Comparison and Retrieval of Process Models using Related Cluster Pairs

Michael Niemann, Melanie Siebenhaar, Stefan Schulte, Ralf Steinmetz

KOM – Multimedia Communications Lab, Technische Universität Darmstadt, Rundeturmstr. 10, 64283 Darmstadt, Germany

Abstract

Although increasingly IT-supported, effective techniques for process model retrieval and identification of process model differences or changes – needed for a variety of management and conformance purposes – are still challenging problems in business process management. Performing automated process comparison and finding relevant reference processes are not routine procedures for today's operational process repositories. Efficient combinations of similarity measures for various process model characteristics can still improve the performance of process comparison and retrieval. The approach at hand introduces the concept of related cluster pairs, parameterises it with semantic, string-based, and novel hybrid metrics for comparing process models, and defines a novel similarity notion for process model retrieval. Evaluations with process data from the SAP reference model show that our approach outperforms current related work and established text search engines.

1. Introduction

In times coined by constantly changing market conditions, new competitive threats and increasing numbers of legal regulations, companies and their process management face new challenges. The resulting flexibility challenges and the required continuous adaptation of internal processes often exceed the reaction and implementation capabilities of process repositories at companies. The ability to react quickly and efficiently to environmental changes by adapting strategy and procedures to new conditions in a sustainable manner is considered to be a crucial competitive factor [1]. Companies with high internal flexibility potential are believed to more probably prevail in the long term [2].

In recent years, companies have increasingly specified their procedural knowledge as process models. Challenges that involve the management of large process model repositories are, however, manifold. Today, immense amounts of process models cause isolated storage of different versions or variants of process models, models with overlapping application scopes, and models at different granularity levels [3]. Efficient process portfolio management that can cope with these increasing flexibility challenges requires extended and improved functionality of process model repositories. Common challenges cover, for example, the realisation of process reuse, efficient storage of large process model numbers, successful process comparison and retrieval for various purposes, operation of effective variant and version management, and implementation of reference processes as well as conformance management [3]. The efficiency of exploitation of procedural knowledge contained in a company's process models bears improvement potential. However, tool support addressing these challenges is yet scarce [4].

Email address: <firstname>.<lastname>@kom.tu-darmstadt.de (Michael Niemann, Melanie Siebenhaar, Stefan Schulte, Ralf Steinmetz)

Over the past years, a number of improvements for process model matching and retrieval have been introduced. Amongst others, contributions cover detailed investigations of various node similarity metrics [5, 6], variant identification based on structural decomposition [7], metrics based on graph matching algorithms (e.g., [8, 9]), semantic approaches (e.g., [10]), and behavioural similarity of process models (e.g., [11, 12]).

The approach at hand applies the concept of *related cluster pairs* on both comparison and retrieval of process models. We developed a technique to determine node-based differences between two process models by disaggregating them into subgraphs that are related by similarity. The focus of the presented approach is the investigation of node labels, rather than in-depth analyses of structural or behavioural aspects. For the determination of word and node label similarity, we apply a number of existing and novel node label similarity metrics. While the resulting node assignments are used to identify the delta between two process models, we also use related cluster pairs to calculate process model similarity. We apply this measure in the context of process model retrieval and compare our results to those achieved by established text search engines as well as current related work. For both application scenarios, we perform evaluation experiments using the SAP reference model.

The remainder of this paper is structured as follows. We discuss related work in Section 2. In Section 3, we introduce the utilised similarity metrics and the word preprocessing techniques we use. We outline the computation of related cluster pairs and present a novel process model similarity measure in Section 4. The evaluation of the approach in the first scenario, comparison of process models, is presented in Section 5. In Section 6, we test the process model similarity measure in the context of retrieval of process models, which is evaluated using a second annotated test case that has kindly been provided by the authors of [5]. Further, we discuss the used test case. Section 7 concludes the paper.

Table 1: Overview of related work

Approaches	Node assignments	String-based node similarity	Semantic node similarity	Hybrid node similarity	Structural process similarity	Process model differences	Change suggestions	(Semi-) automated merging	Result Visualisation
Andrews et al. [13]	–	–	–	–	–	•	–	•	•
Dijkman [14, 15]	–	–	–	–	•	•	–	–	–
Dijkman et al. [5, 6]	•	•	•	–	•	–	–	–	–
Ehrig et al. [10, 16]	•	•	•	•	–	–	–	–	–
Küster et al. [7]	–	•	–	–	•	•	•	•	•
Li et al. [12, 17]	–	–	–	–	•	•	•	–	•
Lu and Sadiq [18]	•	–	–	–	•	–	–	–	–
Madhusudan et al. [19]	•	•	–	–	•	•	–	–	–
Melnik et al. [9]	•	•	–	–	–	–	–	–	–
Minor et al. [20]	•	–	–	–	•	•	–	–	–
Related Cluster Pairs	•	•	•	•	•	•	–	–	•

2. Related Work

In this Section, we provide an overview of related approaches focusing on approaches that target structural and node label-based similarity, as we can compare our technique with these approaches in particular.

Andrews et al. [13] present an approach for visual graph matching. The prototype analyses similarities of graphs and suggests a merged graph, visualising the results. It provides means to the user for final visual assessment and manual node assignment in order to adjust the results. The user can manually edit the resulting graph by, for example, replacing labels or changing a node's position. The approach assumes the external provision of node similarities.

Dijkman [14] outlines an approach to determine process model differences of Event-driven Process Chains (EPC). A detailed computation of differences is conducted, where the type of a difference using the difference topology introduced in [15] as well as the exact position of the differences is determined. For the computation, formal semantics are utilised. The approach has exponential computation complexity and thus requires repeated scoping of the process models. Manual node assignments are required as input.

In [5], Dijkman et al. propose and compare three different similarity metrics for the problem of retrieving business process models in a given repository. In detail, they utilise label matching similarity to determine the similarity of node labels, structural similarity, and behavioural similarity to consider the node labels and causal relationships in the model. For the semantic node similarity, they consider word synonyms. The approaches determine a final similarity score, but do not indicate the location of similarities and differences.

Ehrig et al. [10] propose a (semi-)automatic approach to detect similar elements in business process models. The authors compute similarities based on semantic information. They make use of the Pr/T net ontology, which represents a description of Petri net elements based on OWL-DL introduced in [16]. The actual comparison considers semantic and string-based node similarity measures, which are combined to a similarity measure for concept instances. The concept instances are compared to each other and the similarity values are aggregated to an overall similarity for the two process models. Node similarities and process differences are not explicitly identified.

Küster et al. [7] introduce an approach for process model comparison that compares different versions of a process model in the absence of a change log. The authors use single-entry-single-exit (SESE) fragments and assume externally provided node correspondences based on string-based node similarity. Based on the computation of differences, they derive change suggestions. The differences and change operations are then grouped and associated to the affected SESE fragments. The information about the structural similarity and its association to the SESE fragments permits to create a hierarchical change log that can be applied to resolve all or parts of the differences in order to obtain a consolidated model. The approach was designed to consider different versions of the same process model and has not been evaluated.

Li et al. [12] introduce an approach for a Process-Aware Information System that identifies a generic process reference model for a given set of variants. Based on an aggregated order matrix, they determine activities to be clustered as blocks. Referring to the blocks, they investigate the behavioural similarity (ordering) of activities. The algorithm is validated using simulation on 7000 process models in [17]. The activity assignment matrix, however, is an external requirement.

Lu and Sadiq [18] propose a search approach for retrieving process variants based on the identification of process model fragments. Given a process query, they use three basic structural similarity measures (equality, subsumption, and implication). They consider a number of features per process model and define a similarity function for each of them. The authors do not evaluate their approach.

Madhusudan et al. [19] propose a case-based reasoning approach for workflow modelling and design support. Their approach is based on the so-called similarity flooding by [9]. It consists of a structural process similarity for retrieval, and is based on initial similarity values. The authors state that these values are based on "Natural Language Processing and string-matching techniques". They do not provide an evaluation of their approach.

Melnik et al. [9] present a further graph matching algorithm that determines a mapping between the corresponding nodes of two given graphs. The approach is applicable in different scenarios with diverse data structures, such as matching of two data schemas in data warehousing applications or comparison of process model graphs. The *similarity flooding* represents an iterative fixpoint computation to determine the set of similar nodes. It is based on the assumption that two nodes are more likely to be similar if their adjacent nodes are similar. For the determination of node similarities they use a string-based node similarity. A computation of differences is not performed.

Minor et al. [20] propose an index-based retrieval approach for workflows. The approach targets the search among past or changed workflows to assist authoring of recent workflow instances. They apply structural similarity using graph-edit dis-

tance and do not provide an evaluation.

Generally, for process models, the notions of node label similarity, structural similarity, and behavioural similarity are distinguished. Van der Aalst et al. [11] provide an approach for comparing a process model with a set of event logs based on a behavioural similarity measure. Further approaches consider behavioural similarity for process models [5] and state charts or finite state machines [21, 22]. The problem of correct node assignments, however, represents a fundamental requirement for fully automated process comparison and retrieval approaches, which in turn consider, for example, structural or behavioural process characteristics. Further, it can directly affect the results of these relying approaches, if improved. For this purpose, amongst others, we investigate the integration of semantic node label similarity metrics

Mendling et al. [23] provide interesting related fundamental research in this respect, although their intention is slightly different from the one at hand. They classify verbs occuring in node labels of the process models in the SAP reference model using two established verb classification schemes. Based on the most frequently occuring verb classes, they propose a set of icons to support process modelling practice. Performing a general classification of node labels using the two schemes, their approach achieves a coverage of 0.68 and 0.44, respectively. As a general problem, they identify missing specificity of node labels (e.g., by frequent usage of the verb "to process" in labels). The authors do not aim at investigating the quality of the classification nor do they provide details of the approach. Differing from their approach, we use (amongst others) a technique from natural language processing (NLP), *lemmatising*, and test it, on annotated test sets, aiming at improving current classification result quality.

The approach at hand combines a variety of label-based similarity approaches to tackle this problem. We apply string-based and semantic similarity measures, and by their combination create hybrid similarity measures. It is the general focus to improve the exploitation of label meanings using considerate word preprocessing and reduction techniques in order to improve assignment quality. Additionally, we simultaneously consider node label similarity and structural characteristics of process models. In contrast to Ehrig et al. [10], we do not demand explicit semantic description (e.g., using ontologies), but rely on and exploit the meaning of words in the context of their label(s). As a core part of our approach, we compute *word* and *node label similarities* for process model comparison and retrieval, where a *node label* is a sequence of *words*. Based on these, we determine node assignments, region differences, similarities of process model fragments, and the similarity of process models.

3. Similarity Measures and further Preliminaries

In this section, we outline the fundamentals of our approach. In particular, we introduce the node label similarity measures and word preprocessing techniques we use. For each measure, we provide an example calculation including nodes from the process model pair shown in Figure 2 (on p. 6). Further, we introduce a definition for process model graphs.

Throughout our approach, we consider process models as graphs we call *process model graphs* (PMG).

Definition 1 (Process Model Graph). *Let G be a graph $G = (V, E)$, Λ be a set of labels and Θ be a set of types. A process model graph P is a directed, weakly connected graph defined as tuple $P = (V, E, \lambda, \tau, \alpha)$, where:*

- *V is a finite set of nodes,*
- *$E \subseteq (V \times V)$ is a finite set of edges,*
- *λ is a labelling function: $\lambda : (V \cup E) \to \Lambda$ that assigns labels to nodes and edges,*
- *$\tau : (V \cup E) \to \Theta$ assigns types to nodes and edges, and*
- *$\alpha : (V \cup E) \to (A \to \Lambda)$ assigns attributes to nodes and edges, where A is a set of attributes that are assigned labels.*

In particular, the sets A, Θ and Λ all include ϵ (the `NULL` element).

This definition (adopted from [5]) for process model graphs is applicable to describe a variety of graph-based process description languages, including EPC.

3.1. Word preprocessing

In all cases, we perform preprocessing of words. Generally, we transform labels to lower case and remove punctuation (`.,-;/`) as well as single-character-words, and perform stop word removal. Stop words are frequent function words (such as "the", "as", "in", "over", "by", ...). We decided, however, not to remove words like "with", "without", "needs", "not" or similar ones that are part of common stop word lists, as these can influence and essentially coin the meaning of the activity label they are part of. The intention is to avoid turning node labels with conflictive meaning using similar words into sentences with similar meaning, e.g., "Capacity is available" and "Capacity is not available"[1]. This is a common problem in NLP that must be coped with.

When applying string-based measures, we optionally use Porter's stemming algorithm [24]. Stemming is a rule-based method to reduce similar words to a simple stem. For the words "orders", "warehousing", "adjusting", and "receive", the results of Porter's algorithm are "order", "wareh", "adjust", and "receiv", respectively. Further, we consider the usage of a more advanced method than stemming for word reduction, *lemmatising*, useful. The lemma of a word is its actual base form. This reduction techniques preserves the word meaning. Lemmatising, however, requires and depends on the *part-of-speech (POS)* of a given word. For example, "purchasing" can be lemmatised to the verb "purchase" or the noun "purchasing", depending on the context. If these information are given, words can be automatically reduced to their base form, such as "are" to "be", "communities" to "community", and "warehousing" to "warehouse". We use the lemmatising approach introduced and realised by Schmid (1994) [25].

[1] This is the case in process model ID `1In_b7s7` in the SAP reference model.

3.2. String-based node label similarity

String-based metrics calculate the similarity of given words or sentences based on their syntactic elements (characters) without regarding their meaning. In the following, we introduce the measures we use. For each one, we provide an example using nodes from the examplary process model pair provided in Figure 2.

A common string-based similarity measure is *Levenshtein's string-edit-distance*. For two strings, or node labels L_1 and L_2, it computes insert and deletion operations required to transform L_1 into L_2. Each operation is assigned a cost of 1. The sum is the result of the metric $lev(L_1, L_2)$ [26]. We apply the metric sim^{Lev} defined as

$$\text{sim}^{\text{Lev}}(L_1, L_2) = 1 - \frac{lev(L_1, L_2)}{\max(|L_1|, |L_2|)}, \tag{1}$$

where $|L_1|$ designates the length of string L_1 in terms of characters. The strength of this metric is the identification of the same characters for words occurring in the same sequence. Simple plurals, word variants, or verb conjugations are generally well recognised by this measure. Referring to the example in Figure 2, the similarity value for the labels "transfer to warehouse" (node g) and "warehousing" (node y) is 0.33 (using stop word removal).

In order to address its deficiencies, e.g., in case one of the two labels to be compared contains long additional words, or when processing alternating words (e.g., "transferring to warehouse" and "warehouse transfer", where $lev = 0.11$), we use the Jaccard distance measure [27, 28] sim^{Jac} as *set* metric. A set is assumed to be a sequence of words, divided by a common separator. A set A of length $|A| = n$ is supposed to consist of n words referenced as A_i. Especially, the union $A \cup B$ does not contain identical words. sim^{Jac} divides the number of identical words by the number of all (differing) words:

$$\text{sim}^{\text{Jac}}(L_1, L_2) = \frac{|L_1 \cap L_2|}{|L_1 \cup L_2|}. \tag{2}$$

For example, the labels "transferring to warehouse" and "warehouse transfer" are assigned a similarity value of 0.25, using stop word removal and lemmatising even 1.0. Referring to Figure 2, the similarity $\text{sim}^{\text{Jac}}(f, x)$ equals to $\frac{2}{3}$.

As this measure proceeds from word level, simple word variants are not understood as the same word. The labels "transfer to warehouse" and "warehousing", e.g., are not recognised as similar. Word preprocessing or the combination with other metrics normally remedy this deficiency (cf. Section 3.1).

3.3. Semantic node label similarity

However, string-based similarity metrics for node comparison can lead to significantly wrong similarity interpretations. Comparing "goods" to "good", as well as "procurement" to "purchasing", for example, leads to undesired results. Automatically revealing relationships in these cases requires the consultation of word corpora or lexicons. These are mostly represented as graphs based on word senses like, e.g., the WordNet corpus [29][2], or Wikipedia[3]. Semantic measures that can be used for label comparison are amongst others introduced in [10, 29, 30]. As semantic measures generally compute the similarity for two words, word similarity values need to be aggregated on node label level. For this, we use additional aggregating functions.

As semantic similarity indicators, we use three metrics: a word synonym and a word distance metric both based on WordNet, as well as the "distributional semantic first order" based on Wikipedia. Again, we provide examples referring to the process model pair provided in Figure 2.

WordNet contains the most frequent English words, organised in synsets. A synset is a collection of synonymous words. As a word can have more than one meaning, each word can be contained in several synsets. So, using WordNet, we can systematically derive a set of words that share any of the meanings of a given word. By $\text{sim}_a^{\text{ws}}(w_1, w_2)$, we calculate the similarity of two words (w_1, w_2) based on their WordNet synsets:

$$\text{sim}_a^{\text{ws}}(w_1, w_2) = \begin{cases} 1 & \text{if } w_1, w_2 \text{ are identical words} \\ a & \text{if } \exists \text{ synset } S \text{ with } w_1, w_2 \in S \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

The metric assigns two identical words the value 1.0, while the similarity of synonymous words is indicated by the parameter a (with $0 < a < 1$). However, this definition does not provide the aggregation of word similarities to node label similarities. In preliminary experiments, we identified two scenarios with acceptable results, depending on the aggregation function and the parameter a.

In the first one, we choose $a = 0.75$ and aggregate by taking the mean value of the word similarity values. This metric has also been specified in [6]:

$$\text{sim}^{\text{S}_{\text{mean}}}(L_1, L_2) = \frac{|L_1 \cap L_2| + \sum_{(v,w) \in L_1 \setminus L_2 \times L_2 \setminus L_1} \text{sim}_{0.75}^{\text{ws}}(v, w)}{\max(|L_1|, |L_2|)} \tag{4}$$

In the second scenario, we use the *Monge Elkan similarity metric* sim^{ME} as aggregating metric. It is defined as

$$\text{sim}^{ME}(A, B, \text{sim}^{M}) = \frac{1}{|A|} \sum_{i=1}^{|A|} \max_{j=1}^{|B|} \text{sim}^{M}(A_i, B_j), \tag{5}$$

where sim^{M} can be any metric that operates on words [31]. This metric also uses the dissection of labels into words as outlined prior to Equation 2. As above, the metric distinguishes atomic strings and subfields. An atomic string is either a single word or an abbreviation, while a subfield is a part of a field (or word set) containing atomic strings, i.e., an atomic string represents the smallest, possible subfield. In order to determine the degree of similarity between two given fields A and B, a subfield w_i of the first field A is compared to every subfield of the second field B. The metric returns the mean of the *maxima* of all w_j. For computation of the similarity, the metric wraps the metric $\text{sim}^{M}(A_i, B_j)$.

[2] http://wordnet.princeton.edu

[3] http://wikipedia.org

We choose the Monge Elkan as the aggregating metric and, based on the results of preliminary experiments, define $a = 0.5$ for the second synonym metric:

$$\mathrm{sim}^{\mathrm{S_{ME}}}(L_1, L_2) = \mathrm{sim}^{ME}(L_1, L_2, \mathrm{sim}^{\mathrm{ws}}_{0.5}) \quad (6)$$

As an example, consider the two node labels "client order processing" and "handling of customer order". Using stop word removal, the result using $\mathrm{sim}^{\mathrm{S_{mean}}}$ is $\frac{1.0+0.75+0.0+0.0+0.0}{3} = 0.583$, and using $\mathrm{sim}^{\mathrm{S_{ME}}}$ it is $\frac{0.5+1.0+0.0}{3} = 0.5$. In the process model pairs given in Figure 2, for example, the nodes g and y are assigned a similarity value of 0.50 by both metrics (using stop word removal and lemmatising). Again, slight word variants such as plural forms, are not recognised as identical words unless word preprocessing is performed.

However, for example, the relationship between "purchasing" and "procure" is not identified by these measures. Terms that are not synonyms but still relate, such as hypernyms or hyponyms, are not considered. Therefore, we use a further metric, the word distance. Based on WordNet, it refers to the length of the shortest path between two words w_1 and w_2 in the WordNet taxonomy ($\delta^{\mathrm{WordNet}}(w_1, w_2)$). In order to process label pairs, we define $\Delta^{\mathrm{WN}}(L_1, L_2)$ as

$$\frac{1}{|L_1 \setminus L_2|} \sum_{w_1 \in L_1 \setminus L_2} \min\left\{\delta^{\mathrm{WordNet}}(w_1, w_2) \mid w_2 \in L_2 \setminus L_1\right\}. \quad (7)$$

The metric removes identical words from both labels. We consider all possible POS and word relations (e.g., hyponym, hypernym). The metric calculates the mean of the minimum values of the distances per word in L_1 and all words in L_2. We define the inverse WordNet distance as

$$\mathrm{sim}^{\mathrm{WND}}(L_1, L_2) = \frac{1}{\Delta^{\mathrm{WN}}(L_1, L_2)}. \quad (8)$$

A strength of this metric is the recognition of definite word relationships that are not identified as being synonyms. The shortest path between the words "procure" (node b in Figure 2a) and "purchasing" (node n in Figure 2b), e.g., is 4, leading to a similarity value of 0.25.

WordNet is a collection of selected word meanings, whose relationships are professionally maintained. Thus, its coverage of word meanings is limited. As a valuable resource for word meanings, Wikipedia, in contrast, is constantly updated by thousands of volunteers. Word relationships are not explicitly defined in Wikipedia. Therefore, we compute the corpus-based similarity of words using the so-called "distributional semantic first order similarity" measure by Lin [30]. Two given words w_1 and w_2 are compared by their dependency triples using a context window size of ±3 words. Moving the window over the corpus results in a set of dependency triples for a given word w_x. A dependency triple is of the form (w_x, r, w'), where w_x represents the word whose context is examined, w' is a word occurring in the context of w_x, and r refers to the relationship between w_x and w' (i.e., the relative position of w' with respect to w_x) [30].

$$\mathrm{sim}^{\mathrm{Lin}}(w_1, w_2) = \frac{\sum_{(r,w')}(w_1, *_r, *_{w'}) + (w_2, *_r, *_{w'})}{\sum_{(r,w')}(w_1, *, *) + \sum_{(r,w')}(w_2, *, *)} \quad (9)$$

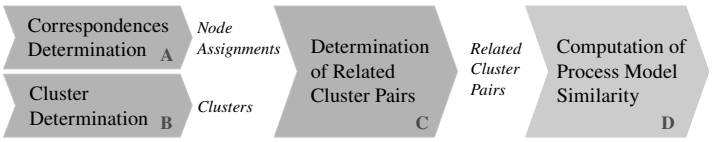


Figure 1: Computation of Related Cluster Pairs

The measure is based on the assumption that the similarity between two words can be expressed as the amount of information contained within the dependency triples which are common to both words, divided by the amount of information contained in all the dependency triples of w_1 and w_2 that match the pattern $(w_1, *, *)$ and $(w_2, *, *)$, where $*$ is a wildcard for r and w', respectively. For application on node labels we define $\mathrm{sim}^{\mathrm{SFO}}$:

$$\mathrm{sim}^{\mathrm{SFO}}(L_1, L_2) = \mathrm{sim}^{\mathrm{ME}}(L_1, L_2, \mathrm{sim}^{\mathrm{Lin}}) \quad (10)$$

The metric identifies word similarities exceeding synonym relationship, based on the linked context in the Wikipedia corpus.

Using this measure, for example, the node labels "determine level of significance" and "assess relevance" have been correctly matched in our experiments, although only two out of six words are synonyms. Further, the example provided in Figure 2 has been computed using this measure. The computed similarity values are provided in Table 2.

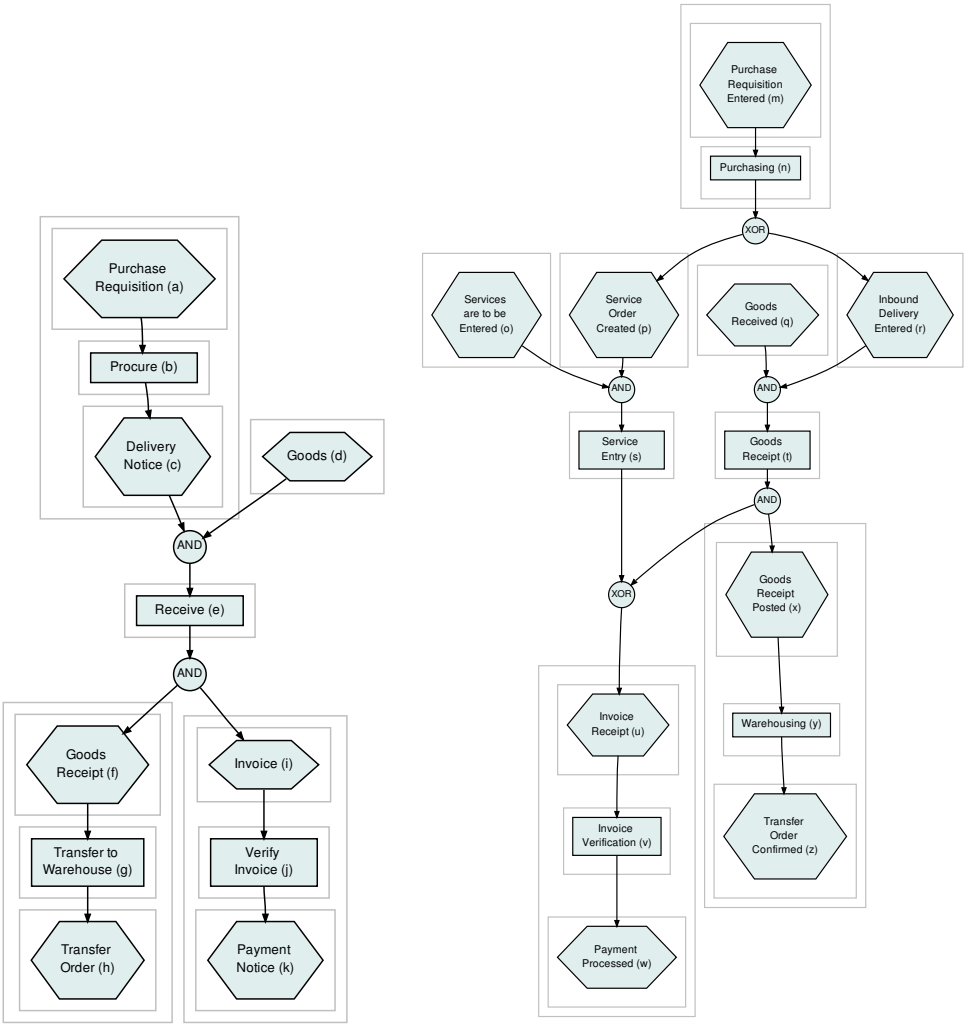
4. Related Cluster Pairs

Related cluster pairs represent mutually assigned (as well as unassigned) subgraphs of graph-based process models (PMG) containing pairwise assigned nodes (cf. [32]). Their computation consists of three major steps (cf. Sections 4.1, 4.2, and 4.3) and a fourth step (cf. Section 4.4) that covers the computation of the process model similarity (cf. Figure 1). Additionally, we outline these steps along an example provided in the Figures 2 and 3, as well as Table 2. In Section 4.5, we discuss the computational complexity and limitations.

4.1. Correspondences Determination

In step A, node correspondences are determined. We apply a number of measures covering word similarity and node label similarity, as well as word preprocessing techniques (cf. Section 3). In order to achieve optimal unique node assignments (in terms of calculated similarity), the resulting matrix is optimised by solving the assignment problem. Hence, this step consists of two inner steps: determine potential match candidates per node ($A1$), and solve assignment problem ($A2$).

As part of $A1$, per node type, all nodes of one graph are compared to all nodes of the other graph. We compute similarity values for node pairs using the introduced word preprocessing techniques (cf. Section 3.1), the measures outlined in Sections 3.2 and 3.3, as well as combinations of them (hybrid measures). All measures calculate a value for a given label pair $(\lambda(n_1), \lambda(n_2))$ for nodes n_1 and n_2. Generally, we consider two



(a) First process model (P_1)

(b) Second process model (P_2)

Figure 2: Example pair of process models: visualisation of determined clusters (grey rectangles)

nodes similar concerning a metric sim_i, if the similarity value of their label pair exceeds a threshold t_i: $\text{sim}_i(\lambda(n_1), \lambda(n_2)) \geq t_i$. Further, in this case, we call node n_1 correspondent of node n_2. For the hybrid measures, we use the following weighting ($\oplus$):

$$\text{sim}(n_1, n_2) = w_{\text{Sem}} \, \text{sim}_{\text{Sem}}(n_1, n_2) + w_{\text{Str}} \, \text{sim}_{\text{Str}}(n_1, n_2), \quad (11)$$

where $w_{\text{Sem}} + w_{\text{Str}} = 1$ and $0 \leq w_{\text{Sem}}, w_{\text{Str}} \leq 1$.

Node pairs with a similarity value above a threshold t are added to a match list which is specified for each single node. We learn the thresholds for the measures by cross-validation (cf. Section 5.1). All other nodes are added to a list of unassigned nodes. After computation, every node's match list contains all potentially corresponding nodes of the other process model graph in terms of similarity. Combining these match lists results in an $m \times n$-assignment matrix (with m and n indicating the respective node counts).

In the second step ($A2$), the node assignment matrix is optimised by solving the assignment problem. We use it for maxi-

Table 2: Similarity values and node assignments for process models P_1 and P_2 (cf. Figure 2) using the measure sim^{SFO}

$n \in P_1$	*List of match candidates from* P_2				
a	m: 0.6806	w: 0.0288	r: 0.0268	q: 0.0242	o: 0.0232
	z: 0.0211				
b	n: 0.0332				
c	r: 0.3402	w: 0.0337	z: 0.0276	o: 0.0215	
d	q: 0.5040	x: 0.3352			
e	t: 0.0208				
f	x: 0.6685	u: 0.5155	q: 0.5168	w: 0.0246	
g	y: 0.5065				
h	z: 0.6701	p: 0.3422	w: 0.0254	r: 0.0226	m: 0.0221
i	u: 0.5155				
j	v: 0.5121	t: 0.0216			
k	w: 0.5030	u: 0.0360	q: 0.0258	r: 0.0257	z: 0.0238
	m: 0.0231	x: 0.0208			

mising the overal sum of similarity values of selected node assignments. We further assume, without loss of generality, that the larger process model contains n nodes, i.e., $n \geq m$. Further, when comparing two process models based on nodes, node counts most probably differ. As the original assignment problem deals with square assignment matrices, we modified the problem model to fit rectangular matrices according to the proposal by Bourgeois and Lassalle [33, 34]. This way, the result of the algorithm – additionally to the optimal assignments concerning similarity values – indicates all nodes that are not assigned a correspondent in the other process model. To allow unassigned nodes in the smaller model, we ignore all assignments between nodes with a similarity value of zero that have been selected by the algorithm. According to Bourgeois and Lassalle [33], the mean execution time of this algorithm is $O(cn^2m)$, where c is a constant (and $n \geq m$).

As an example, Table 2 provides the similarity values and node assignments determined for the process model pair shown in Figure 2. For this example, we used the measure sim^{SFO} (cf. Equation 10). Nodes from P_2 that do not appear in the lists have been assigned a similarity value below the threshold (here $t = 0.02$) for the respective combination. Underlined node IDs indicate that the correspondig node from P_2 has been assigned to the respective node from P_1.

4.2. Cluster Determination

In parallel to step A, in step B, we consider structural process fragments, i.e., SESE regions and clusters. SESE regions are subgraphs of a process model graph having but one incoming and one outgoing arc. They were originally introduced for the construction of program trees in compiler analysis [35, 36], and have also been applied to process model variant analysis of process models [7]. The basis for related cluster pairs are the the so-called *clusters* that are, intuitively, typed SESE regions. Clusters are defined as follows.

Definition 2 (Cluster). *Let $P = (V_P, E_P, \lambda_P, \tau_P, \alpha_P)$ be a PMG and Θ a set of cluster types. A cluster L in P is a connected subgraph $(V, E, \lambda, \tau, \alpha, t)$ such that*

- *$V \subseteq V_P$ (nodes) and $E \subseteq E_P$ (edges)*
- *$S = (V, E)$ is a SESE region:*
  $|\{(u, v) \mid (u, v) \in E_P \wedge v \notin V\}| = 1 \quad \wedge$
  $|\{(u, v) \mid (u, v) \in E_P \wedge u \notin V\}| = 1$
- *$\lambda = \lambda_P$, $\tau = \tau_P$ and $\alpha = \alpha_P$ are functions as in Def. 1*
- *the function $t : \{L\} \rightarrow \Theta$ assigns a type to the cluster*

The set of cluster types Θ covers clusters that contain *single nodes*, *node or cluster sequences*, *node or cluster loops*, and *split/join constructs of nodes or clusters*. Trivially, each node that is not a gateway represents a cluster. Further, clusters are typically part of parent clusters, i.e., represent nested clusters.

Figure 2 shows all clusters that have been identified for an exemplary process model pair. Clusters are shown as grey rectangles and are either nested or disjoint. In the next step, these structural information are combined with the node similarity values computed for each node pair.

4.3. Determination of Related Cluster Pairs

Based on the determined node assignments and identified clusters, in step C, related cluster pairs are determined. For their computation, the information about node similarity and process regions are combined.

Definition 3 (Related Cluster Pair). *Let L_1 and L_2 be clusters $L_1 = (V_1, E_1, \lambda_1, \tau_1, \alpha_1, t_1)$ and $L_2 = (V_2, E_2, \lambda_2, \tau_2, \alpha_2, t_2)$, and $V_Q \subseteq (V_1 \times V_2)$ a set of nodes. A related cluster pair Q is defined as $Q = (V_Q, \text{sim}^{Node}, t)$, where sim^{Node} is a node similarity function and t a similarity threshold $t \in \mathbb{R}$ with $0 \leq t \leq 1$.*
For all $(x, y) \in V_Q$, the following conditions must hold:

- *$\text{sim}^{Node}(\lambda_1(x), \lambda_2(y)) \geq t$ (Similarity of nodes)*
- *$\nexists (v, w) \in V_Q : v = x \vee w = y$ (Unique node assignments)*
- *$\tau_1(x) = \tau_2(y)$ (Equality of node types)*
- *$t_{L_1}(L_1) \sim^{ctSim} t_{L_2}(L_2)$, (Similarity of cluster types)*

where $\sim^{ctSim}$ is a binary relation specifying the similarity of the cluster types contained in Θ (cf. Def. 2).

Initial related cluster pairs consist of two smallest possible subgraphs (clusters) in different models, each consisting of at least one node (e.g., "Cluster 12" in Figure 3). In order to form the largest possible related clusters pairs, clusters are merged in a hierarchical as well as sequential manner. Adjacent initial related clusters pairs and unrelated clusters are simultaneously enlarged, respectively. As result, all unassigned nodes are also organised in clusters. While the latter are merged per model, the merging of the first ones is model spanning and adheres to conditions. The first condition demands from an adjacent node B to node A in model P_1, that its corresponding node B' in model P_2 is adjacent (on the same side) to A', which is the correspondent to A. The second condition is, that the resulting node groups must in turn be (related) clusters.

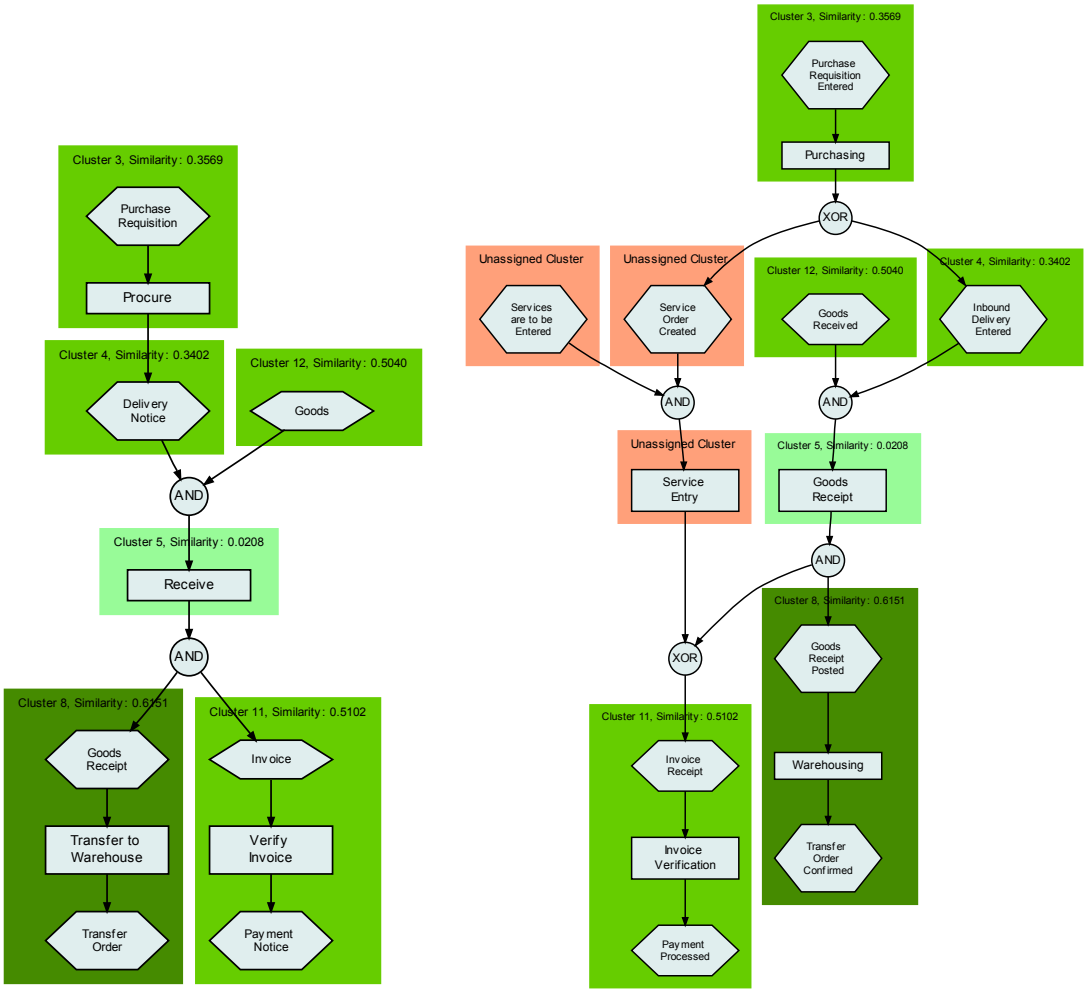
The aggregation of neighbouring clusters creates sets of largest possible related clusters. The result of the computation of related cluster pairs contained in the process model pair from Figure 2 are shown in Figure 3. Referring to P_1 in Figure 3a, cluster 11 has been constructed from three one-node-clusters. The size of three nodes could be realised, because three similar nodes have been found in P_2 (Figure 3b). For clusters 3 and 4 in P_1 (cf. Figure 3b, cf. also Figure 2a), for example, this is not the case.

Using this technique to decompose process models, we are able to identify differences between the models, to determine the similarity of cluster subgraphs, and to infer an overall similarity notion for process models (cf. Section 4.4). The computation and results visualisation (cf. Figure 3) has been realised as a plug-in for ProM[4].

4.4. Process Model Similarity

Based on the information gathered throughout the computation of related cluster pairs, we define a novel process model similarity measure (step D). The related cluster pairs used for the computation of the similarity of process model graphs P_1 and P_2 are shown in Figure 3.

[4] cf. `http://www.promtools.org/prom5/`



(a) First process model (P_1)

(b) Second process model (P_2)

Figure 3: Identification of related cluster pairs: computation results.

Let N be the set of all nodes of P_1 and P_2, i.e., $N = V_{P_1} \cup V_{P_2}$. Based on the identified related cluster pairs, we classify all nodes in N into two disjoint node sets A and U, i.e., $A \cup U = N$ and $A \cap U = \emptyset$. The set of nodes with *assigned* node correspondents is denoted as $A = \left\{n \in N \mid \text{sim}^{\text{Node}}(n, \cdot) \geq t\right\}$. For sim^{Node} and t we refer to Definition 3, while we define $\text{sim}^{\text{Node}}(n, \cdot)$ as the similarity value for node n, calculated by sim^{Node} for the assignment to its correspondent. Analogously, U is the set of all *unassigned* nodes from N, i.e., all nodes residing in unrelated clusters: $U = N \setminus A$. For a related cluster pair consisting of clusters L_1 (in P_1) and L_2 (in P_2), we refer to the number of nodes contained in cluster L_1 ($n \in V_{L_1}$) with $\left|V_{L_1}\right|$ (cf. Def. 3). Further,

$$A = \bigcup_{i=1}^{m_{\max}} A_i \quad \text{and} \quad A_k = \{n \mid n \in V_L \wedge |V_L| = k\} \tag{12}$$

where $i, k \in \mathbb{N}$ and $m_{\max}$ indicates the size of the largest related cluster pairs of the current comparison, and V_L is the node set of a cluster L. Each node $n \in A_k$ is contained in a related cluster L with node size $|V_L| = k$. Accordingly, for example, A_3 refers to the node set that is part of related cluster pairs with size 3.

Based on these definitions, we define the node-based similarity $\text{sim}^{\text{Node}}_{\text{PM}}(P_1, P_2, \text{sim}^{\text{Node}})$ of two process models P_1 and P_2 as follows:

$$\frac{\sum_{n_i \in A_1} \text{sim}^{\text{Node}}(n_i, \cdot) + \sum_{m=2}^{m_{\max}} (1 + q(m)) \sum_{n_j \in A_m} \text{sim}^{\text{Node}}(n_j, \cdot)}{|A| + |U| + \sum_{m=2}^{m_{\max}} q(m) \cdot |A_m|} \tag{13}$$

As weighting function $q : \mathbb{N} \to \mathbb{N}$, we use $q(m) = m + 1$, where m is the index of A, i.e., the number of nodes the current cluster contains. While smallest clusters (with size 1) are generally weighted with 1, all other clusters are weighted according to their node size. The larger a cluster is, the higher are the weights for the contained node similarity values; q specifies the intensity of the weighting. The overall similarity measure represents a dynamic weighted average based on q. Our metric hence emphasises large similar node sequences.

As an example, let P_1 be the model shown in Figure 3a and P_2 be the one outlined by Figure 3b. We use the similarity metric $\text{sim}^{\text{Node}} = \text{sim}^{\text{SFO}}$, i.e., the distributional semantic first order based on Wikipedia (cf. Eq. 10) and the threshold $t = 0.02$ (that has been learned using cross validation). Note that the metric does not consider gateways of EPCs, the numbers of nodes hence refer to functions and events only. While their total number is $|N| = 25$, we see that $|A| = 22$ nodes have correspondents and $|U| = 3$ do not. With, obviously, $m_{\max} = 3$, we retrieve $|A_1| = 6, |A_2| = 4$, and $|A_3| = 12$ (cf. Eq. 12). With $q(m) = m + 1$, we calculate for the metric $\text{sim}^{\text{Node}}_{\text{PM}}(P_1, P_2, \text{sim}^{\text{SFO}}) = \frac{1.73+(1+3)\cdot 1.4276+(1+4)\cdot 6.7518}{22+3+3\cdot 4+4\cdot 12} = \frac{41.1994}{85} \approx 0.4847$. Generally, the two related cluster pairs containing 3 nodes in each model are strongly emphasised, i.e., weighted five times higher than single assigned nodes. Cluster 5 with a very low similarity value and the 3 unassigned clusters decrease the overall similarity value.

4.5. Computational Complexity and Limitations

Generally, the primary intention of this approach is to improve node assignment quality, rather than to provide a very efficient approach. As also experienced in [5], the computation of string-based metrics is "very fast". Additionally, in our implementation, the runtime of dictionary lookups for the semantic metrics could be significantly improved by implementing simple cache mechanisms. For example, the runtime for the computation of the related cluster pairs in the process model pair shown in Figures 2 and 3 took 217 ms. While the runtime of a specific computation heavily depends on the available computational power and the efficiency of the implementation, the O-Notation provides general comparability.

Concerning the computational complexity of this approach, the calculation of the correspondences matrix is $O(mn)$ for n and m nodes of the two models, respectively, where $n > m$, without loss of generality. The implementation solves the assignment problem, whose processing costs $O(n^2m)$ [37, 38]. The complexity of the computation of the clusters (and initial SESE regions) is $O(e)$, where e refers to the number of edges of a graph [36]. Graphs of business process models are typically weakly connected, so the number of edges in a graph with n nodes should not exceed n^2.

Summarising, the worst case computational complexity of our approach is polynomial and depends on the complexity of solving the assignment problem, which is a requirement for the computation of node mappings.

Concerning limitations, process behaviour has, so far, not explicitly been considered. We include structural investigations that, e.g., demand the same order of activities in clusters and distinguish cluster types. However, gateway semantics, for example, are not considered so far, although their inclusion is thinkable. On word level, we experience common NLP challenges such as stopwords as "not". Further, often words refer to meanings that are too specific such that they are not covered by word *lexica* (WordNet or Wikipedia). For process models including many gateways compared to the node count, the determined related cluster pairs tend to be of small sizes (scattered node assignments), which affects the results of the process similarity measure. In this case, the process similarity considerations for the retrieval scenario are reduced to pure node label similarity, neglecting parts of structural information. As countermeasure, the weighting function $q(m)$ can be set to an extended linear (e.g., $q(m) = cm + d$) or polynomial variant (e.g., $q(m) = cm^d$), in order to increase the emphasis on clusters of size 2 (assuming there are very few clusters that are larger).

5. Process Model Comparison

In the first scenario, we assess the approach's quality of identifying (correct) node assignments.

5.1. Evaluation Setup

We evaluated the approach using the 604 processes from the SAP reference model. We randomly selected 48 processes from the collection. For the evaluation, we modified each of them, forming 48 pairs of process models. We applied the following modification types:

(LC) Node labels of the original process model were changed such that, to a person (non-expert), the meaning remains the same. This variation is intended to challenge the approach's label matching ability.

(LS) Additionally to LC, nodes have been randomly deleted or new ones have been inserted. Nodes are reordered by randomly swapping two functions and/or two events, respectively.

We perform these modifications in three modification intensities $r1$, $r3$, and $r6$. While in $r1$, 10% of the nodes (not considering gateway nodes) of a process model are subject to modification, in $r3$ and $r6$ we change 30% and 60%, respectively. This results in 6 evaluation classes. In $LCr3$, for example, 30% of the nodes' labels are modified.

In order to produce unbiased results, we perform a 3-fold cross-validation for each metric. Each fold contains 16 process pairs. The training set consists of 32 process pairs (2 folds), and we test the learned configuration on the remaining test set of 16. As baseline for our consideration, we use the Levenshtein distance (Eq. 1).

To assess the quality of the matching approach, we make use of *match accuracy* for automated matching tasks [9]. This metric evaluates the quality of matching algorithms that require human quality assessment. The metric is based on the effort a user needs to convert an automatically created matching result $S = \{(s_1, t_1), ...(s_n, t_n)\}$ suggested by the matching algorithm into the correct result $P = \{(p_1, r_1), ...(p_m, r_m)\}$, where s, t, r, p are nodes, and n the number of predicted matches and m the number of right matches. The required effort is measured in terms of adjustment operations (additions and deletions) proceeding from S. The match accuracy is defined as [9]:

$$\text{Acc}_{\text{Match}} = 1 - \frac{(n-c)+(m-c)}{m} = \frac{c}{m}\left(2 - \frac{n}{c}\right), \qquad (14)$$

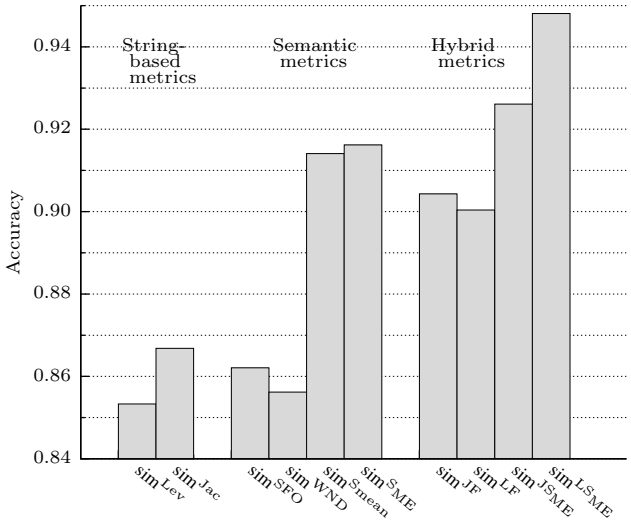


Figure 4: Cross-validation results (accuracy per metric)

where $c = |S \cap P|$ is the number of correct suggestions. $(n-c)$ is the number of false positives (to be deleted from S), and $(m-c)$ the number of false negatives (to be added to S). If the matching process would be performed manually, m add operations would be required.

In this context, the *recall* of a metric ($\frac{c}{m}$) is the number of correctly predicted matches by the number of all matches. *Precision* ($\frac{c}{n}$) corresponds to the amount of correctly suggested matches in the set of predicted matches. A way to consider both qualities is the F_1-measure, calculated as the harmonic mean of precision (Prec) and recall (Rec) [27]: $F_1 = \frac{2 \cdot Prec \cdot Rec}{Prec + Rec}$.

5.2. Evaluation Results

We performed a cross-validation evaluation on the described test data for each of the following single metrics: the Levenshtein string-edit distance sim^{Lev}, the Jaccard coefficient sim^{Jac}, the synonym measures $\text{sim}^{\text{S}_{\text{mean}}}$ and $\text{sim}^{\text{S}_{\text{ME}}}$, the WordNet word distance sim^{Dist}, and the semantic first order measure sim^{SFO} based on Wikipedia.

The results show that the baseline, the Levenshtein distance at 85.33% accuracy, is outperformed by all of the other metrics (cf. Figure 4 and Table 3). Among the string-based metrics, the Jaccard coefficient sim^{Jac} performs best with an accuracy of 87%. The WordNet distance does not perform much better than the baseline. In contrast, the synonym functions based on WordNet reveal clearly better results than the string-based metrics. Best performing among the semantic metrics, and far better than the Jaccard metric, is the synonym measure $\text{sim}^{\text{S}_{\text{ME}}}$ using the synonym constant 0.5 and the Monge Elkan metric as aggregating metric (92% accuracy). The other synonym measure $\text{sim}^{\text{S}_{\text{mean}}}$ follows closely at 91% accuracy. However, it is not as stable as $\text{sim}^{\text{S}_{\text{ME}}}$ concerning the similarity threshold.

In order to achieve further improvements, we combine string-based and semantic metrics, that performed best in these experiments. Accordingly, we created the following hybrid metrics using the weighted combination $\oplus$ specified in Eq. 11: $\text{sim}^{\text{JS}_{\text{ME}}}$, sim^{JF}, $\text{sim}^{\text{LS}_{\text{ME}}}$, and sim^{LF} (cf. also Table 3).

Clearly, the metric $\text{sim}^{\text{LS}_{\text{ME}}}$ outperforms all other metrics at an accuracy value greater than 94% on unseen data as average of three folds. Second ranks $\text{sim}^{\text{JS}_{\text{ME}}}$ (92.61%). As part of the results of $\text{sim}^{\text{LS}_{\text{ME}}}$, concerning the modification type *LS*, the average accuracy (of the 24 contained pairs) is at 93%, while for *LC* it equals to 97%. For the modification classes *LSr*1, *LSr*3, and *LSr*6, the average accuracy values are at 97%, 95%, and 87%, respectively. With increasing modification intensity, the accuracy decreases. However, increasing the count of changed nodes by 50% only results in about 10% performance loss.

The threshold and weight values for the string-based part (w_{Str}) that were learned for each metric during the evaluation are outlined in Table 3. Apart from the mean values (across folds), the standard deviations indicate the stability of the metric concerning its parameters. For half of the single metrics, the threshold and the weight have been determined unambiguously (standard deviation of 0.0). In particular, all metrics that performed best ($\text{sim}^{\text{JS}_{\text{ME}}}$, $\text{sim}^{\text{LS}_{\text{ME}}}$, $\text{sim}^{\text{S}_{\text{ME}}}$), excel concerning parameter stability.

Concluding, the evaluation shows that the metrics provide promising results concerning work assisstance compared to manual comparison. The accuracy values of the best metrics are clearly beyond 90%, the F1-measure clearly greater than 95%. Our hybrid metrics clearly improve the results achieved by the respective single metrics. The top metrics ($\text{sim}^{\text{JS}_{\text{ME}}}$ and $\text{sim}^{\text{LS}_{\text{ME}}}$) provide reliable results, independent on their training data. Further, the hybrid metrics we specified do not only reveal the best results, but also are the most stable metrics on unseen data.

6. Process Model Retrieval

In the second scenario, we apply the related cluster pair approach on retrieving similar process models given a query model. For the selection of the metrics, we use information concerning performance and parametrisation learned during the first scenario's evaluation.

6.1. Evaluation Setup

We performed the evaluation using the same set of 604 process models as described above. In particular, we used the same test case as used in [5], consisting of 100 randomly selected process models that form the process repository and 10 query models (taken out of the 100). Pairwise, two query models have been changed according to 5 modification types. Two models (1+2) have not been changed at all. Models 3 and 4 have been subject to label change. Labels are changed without changing the meaning for a human reader, challenging the label matching ability. Two further models have been reduced to approximately 50%-subgraphs of their own (5+6) as a structural variation. In two models (7+8), the connectors of the original model were randomly changed to different connectors, affecting at most the model's behaviour. The last two models have been modified by randomly swapping of function pairs and event pairs (9+10) to create structural and behavioural changes.

Table 3: Cross validation results for process model comparison

Metric	Id.	Eq.	*Accuracy [%]*	F_1 *[%]*	*Mean values for Threshold*	*Weight w_{Str}*
Single string-based metrics						
Levenshtein distance	sim^{Lev}	(1)	85.33	92.93	0.10 (±0.0000)	–
Jaccard coefficient	sim^{Jac}	(2)	86.68	93.23	0.10 (±0.0000)	–
Single semantic metrics						
Semantic first order	sim^{SFO}	(10)	86.21	93.35	0.02 (±0.0125)	–
WordNet distance	sim^{WND}	(8)	85.62	93.03	0.05 (±0.0471)	–
Synonym (ME)	$sim^{S_{ME}}$	(6)	91.62	95.88	0.21 (±0.0000)	–
Synonym (mean)	$sim^{S_{mean}}$	(4)	91.41	95.76	0.25 (±0.0613)	–
Combined metrics						
Levenshtein distance/ Synonym (ME)	$sim^{LS_{ME}}$	(1) ⊕ (6)	94.81	97.48	0.18 (±0.0000)	0.10 (±0.0000)
Jaccard coefficient/ Synonym (ME)	$sim^{JS_{ME}}$	(2) ⊕ (6)	92.61	96.41	0.05 (±0.0000)	0.40 (±0.0000)
Levenshtein distance/ Semantic first order	sim^{LF}	(1) ⊕ (10)	90.04	95.19	0.16 (±0.0283)	0.23 (±0.0943)
Jaccard coefficient/ Semantic first order	sim^{JF}	(2) ⊕ (10)	90.43	95.38	0.02 (±0.0047)	0.43 (±0.0624)

Standard deviation provided in parentheses; w_{Str} indicates the weight of the string-based part measure

The models contained in the collection of 604 process models do not only specify disjunct procedures; in fact, some describe similar activity flows. For each of the 10 query models, all 100 process models of the test case have been annotated by process model experts. This way, a list of relevant models per query has been determined.

As a baseline for our comparison, we consider the results of the Indri text search engine (cf. [5]). As second search engine, we use the well-known and widely used Apache Lucene engine[5]. For the text search, the node labels of all 110 process models have been extracted into text files. Further, we compare the results to the results of the *Label Similarity* metric by Dijkman et al. [5].

For each metric, we perform one search experiment per query model. We measure the performance of these experiments in terms of *average precision*, *R-Precision*, as well as *first-n-Precision* for $n = 5$ and $n = 10$. The average precision characterises the quality of a single search query. For one search query, it equals the *mean average precision (MAP)*. For their calculation we use the following metric. Q is the set of searches that are performed. For each query $q_j \in Q$, the set of relevant documents is defined as $\{d_1, ..., d_{m_j}\}$, and R_{jk} as the ranked list of retrieval results from the best result until and including document d_k. MAP is defined as follows [27]:

$$\text{MAP}(Q) = \frac{1}{|Q|} \sum_{j=1}^{|Q|} \frac{1}{m_j} \sum_{k=1}^{m_j} \text{Precision}(R_{jk}) \qquad (15)$$

R-Precision is the precision of a query calculated after the first R documents have been found, where R is the number of relevant documents for this query. So R varies with the query [27]. The first-n-Precision provides the query precision after a cut-off after the n^{th} predicted document.

During the evaluation, we investigate two metrics: $sim^{LS_{ME}}$ and $sim^{JS_{ME}}$. They are set up (concerning weight and threshold) as indicated in Table 3. We use the node-based process model similarity $sim^{Node}_{PM}(QM_i, MP_j, sim^{S_{ME}})$, where QM_i refers to the 10 query models, and MP_j represents all 100 models of the process model pool (cf. Eq. 13) . $sim^{JS_{ME}}$ is used analogously.

6.2. Evaluation Results

The results of the retrieval evaluation are shown in Figure 5 and Table 4. Generally, concerning the results shown, we outlined only the best performing metrics. In 90% of the queries, our result clearly outperforms the baseline as well as the Label Similarity approach by Dijkman et al. [5].

For process model 3, $sim^{LS_{ME}}$ performs better than the label similarity approach. Both search engines, however, perform better. In case of query 4, the performance of $sim^{LS_{ME}}$ is slightly worse than label matching. However, a definite improvement concerning Indri is obvious. In case of query 4 and 5, Lucene outperforms all other specialised approaches. For all remaining queries, our approaches represent a clear improvement compared to all both text search approaches and related work.

Referring to the mean average precision values (cf. Table 4), our approaches outperform all others. Regarding the first-10-precision, the approaches perform comparably, showing in average 79% of the relevant documents in the first 10 results. In terms of the first-5-precision, our approach $sim^{LS_{ME}}$ outperforms Lucene Text Search. However, if the number of relevant documents is low, say 5, the *first-n-precision* metric (at

Table 4: Overall results of process retrieval metrics

Metric	*MAP*	*Mean R-precision*	*First-10-precision*	*First-5-precision*
$sim^{LS_{ME}}$ (node-based)	0.8609	0.7774	0.7900	0.9600
$sim^{JS_{ME}}$ (node-based)	0.8591	0.7708	0.7900	0.9400
Label Similarity [5]	0.8000	–	0.7900	–
Lucene Search Engine	0.8211	0.7491	0.6800	0.9400
Indri Text Search [5]	0.7611	–	0.7000	–
Structural Similarity [5]	0.8300	–	0.7800	–
Behavioural Sim. [5]	0.8000	–	0.7400	–

[5] `http://lucene.apache.org/`

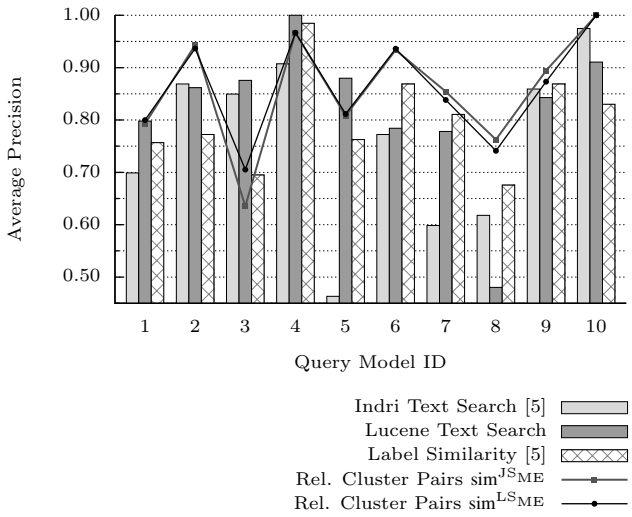


Figure 5: Average precision per query model

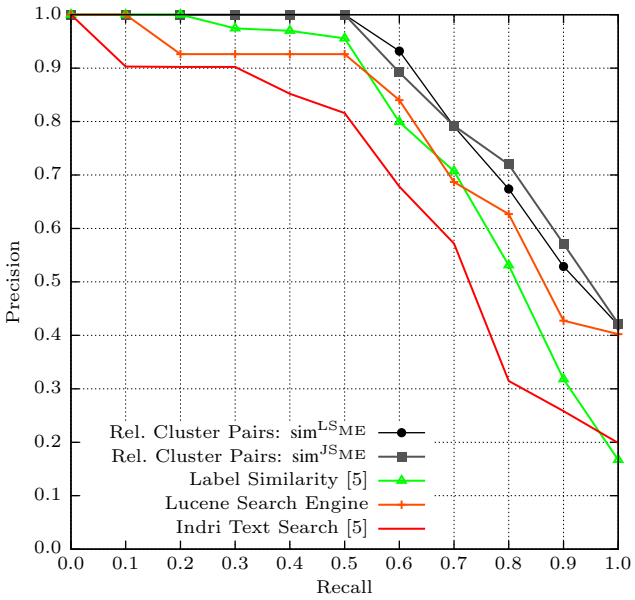


Figure 6: Process model retrieval: Precision-recall curve

$n = 10$) is limited to indicate low precision values[6] – in that case to a maximum of $5/10 = 0.5$. This is not the case for the R-precision measure. The results show that, at a point where all relevant process models could have been found for one query, more than two third of the models have been identified by our approaches in average, which is a promising result.

We summarise the overall results of this evaluation in an interpolated precision-recall curve ([27], cf. Figure 6).[7] The values for the Indri search engine, as well as for the Label Similarity have been taken from [5]. Overall, the metric $\text{sim}^{\text{LS}_{\text{ME}}}$, as combination of the Levenshtein distance and the synonym metric $\text{sim}^{\text{S}_{\text{ME}}}$ performs best. In particular, at a recall level of 0.5 (i.e., having reported 50% of the relevant process models as part of the result list) 100% of the listed models are relevant. After having reported more than the half of relevant documents (at 0.6 recall), the precision for $\text{sim}^{\text{LS}_{\text{ME}}}$ is still clearly beyond 90%. For a search engine, this indicates a very good result.

Concluding, our approach outperforms the current related work as well as established search engines. Concerning the Indri search engine, this shows that the additional computation complexity of more sophisticated search approaches is rewarded.

Concerning related work, we assume that word preprocessing is a major reason for the improvement. Lemmatising has, to the best of our knowledge, not been utilised in related approaches. In preliminary experiments we found, that using stemming did not improve results in all cases. Especially for semantic measures, improvement by stemming mostly fails, as dictionaries mostly cannot cope with stemmed strings. This is why we did not employ Porter's stemming algorithm for the metrics $\text{sim}^{\text{LS}_{\text{ME}}}$ and $\text{sim}^{\text{JS}_{\text{ME}}}$. Further, we carefully revised the stop word list. In traditional stop word lists, "needs", "omitted", or "part" and similar ones are considered function words. However, for processing of operational process models, we found that many of these can have high impact on a label's overall meaning, and in particular improve results, especially when using semantic similarity. In fact, we used lemmatising for all metrics. For semantic similarity metrics, reducing labels and words to a meaningful form is an unconditional technique for preserving information and meaning. It ensures reasonable lookups and obviously improves the overall performance.

As part of future work, the label meanings per node can be investigated in more depth. Using current *community mining* approaches from NLP, for example, the basic notion (positive vs. negative) of a node's meaning can be analysed and made processable to further improve the matching process.

6.3. Discussion

Generally, as the selected test case is completely independent from the work at hand, we meet the requirement of "fair testing" as we do not make use of an artificially created test case that meets the requirements of the presented approach [39]. Further, to the best of our knowledge, in this paper, we provide one of the few quantitative evaluations in this area whose results can be directly compared to other approaches as the test case is available. However, even though the evaluation approach and test case used in this paper were suitable to perform the evaluation of the related cluster pair approach, further evaluations could be performed to assess its performance in other contexts. In particular, the approach could be evaluated on a test case other than the SAP reference model, i.e., a test case consisting of more heterogeneous models – process models covered by the used reference model are aligned concerning terminology and modelling style.

[6] The overall minimum number of relevant process models per query in this test case is 5 (for query model 4).

[7] The x-axis represents the recall intervals [0.00, 0.05), [0.05, 0.10), ..., [0.95, 1.00].

Concerning taxonomy, our approach covers techniques that can be used to tackle different modelling taxonomies. As considerations of semantic node similarity, apart from the best performing synonym measures, we investigate a variety of semantic word relations (from used language via Wikipedia and from a linguistic viewpoint via WordNet) and apply them together with a well-approved and established word reduction technique from the NLP domain, lemmatising. It avoids loss of word meaning and yields results that can always be further processed (in contrast to, e.g., stemming).

The utilised string-based measures are well-known reliable measures that achieve good results in a variety of application areas. As we could show, hybrid measures (i.e., efficient combination of string-based and semantic measures) clearly yield improvements compared to the respective single measures' results. From our viewpoint, an impairment of their results when applied to heterogeneous process models is improbable.

Concerning modelling style, as mentioned before, in case of scattered node assignments, our measure of process model similarity is reduced to a function of node similarities, which still yields good results (cf. Section 5.2). In this case, the weighting function can be adjusted (as outlined in Section 4.5). As our approach does primarily target the determination of correct node assignments, we consider changes in modelling terminology to have a higher impact on the results than different modelling styles.

Summarising, we use a variety of techniques to address different modelling terminologies. Given these details of our approach, the achievement of comparable or better results is thinkable. Unfortunately, appropriate test cases (especially for the heterogeneous case) are hardly available in the research community so far.

7. Conclusion

The problem of determination of correct node assignments represents a fundamental requirement for fully automated process comparison and retrieval approaches that consider, e.g., structural or behavioural characteristics. In our approach, we target the improvement of solutions to this problem. Using related cluster pairs, we apply both string-based similarity and semantic similarity metrics and create hybrid measures from these, and combine these information with structural process characteristics. The application of this concept in two scenarios, process model comparison and process model retrieval, has been evaluated using the SAP reference model.

In the first scenario, we focused on identifying the delta of two process models. Based on the calculation of related cluster pairs, we set up a metric to identify changes made to particular process models. We performed a cross validation evaluation for a number of semantic, string-based, and hybrid metrics and learned the metric threshold as well as the weighting in case of hybrid metrics. As results, most of the metrics show accuracy values of beyond 90%, which refers to an according work assistance compared to manual work. The best hybrid measure performed at 94.8% accuracy and an F_1 score of 97.5% on unseen data. Further, the best performing metrics are stable concerning all their parameters. Generally, these are semantic measures and in particular, a synonym measure based on WordNet. As these results were achieved based on a test case consisting of activities from practice, the practical applicability of this approach is emphasised. In a further evaluation, the best performing measures have been tested in the context of process model retrieval on a second test case based on the SAP reference model. In particular, we tested a novel similarity notion for process models based on related cluster pairs. We compared our results with the results of two established text search engines as well as related work. At a mean average precision of 86%, our approach, parameterised with one of our hybrid measures, clearly outperforms text search engines as well as related work.

In our experiments, we generally considered a particular word preprocessing technique. Additionally to stop word removal, we reduced words to their base forms, preserving their word meaning, which thus can still be looked up by consultation of word *lexica*. Further, the concept of related cluster pairs combines node similarity with structural process characteristics. We assume that the combination of these techniques contributed to the realised improvement.

Even though the evaluation approach and test case used in this paper were suitable to perform the quantitative evaluation of the process similarity measure based on related cluster pairs, further evaluations could be performed to assess its performance in other process model contexts. The base test case, the SAP reference model, is not stringently representative for typical process models, although they are applicable for a quality assessment of similarity measures. For a test case of heterogeneous models, however, our approach seems well equipped with semantic word processing techniques.

As future work, the related cluster pairs concept can be further enhanced. Currently, the approach does not explicitly process behaviour or process structure. In particular, the consideration of gateway semantics could be included. However, the improvements yielded in this approach in the area of node assignments can be used as basis for other approaches focussing on behavioural and structural process similarity.

References

[1] M. Hammer, J. Champy, Reengineering the Corporation: A Manifesto for Business Revolution, HarperBusiness, New York, NY, USA, 2003.

[2] J. Becker, M. Rosemann, M. Kugeler, Process Management. A Guide for the Design of Business Processes, Springer-Verlag New York, Inc., 2003.

[3] C. Raduescu, H.-M. Tan, M. Jayaganesh, W. Bandara, M. zur Muehlen, S. Lippe, A Framework of Issues in Large Process Modeling Projects, in: Proceedings of the European Conference on Information Systems (ECIS 2006), Goteborg, Sweden, 1594–1605, 2006.

[4] H. A. Reijers, R. S. Mans, R. A. van der Toorn, Improved Model Management with Aggregated Business Process Models, Data Knowledge Engineering 68 (2009) 221–243.

[5] R. Dijkman, M. Dumas, B. van Dongen, R. Käärik, J. Mendling, Similarity of Business Process Models: Metrics and Evaluation, Information Systems 36 (2) (2011) 498–516.

[6] B. van Dongen, R. Dijkman, J. Mendling, Measuring Similarity between Business Process Models, in: Proceedings of the 20th International Conference on Advanced Information Systems Engineering (CAiSE 2008), Montpellier, France, 450–464, 2008.

[7] J. M. Küster, C. Gerth, A. Förster, G. Engels, Detecting and Resolving Process Model Differences in the Absence of a Change Log, in: Proceedings of the Sixth International Conference on Business Process Management (BPM 2008), Milan, Italy, 244–260, 2008.

[8] R. Dijkman, M. Dumas, L. García-Bañuelos, Graph Matching Algorithms for Business Process Model Similarity Search, in: Proceedings of the Seventh International Conference on Business Process Management (BPM 2009), vol. 5701 of *Lecture Notes in Computer Science*, Springer-Verlag, Berlin Heidelberg, 48–63, 2009.

[9] S. Melnik, H. Garcia-Molina, E. Rahm, Similarity Flooding: A Versatile Graph Matching Algorithm and Its Application to Schema Matching, in: Proceedings of the 18th International Conference on Data Engineering (ICDE 2002), San Jose, CA, USA, 117–128, 2002.

[10] M. Ehrig, A. Koschmider, A. Oberweis, Measuring Similarity between Semantic Business Process Models, in: Proceedings of the Fourth Asia-Pacific Conference on Conceptual Modelling (APCCM 2007), Ballarat, Victoria, Australia, 71–80, 2007.

[11] W. van der Aalst, A. K. A. de Medeiros, A. Weijters, Process Equivalence: Comparing Two Process Models Based on Observed Behavior, in: Business Process Management, vol. 4102 of *Lecture Notes in Computer Science*, chap. 10, Springer-Verlag, Berlin Heidelberg, 129–144, 2006.

[12] C. Li, M. Reichert, A. Wombacher, Discovering Reference Process Models by Mining Process Variants, in: Proceedings of the International Conference on Web Services (ICWS 2008), Beijing, China, 45–53, 2008.

[13] K. Andrews, M. Wohlfahrt, G. Wurzinger, Visual Graph Comparison, in: Proceedings of the 13th International Conference on Information Visualisation (IV 2009), Washington, DC, USA, 62–67, 2009.

[14] R. Dijkman, Diagnosing Differences between Business Process Models, in: Proceedings of the Sixth International Conference on Business Process Management (BPM 2008), vol. 5240 of *Lecture Notes in Computer Science*, Springer-Verlag, Berlin Heidelberg, 261–277, 2008.

[15] R. Dijkman, A Classification of Differences between Similar Business Processes, in: Proceedings of the 11th International Enterprise Distributed Object Computing Conference (EDOC 2007), Annapolis, MD, USA, 37–50, 2007.

[16] A. Koschmider, A. Oberweis, Ontology based Business Process Description, in: Proceedings of the Conference on Advanced Information Systems Engineering (CAiSE 2005) - Workshops, 2, Porto, Portugal, 321–333, 2005.

[17] C. Li, M. Reichert, A. Wombacher, Discovering Reference Models by Mining Process Variants Using a Heuristic Approach, in: Proceedings of the Seventh International Conference on Business Process Management (BPM 2009), vol. 5701 of *Lecture Notes in Computer Science*, Springer-Verlag, Berlin Heidelberg, 344 – 362, 2009.

[18] R. Lu, S. W. Sadiq, On the Discovery of Preferred Work Practice Through Business Process Variants, in: Proceedings of the 26th International Conference on Conceptual Modeling, vol. 4801 of *Lecture Notes in Computer Science*, Springer-Verlag, Berlin Heidelberg, 165–180, 2007.

[19] T. Madhusudan, J. L. Zhao, B. Marshall, A Case-based Reasoning Framework for Workflow Model Management, Data Knowledge Engineering 50 (2004) 87–115.

[20] M. Minor, A. Tartakovski, R. Bergmann, Representation and Structure-Based Similarity Assessment for Agile Workflows, in: Proceedings of the Seventh International Conference on Case-Based Reasoning: Case-Based Reasoning Research and Development, vol. 4626 of *Lecture Notes in Computer Science*, Springer-Verlag, Berlin Heidelberg, 224–238, 2007.

[21] S. Nejati, M. Sabetzadeh, M. Chechik, S. M. Easterbrook, P. Zave, Matching and Merging of Statecharts Specifications, in: 29th International Conference on Software Engineering (ICSE 2007), 54–64, 2007.

[22] A. Wombacher, Evaluation of Technical Measures for Workflow Similarity Based on a Pilot Study, in: 14th International Conference on Cooperative Information Systems, vol. 4275 of *Lecture Notes in Computer Science*, Springer, 255–272, 2006.

[23] J. Mendling, J. Recker, H. A. Reijers, On the Usage of Labels and Icons in Business Process Modeling, International Journal of System Modeling and Design 1 (2) (2010) 40–58.

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

[25] H. Schmid, Probabilistic Part-of-Speech Tagging Using Decision Trees, in: Proceedings of the International Conference on New Methods in Language Processing, Manchester, UK, 1994.

[26] V. I. Levenshtein, Binary code capable of correcting deletions, insertions and reversals, Cybernetics and Control Theory (Soviet Physics Doklady) 10 (8) (1966) 707–710.

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

[28] W. W. Cohen, P. Ravikumar, S. E. Fienberg, A Comparison of String Distance Metrics for Name-Matching Tasks, in: Proceedings of the International Joint Conference on Artificial Intelligence – Workshop on Information Integration, Acapulco, Mexico, 73–78, 2003.

[29] C. Fellbaum (Ed.), WordNet: An Electronic Lexical Database (Language, Speech, and Communication), MIT Press, Cambridge, MA, USA, 1998.

[30] D. Lin, Automatic Retrieval and Clustering of Similar Words, in: Proceedings of the 17th International Conference on Computational Linguistics, Montreal, Quebec, Canada, 768–774, 1998.

[31] A. Monge, C. Elkan, The Field Matching Problem: Algorithms and Applications, in: Proceedings of the Secon International Conference on Knowledge Discovery and Data Mining, 267–270, 1996.

[32] M. Niemann, M. Siebenhaar, J. Eckert, R. Steinmetz, Process Model Analysis using Related Cluster Pairs, in: Business Process Management Workshops: BPM 2010 International Workshops and Education Track, vol. 66 of *Lecture Notes in Business Information Processing*, Springer-Verlag, Berlin Heidelberg, 547–558, 2011.

[33] F. Bourgeois, J.-C. Lassalle, An Extension of the Munkres Algorithm for the Assignment Problem to Rectangular Matrices, Communications of the ACM 14 (12) (1971) 802–804.

[34] F. Bourgeois, J.-C. Lassalle, Algorithm 415: Algorithm for the Assignment Problem (Rectangular Matrices), Communications of the ACM 14 (12) (1971) 805–806.

[35] R. Johnson, D. Pearson, K. Pingali, Finding Regions Fast: Single Entry Single Exit and Control Regions in Linear Time, Technical Report, Cornell University, Ithaca, NY, USA, 1993.

[36] R. Johnson, D. Pearson, K. Pingali, The Program Structure Tree: Computing Control Regions in Linear Time, in: Proceedings of the ACM SIGPLAN 1994 Conference on Programming Language Design and Implementation, Orlando, FL, USA, 171–185, 1994.

[37] R. Burkard, M. Dell'Amico, S. Martello, Assignment Problems, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2009.

[38] D. Alevras, Assignment and Matching, in: C. A. Floudas, P. M. Pardalos (Eds.), Encyclopedia of Optimization, Springer-Verlag New York, Inc., 106–108, 2009.

[39] J. Zobel, Writing for Computer Science, Springer-Verlag, New York, 2004.