006, LNCS 4275, 255-272 (2006)
20. van der Aalst, W. M. P., de Medeiros, A. K. A., Weijters, A. J. M. M.: Process Equivalence: Comparing Two Process Models Based on Observed Behavior. BPM 2006, LNCS 4102, 129-144 (2006)
21. Mendling, J., Lassen, K. B., Zdun, U.: Transformation Strategies between Block-Oriented and Graph-Oriented Process Modelling Languages. Multikonferenz Wirtschaftsinformatik 2006 (MKWI 2006), GITO-Verlag Berlin, 297-312 (2006)
22. Kopp, O., Martine, D., Wutke, D, Leymann, F.: The Difference Between Graph-Based and Block-Structured Business Process Modelling Languages. Enterprise Modelling and Information Systems Architecture, Vol. 4, No. 1 (2009)
23. van Dongen, B. F., Mendling, J., van der Aalst, W. M. P.: Structural Patterns for Soundness of Business Process Models. EDOC'06, IEEE , 116-128 (2006)
24. Zhao, W., Hauser, R., Bhattacharya, K., Bryant, B. R., Cao, F.: Compiling business processes: untangling unstructured loops in irreducible flow graphs. Int. J. Web and Grid Services, Vol. 2, No. 1, 68-91 (2006)
25. Kunze, M., Weske, M.: Metric Trees for Efficient Similarity Search in Large Process Model Repositories. BPM 2010, LNBIP 66, 535-546 (2011)