Detecting Inconsistencies between Process Models and Textual Descriptions

Han van der Aa[1], Henrik Leopold[1], Hajo A. Reijers[1,2]

[1] Department of Computer Sciences, Vrije Universiteit Amsterdam, Faculty of Sciences, De Boelelaan 1081, 1081HV Amsterdam, The Netherlands
[2] Department of Mathematics and Computer Science, Eindhoven University of Technology, PO Box 513, 5600MB Eindhoven, The Netherlands

Abstract. Text-based and model-based process descriptions have their own particular strengths and, as such, appeal to different stakeholders. For this reason, it is not unusual to find within an organization descriptions of the same business processes in both modes. When considering that hundreds of such descriptions may be in use in a particular organization by dozens of people, using a variety of editors, there is a clear risk that such models become misaligned. To reduce the time and effort needed to repair such situations, this paper presents the first approach to automatically identify inconsistencies between a process model and a corresponding textual description. Our approach leverages natural language processing techniques to identify cases where the two process representations describe activities in different orders, as well as model activities that are missing from the textual description. A quantitative evaluation with 46 real-life model-text pairs demonstrates that our approach allows users to quickly and effectively identify those descriptions in a process repository that are inconsistent.

1 Introduction

Organizations use business process models for documenting and improving business operations as well as for the specification of requirements for information systems [18]. As a result, many companies maintain huge process model repositories, often including several hundred or even thousand models [32]. While process models turn out to be useful artifacts in numerous contexts, many organizations are also aware of their limitations. One major challenge is that process models are not intuitive to every employee. Particularly business professionals, those who actually conduct the various process tasks, often do not feel confident in reading and interpreting process models [7,11]. For this reason, the value of maintaining text-based business process descriptions alongside model-based ones has been recognized [24]. A textual description uses natural language to outline the steps of a business process. While such a description may not be suitable to exactly represent all complex aspects of a process [3], it has the advantage that it can be understood by virtually everyone. Companies can thus ensure that information about their processes is widely accessible by using textual descriptions next to using process models for analytical and technical purposes [1].

Despite its merits, the existence of multiple representation formats describing the same process can lead to considerable problems as well. When a text and a process model both describe the same business process, it is crucial to prevent inconsistencies in terms of contradicting information. Inconsistencies occur in particular when documents are being developed or maintained independently from each other [31]. Once conflicts start occurring over time, the effort that is needed to identify and clear up the differences is considerable, even more so when organizations have already built up huge process repositories.

To effectively deal with the problem of inconsistencies between process model and text, we propose in this paper a technique that automatically detects differences between textual and model-based process descriptions. The technique can be used to quickly identify those process models in a collection that are likely to diverge from their accompanying textual descriptions. This allows organizations to focus their efforts on those processes that can be expected to contain such inconsistencies. Focusing on such key processes is crucial for organizations, since few have the resources required to analyze all their processes in detail [11]. Our quantitative evaluation demonstrates that the proposed technique is indeed able to quickly identify the vast majority of problematic processes in a collection of model-text pairs obtained from practice.

The remainder of this paper is structured as follows. Section 2 explains the research problem using an illustrative example. Section 3 discusses related work and identifies the research gap of interest. Section 4 describes the proposed approach for inconsistency detection. In Section 5, we present a quantitative evaluation of the approach. Finally, we conclude the paper and present directions for future research in Section 6.

2 Problem Illustration

To illustrate the challenges that are associated with the endeavor to detect inconsistencies between textual and model-based process descriptions, consider the model-text pair shown in Figure 1. It includes a textual and a model-based description of a bicycle manufacturing process. On the left-hand side, we observe a textual description, which comprises eleven sentences. On the right-hand side, a corresponding model-based description can be seen, expressed in the Business Process Model and Notation (BPMN). The model contains nine activities, which are depicted using boxes with rounded edges. The diamond shapes that contain a plus symbol indicate concurrent streams of action; the diamond shapes containing a cross represent decision points. The gray shades suggest correspondences between the sentences and the activities of the process model.

A closer look at the example reveals that many connections between the two artifacts are evident. For example, there is little doubt that sentence (7) describes the activity "*reserve part*" or that sentence (8) describes the activity "*back-order part*". In some cases, however, there is clearly an inconsistency between the two process representations. For instance, there is no sentence that is related to the activity "*ship bicycle to customer*", i.e. that activity is missing from the

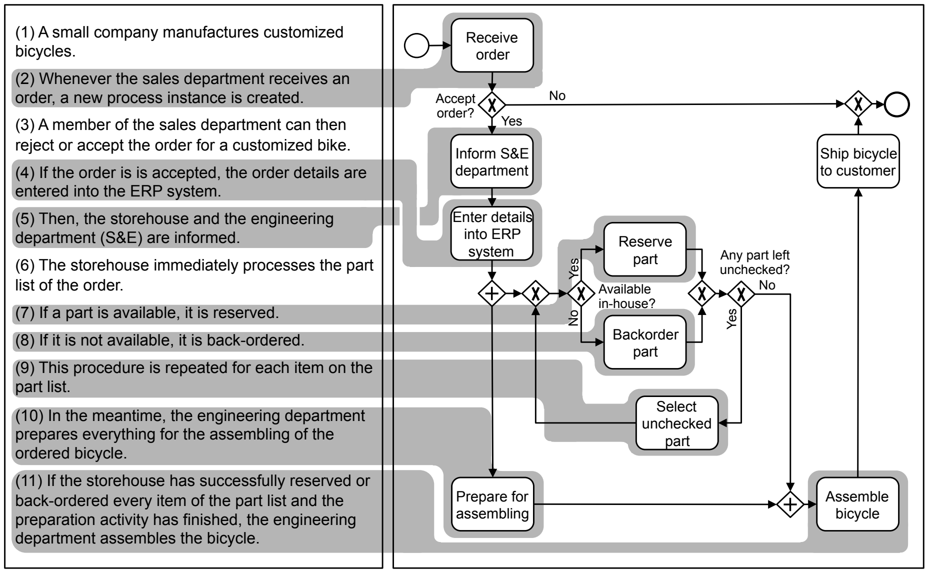

Fig. 1. A textual and a model-based description of a bicycle manufacturing process

textual description. Likewise, we can observe that sentences (4) and (5) occur in a different order than the corresponding activities in the model.

In other cases it is *less* straightforward to decide on the consistency – or lack thereof – between the representations. For example, the text of sentence (9) simply indicates that a part of the process must be repeated. By contrast, the model includes an activity, "*select unchecked part*", which associates an explicit action with this repetition. Whether or not sentence (9) actually describes an activity, and thus should be considered an inconsistency, seems to be open for debate. Ambiguous cases that are already difficult to resolve for human readers pose even greater problems when texts are analyzed in an automatic manner.

The brief illustration of the model-text pair from Figure 1 shows that an appropriate technique for detecting inconsistencies (i) must consider several types of inconsistencies and (ii) must deal with considerable challenges caused by the ambiguous nature of natural language.

3 Related Work

The work presented in this paper relates to two major streams of research: semantic matching and transformations between model and text.

Semantic matching refers to the task of identifying relations between concepts [15]. Particularly in the field of schema and ontology matching it has received considerable attention [10,13,30]. However, in recent years the potential of matching was also recognized in the domain of process modeling [5]. So-called process model matchers are capable of automatically identifying correspondences between the activities of two process models. The application scenarios of these matchers range from harmonization of process model variants [21] to the detection of of process model clones [34]. To accomplish these goals, matchers exploit different process model features, including natural language [12], model structure [9], and behavior [20]. Nevertheless, due to the different nature of our problem, these matchers cannot be applied in a straightforward fashion. Natural language texts neither explicitly provide structural nor behavioral information. Natural language information in texts also differs significantly from what we can find in process model activities. The labels of model activities are shorter than sentences; they also lack the background information and conditional sentences that are provided by natural language texts [23].

The field of transformations between model and text can be further subdivided into two groups. The first group relates to techniques that automatically derive models from natural language text material. Such techniques have been defined for UML class diagrams [4], entity-relationship models [16], and process models [14]. The second group includes techniques that transform a given model into a textual description. Such techniques have been defined for UML diagrams [28], object models [22], and process models [24]. What both groups have in common is that they provide insights on how to move from model to text and vice versa. Among others, they address the problem of inferring structural and behavioral information from textual descriptions. However, to achieve satisfactory results, these techniques require human input. Hence, they are not suitable for supporting the automatic identification of correspondences between a textual and a model-based description.

In summary, we can state that existing techniques do not provide the means to adequately compare textual and model-based process descriptions. In light of this research gap, we define an approach that detects inconsistencies between textual and model-based process descriptions in the subsequent section.

4 Approach

This section describes our approach to identify inconsistent model-text pairs in a process model repository, which consists of various steps. It ultimately provides a quantification of the likelihood that any particular model-text pair contains inconsistencies. Section 4.1 presents an overview of the approach. Sections 4.2 through 4.5 subsequently describe the steps of the approach in detail.

4.1 Overview

As depicted in Figure 2, the first three steps in our approach set out to create an *activity-sentence* correspondence relation between a process model's activities and the sentences of a textual process description. This aligns each process model activity to the sentence that best describes it, if any. To obtain an optimal correspondence relation, we first subject the textual process description and the labels of the activities in the process model to a linguistic analysis. Second, we compute similarity scores between individual activities and sentences, which quantify how well a given sentence describes an activity. Third, we compute an optimal activity-sentence correspondence relation. We do so by complementing the similarity scores with a consideration of the ordering relations that exist between the various process elements. In the fourth and final step, the approach evaluates the quality of the obtained correspondence relation. The quality is here assessed in terms of the similarity between activities and sentences included in the optimal correspondence relation. If this quality is deemed sufficient, we expect that the model-text pair does not contain any inconsistencies. If, however, the correspondence relation has severe quality issues, we predict that the model-text pair contains inconsistencies.

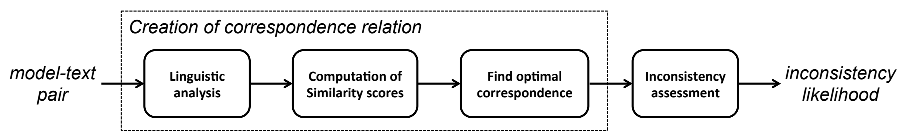

Fig. 2. Outline of the approach

4.2 Linguistic Analysis

In order to create an accurate activity-sentence correspondence for a model-text pair, we first subject the textual process description and the activity labels to a linguistic analysis. In this step we make extensive use of the *Stanford Parser*, a widely employed Natural Language Processing (NLP) tool [27]. It is used to identify base forms of words (i.e. *lemmatization*), and for *part-of-speech* tagging. The latter task assigns a category, i.e. the part of speech, to each word in a text [17]. Common parts of speech include *nouns*, *verbs*, *adjectives*, and *adverbs*.

This step consists of three sub-steps: (i) *anaphora resolution*, (ii) *clause extraction*, and (iii) *text sanitization*. With these three sub-steps, we aim to obtain a representation that accurately reflects the important parts of a sentence, while abstracting from irrelevant details. To illustrate the sub-steps, we consider their impact on sentence (8) from the running example. This sentence is initially represented by the following *bag-of-words*:
{`if, it, is, not, available, it, is, back-ordered`}.

Anaphora Resolution A problem that must be tackled when analyzing natural language texts is the resolution of anaphoric references or *anaphors*. Anaphors are usually pronouns ("*he*", "*her*", "*it*") or determiners ("*this*", "*that*") that refer to a previously introduced unit. These references represent an important challenge in the context of assessing the similarity between an activity and a sentence. Anaphoric references must be properly resolved in order to correctly determine the object that some action refers to. As an example, consider the sentence "*If it is not available, it is back-ordered*". Here, the approach has to identify that "*it*" refers to the word "*part*", which is contained in the preceding sentence. To augment sentences with such important information, we introduce an anaphora resolution technique.

The anaphora resolution technique in our approach sets out to identify the objects contained in a sentence. We identify objects by considering *Stanford Dependencies*, which reflect grammatical relations between words [8]. To identify objects in a sentence, the most important relations to consider include *direct objects* and *nominal subjects*. For instance, in sentence (7) the Stanford Parser identifies the relation *nsubj(reserved, part)*, indicating that the business object "*part*" (acting as the nominal subject in the sentence) is the object being reserved. If all the objects in a sentence are anaphoric references, i.e. the sentence includes only pronouns and determiners, we resolve the references by replacing them with the objects contained in the previous sentence. For sentence (8), this results in: {`if, part, is, not, available, part, is, back-ordered`}.

Relevant Clause Extraction Sentences in a textual description describe actions that are performed in a process, its flow, and additional information. To accurately align process model activities, it is important to identify (parts of) sentences related to actions, while excluding parts unrelated to these actions from consideration. The most problematic cases are conditional sentences, in which the *dependent clause* that specifies a condition, contains terms similar or equal to those used in activity labels. Consider, for example, sentence (11): "*If the storehouse has successfully reserved or back-ordered every item of the part list and the preparation activity has finished* [...]" When considered naively, this sentence has a high term similarity to the activities "*reserve part*" and "*back-order part*". However, it is clear that these activities are actually described elsewhere in the description. By focusing only on the *main clause* of such sentences, we therefore remove potential confusion caused by conditional expressions.

In order to differentiate between conditions and main clauses, we use the parse trees generated by the Stanford Dependency Parser. In these trees, conditional expressions are represented as subordinate clauses (SBAR), starting with a conditional term, e.g. "*if*", "*in case*", or "*once*". The parse tree for sentence (8) is shown in Figure 3. By extracting the main clause from this sentence, the following bag-of-words remains: {`part, is, back-ordered`}.

Text Sanitization The final linguistic analysis sub-step involves text sanitization on both (previously processed) sentences and activity labels. Text sanitization sets out to create a similar and comparable representation of activity labels and sentences, and of their individual terms. Sanitization comprises the removal of stop words and word lemmatization.

First, we remove all *stop words* from each activity label and sentence. Stop words are common words that are of little value when considering similarity between texts (i.e. labels and sentences) [26]. We remove *closed class* determiners, prepositions, and conjunctions (e.g. "*the*", "*in*", "*to*", "*for*") from the activity

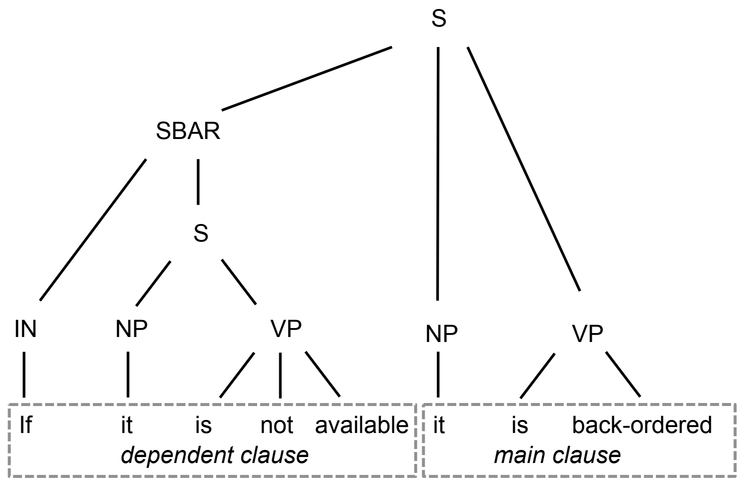

Fig. 3. Simplified parse tree for sentence (8).

labels and sentences. This procedure is in line with many approaches from the domain of process model matching (see e.g. [6,19,36]). Second, we lemmatize the remaining words using the Stanford Parser. The resulting *lemmas* represent grammatical base forms of words. By considering lemmas, it is straightforward to determine whether two words have a similar root. E.g. "*sing*", "*sang*", and "*sung*" are all mapped to the common lemma "*sing*" [17].

Text sanitization concludes the linguistic analysis. For sentence (8), this results in the final bag-of-words representation: {`part, be, back-order`}. The next step takes the processed activity labels and sentences as input, to determine their similarity.

4.3 Computation of Similarity Scores

The ability to judge the similarity between a sentence and an activity is critical to the performance of our approach. A sentence and an activity are considered to be similar if they refer to the same stream of action. To accurately judge this, the variability of natural language expressions contained in the sentences should be taken into account [2]. To deal with this variability, we select a semantic measure to assess the similarity of a sentence to an activity. Specifically, we use a semantic similarity measure proposed by Mihalcea et al. [29] because it combines *word semantic similarity* with *word specificity* scores. The similarity between an activity a and a sentence s is formalized in Equation 1.

$$sim(a,s) = \frac{1}{2}\left(\frac{\sum_{t \in \{a\}} maxSim(t,s) \times idf(t)}{\sum_{t \in \{s\}} idf(t)} + \frac{\sum_{t \in \{s\}} maxSim(t,a) \times idf(t)}{\sum_{t \in \{s\}} idf(t)}\right) \quad (1)$$

Here, $maxSim(t_1, s)$ denotes the maximum semantic similarity between a term t_1 and any term t_2 contained in s.

$$maxSim(t_1, s) = \max\{Lin(t_1, t_2) \mid t_2 \in s\} \tag{2}$$

To compute the semantic similarity $Lin(t_1, t_2)$ between two terms, we employ a WordNet-based implementation of the similarity measure defined by Lin[3]. It is a measure from the domain of information theory, which has been widely adopted for computing semantic similarity. What is more, it has been shown to correlate well with human judgments [25].

To determine the similarity between a sentence and an activity, it is not only important to consider the similarity between individual terms. The relative importance of words or *word specificity* also plays an important role. Common terms have little discriminating power in determining similarity, while more unique terms represent important similarity indicators. For this reason, Equation 1 incorporates the *Inverse Document frequency* (idf) of terms. The idf assigns a low score to terms that occur in a large number of activity labels or sentences and, therefore, have lower discriminating power. The idf for a term t is given by Equation 3, where document collection D comprises all activity labels and sentences.

$$idf(t, D) = \log \frac{|D|}{|d \in D : t \in d|} \tag{3}$$

The similarity between a sentence and an activity plays an important role in the creation of a correspondence relation between a process model and a textual description. To further improve the results, our approach also considers the order in which activities and sentences appear, as detailed in the next section.

4.4 Optimal Correspondence Relation

This section describes how we obtain an optimal correspondence relation between activity set A and sentence set S. To achieve this, we not only consider the similarity of activities and sentences, but also the order in which activities are described the textual description and contained in the process model. We refer to a correspondence relation that respects these orders as *coherent*.

Textual process descriptions generally describe process steps in a chronological order [33]. That means that if activity a precedes activity b in a process, the text describes activity a prior to b. For a process model, these relations are explicitly captured in a *partial order* relation $\leq$. The relation $\leq$ defines for each activity which other activities precede and succeed it. Such an order is only partial (as opposed to a strict order), because processes may contain *alternative* and *concurrent* execution paths. For instance, the process of Figure 1 executes either of the alternative activities "*reserve part*" and "*back-order part*", depending on the availability of a given part. To construct a partial order, we employ

[3] https://code.google.com/p/ws4j/

the behavioral profile computation as defined in [37]. A correspondence relation C between an activity set A and a sentence set S is considered to be coherent if it adheres to the following constraint: Given process model activities a and b, and the sentences s_a and s_b to which they are, respectively, aligned. If activity a is a predecessor of activity b, i.e. $a \leq b$, then sentence s_a should not occur in the text *after* sentence s_b.

The optimal correspondence relation $\hat{C}$ is then the coherent correspondence relation C with the highest total similarity score between the activities and sentences. This is defined in Equation 4.

$$\hat{C} = \underset{C}{\operatorname{argmax}} \sum^{(a,s)\in C} sim(a, s) \tag{4}$$

Because the majority of process models are not purely sequential, finding $\hat{C}$ is not straightforward. Each ordering in which the activities of a process model can be executed must be considered as a possible ordering in which the activities are contained in the textual process description. Given that each of these orderings has a potentially huge set of possible correspondence relations to the sentence set S, this problem calls for an efficient solving approach.

We adopt a *best-first search* algorithm similar to those used in machine translation problems [17]. Instead of aligning one language to another, we here align the activities of A with sentences of S. Intuitively, the best-first search algorithm traverses a search space of partial hypotheses, which consist of activity-sentence alignments between A and S. The algorithm explores the search space by expanding the partial hypothesis with the highest possible score, while it exempts unpromising hypotheses from expansion. Because this approach exempts unpromising hypotheses from expansion, the explored search space is greatly reduced. Since the algorithm merely affects computational efficiency – not the resulting optimal correspondence relation $\hat{C}$ – we abstract from further details for reasons of brevity.[4] Section 4.5 describes how we assess the optimal correspondence relation to quantify the likelihood that it contains inconsistencies.

4.5 Inconsistency Assessment

The optimal correspondence relation $\hat{C}$ represents the best coherent alignment possible between activity set A and sentence set S. If this alignment is of insufficient quality, it can be expected that the model-text pair contains inconsistencies. An inconsistency exists if an activity cannot be coherently aligned to a sentence that refers to the same action. The semantic similarity measure $sim(a, s)$ quantifies this. An optimal correspondence $\hat{C}$ that contains an activity-sentence pair with a low similarity score thus implies that an activity exists that cannot be aligned to a sentence with a similar meaning. This means that the textual and model-based descriptions likely contain one or more inconsistencies. As a quantification of this likelihood we define a likelihood indicator ρ as the

[4] The interested reader is referred to e.g. [17,35] for a detailed description.

lowest similarity value found in the optimal correspondence relation. Equation 5 formalizes this concept.

$$\rho = \min\{sim(a, s) \mid (a, s) \in \hat{C}\} \tag{5}$$

Section 5 demonstrates the usefulness of the likelihood indicator ρ and, accordingly, the ability of our approach to identify inconsistent processes in a process model repository.

5 Evaluation

This section presents a quantitative evaluation that demonstrates how well the proposed approach is able to identify inconsistent model-text pairs in a collection. We have manually annotated the inconsistencies in a collection of 46 model-text pairs obtained from practice. This annotation is referred to as the *gold standard* against which we compare the results of our approach. Subsequently, we present the set-up of the evaluation, its results, and a discussion of the strengths and weaknesses of our approach.

5.1 Test Collection

To evaluate our approach, we use an existing collection of pairs of process models and manually created textual descriptions from [14]. The collection contains 46 model-text pairs that originate from different sources including academia, industry, textbooks, and public sector models.[5] The included process models are heterogeneous with regard to several dimensions, such as size and complexity. Also, the corresponding textual descriptions vary in several regards. For instance, they describe the processes from different perspectives (first and third person) and differ in terms of how explicitly and unambiguously they refer to the process model content. Hence, we believe that the collection is well-suited for achieving a high external validity of the results. Table 1 summarizes the main characteristics of the collection and the contained model-text pairs.

We involved three researchers in the creation of the gold standard. Two of them independently identified activity-to-sentence mappings for each model. This yielded an inter-annotator agreement of 92.9%. The biggest cause for discussion was the implicitness of some activity descriptions, such as seen for the "*select unchecked part*" activity in the bicycle manufacturing example. The 27 differences were discussed, involving a third researcher to settle ties.

Out of the 378 activities contained in the process models, five activities are described in the wrong place, whereas 26 activities can be considered to be missing. These lead to a gold standard that consists of 24 correct processes and 22 that contain between one and three erroneously described activities.

[5] For more details about the sources of the collection, please refer to [14].

Table 1. Overview of the test collection

ID	Source	P	P_i	Type	A	G	S
1	HU Berlin	4	1	Academic	9.0	6.5	10.3
2	TU Berlin	2	2	Academic	22.5	10.5	34.0
3	QUT	8	0	Academic	6.1	2.0	7.1
4	TU Eindhoven	1	1	Academic	18.0	8.0	40.0
5	Vendor Tutorials	3	2	Industry	5.3	1.3	7.0
6	inubit AG	4	3	Industry	9.0	3.8	11.5
7	BPM Practitioners	1	0	Industry	4.0	1.0	7.0
8	BPMN Practice Handbook	3	3	Textbook	5.0	2.0	4.7
9	BPMN Guide	6	5	Textbook	7.0	3.2	7.0
10	Federal Network Agency	14	5	Public Sector	8.0	3.1	6.4
	Total	**46**	**23**	-	**8.1**	**3.5**	**9.0**

Legend: P = Model-text pairs per source, P_i = Inconsistent pairs per source, A = Activities per model, G = Gateways per model, S = Sentences per Text,

5.2 Setup

To demonstrate the applicability of the approach presented in this paper, we test the following, different configurations:

- **Baseline:** As a baseline configuration, we aligned every activity a to the sentence s with the highest value for $sim(a, s)$. Prior to the computation of the similarity scores, we sanitize all sentences and activity labels.
- **Linguistic analysis:** For this configuration, prior to the computation of similarity scores, we applied all linguistic analysis activities described in Section 4.2. We thus subjected the textual description to text sanitization, resolved anaphoric references, and extracted relevant clauses.
- **Full configuration:** For the full configuration, we performed all linguistic analysis activities *and* included the ordering constraint described in Section 4.4. This configuration computes a correspondence relation between activity set A and sentence set S that achieves a maximal similarity score, while respecting the ordering constraints implied by the partial ordering of the process model.

We assess the performance of each of these configurations with standard information retrieval metrics. More specifically, we calculate *precision* and *recall* by comparing the computed results against a manually created gold standard. For a process collection P, we define a set P_τ as the set of processes with an assigned likelihood indicator ρ in the range $[0.0, \tau]$. P_I is the set of 22 processes that are inconsistent according to the gold standard. Each process in P_I contains at least one activity that is not included in the textual description or has activities that are described in a different order.

For a given P_τ, *precision* describes the fraction of processes a configuration classified as inconsistent that are also contained in P_I. *Recall* represents the fraction of all inconsistent processes from the gold standard which our approach successfully identified. Formally, the metrics are defined as shown by Equations 6 and 7.

$$\text{precision} = \frac{|P_I \cap P_\tau|}{|P_\tau|} \qquad (6) \qquad\qquad \text{recall} = \frac{|P_I \cap P_\tau|}{|P_I|} \qquad (7)$$

5.3 Results

We computed precision and recall scores for different values of threshold τ for each of the three configurations. When the value of τ increases, more model-text pairs are predicted to contain inconsistencies and thus included in P_τ.

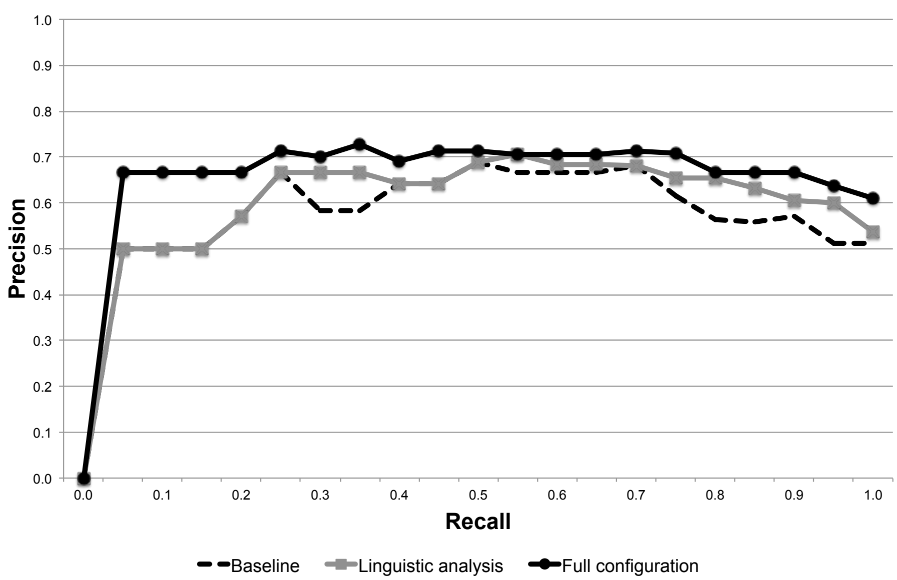

Fig. 4. Precision-recall graph for the performance of the three configurations

The precision-recall graph of Figure 4 shows that the full configuration consistently outperforms the baseline. The curves for the linguistic analysis and full configurations are always equal to or higher than the curve for the baseline configuration. This means that there are numerous cases for which the inclusion of these additional steps improves the results, it furthermore indicates that these steps *never* negatively impact the performance of the approach.

The improved results for the full approach also become apparent when considering the *F-measures* of the configurations. The F-measure represents the harmonic mean between precision and recall. For the baseline configuration, the maximum achieved F-measure equals 0.70. This performance is already promising, signaling that the semantic similarity measure we have selected is able to correctly identify a considerable number of inconsistencies. At this point, the baseline yields a recall of 0.91 against a precision of 0.57. The performance of the approach is further improved by including a linguistic analysis. This configuration achieves a maximum F-measure of 0.74, simultaneously improving both recall (0.96) and precision (0.60) in comparison to the baseline. The full configuration achieves an even higher F-measure of 0.77. It reaches a recall of 0.91 with a precision of 0.68. The full approach thus outperforms the precision of the baseline configuration by 11 percentage points.

The performance of the full approach is also demonstrated when we consider the point at which the approach has successfully identified all 22 inconsistent model-text pairs, i.e. the point when recall equals 1.0. The baseline configuration only reaches this point after considering 43 model-text pairs. It therefore hardly yields any benefits in comparison to a random selection as it makes 21 incorrect predictions. By contrast, the full configuration identifies all inconsistent processes after considering just 36 model-text pairs. Due to our linguistic analysis and the consideration of order, we thereby reduce the number of incorrect predictions by more than 33%.

5.4 Discussion

The evaluation shows that the full approach successfully identifies inconsistent model-text pairs from a collection while limiting the number of false positives. A post-hoc analysis reveals that the approach faces two main types of challenges.

First, the approach sometimes fails to recognize that an activity is contained in a textual description. These cases mainly occur when the description of activities is highly implicit or context-dependent. Consider, for example, an activity labeled "*use other sources*", as present in a process related to the procurement of information through various channels. The sentence fragment that describes this activity is "[..] *and sometimes you just happen to know somebody*". Due to its implicit description, aligning that activity to the appropriate sentence is difficult using natural language processing techniques. Similar problems occur when a textual description describes actions using references to earlier parts of the text. Most notably due to the anaphora resolution, the linguistic analysis successfully mitigates the impact of such problematic cases. Consequently, the full configuration of our approach detects inconsistencies more precisely.

Second, the approach, especially the baseline configuration, can return false negatives when it fails to detect inconsistencies in a model-text pair. In these cases, an activity is aligned with a sentence even though the activity is actually missing in the textual description. This happens when a strong semantic similarity between certain terms in an activity label and terms in the sentence exists, although neither this, nor any other sentence in the textual description, is related to the activity. The evaluation results demonstrate that the introduction of ordering constraints successfully avoids a number of such cases. For this configuration, it is no longer sufficient that a sentence just contains words semantically similar to those used in an activity label. Rather, the sentence must also occur in a proper location in the textual description. Hence, the approach including these constraints achieves higher recall values than the baseline configuration.

6 Conclusions

In this paper, we presented the first approach to automatically detect inconsistencies between textual and model-based process descriptions. The approach combines linguistic analysis, semantic similarity measures, and ordering relations to obtain a correspondence relation between the activities of a process model and the sentences of a textual process description. The approach subsequently assesses the quality of the obtained correspondence relation to predict whether or not a model-text pair contains inconsistencies. A quantitative evaluation shows that this approach successfully identifies the majority of incorrect process models, while yielding a low number of false positives. These insights result from a comparison of the predictions made by the approach against a manually constructed gold standard for a collection of real-life process descriptions. The evaluation furthermore reveals that the quality of the results is greatly improved due to the inclusion of tailored natural language processing techniques. By using our approach, organizations can thus quickly gain insights into the processes for which conflicts between the textual and model-based process descriptions are most likely. The effort that is needed to identify differences in large process model repositories is thereby greatly reduced. As such, organizations can focus their redesign efforts on the analysis and improvement of only their most problematic process descriptions.

In future work we set out to further develop approaches aimed at processes described using both models and text. The current approach can be extended by considering information beyond the control-flow dimension of a process. For instance, by deriving "*who does what, to whom, and where*" from sentences, or by comparing a model's conditional branches to the discourse in a textual description. The approach can also be broadened by performing a completeness check, i.e. by verifying whether all described activities are contained in the process model. Furthermore, the activity-sentence correspondence relation we obtain can be used for other purposes. Instead of using them to identify inconsistencies *ex post facto*, correspondence relations can form a basis to directly propagate one-sided process updates. In this way, the consistency between multiple process representations can be ensured, rather than corrected. Finally, we recognize that organizations also capture process information in formats other than the textual and model-based descriptions considered in this paper. Common examples include checklists, rules and regulations, and spreadsheets. In the future, we therefore aim to apply the techniques developed here on a broader spectrum of process representation formats.

References

1. Van der Aa, H., Leopold, H., Mannhardt, F., Reijers, H.A.: On the fragmentation of process information: Challenges, solutions, and outlook. In: Enterprise, Business-Process and Information Systems Modeling. Springer (2015)
2. Achananuparp, P., Hu, X., Shen, X.: The evaluation of sentence similarity measures. In: Data Warehousing and Knowledge Discovery, pp. 305–316. Springer (2008)
3. Allweyer, T.: BPMN 2.0: introduction to the standard for business process modeling. BoD–Books on Demand (2010)
4. Bajwa, I.S., Choudhary, M.A.: From natural language software specifications to UML class models. In: Enterprise Information Systems, pp. 224–237. Springer (2012)
5. Cayoglu, U., Dijkman, R., Dumas, M., Fettke, P., Garcıa-Banuelos, L., Hake, P., Klinkmüller, C., Leopold, H., Ludwig, A., Loos, P., et al.: The process model matching contest 2013. In: 4th International Workshop on Process Model Collections: Management and Reuse (PMC-MR'13) (2013)
6. Cayoglu, U., Oberweis, A., Schoknecht, A., Ullrich, M.: Triple-s: A matching approach for Petri nets on syntactic, semantic and structural level
7. Chakraborty, S., Sarker, S., Sarker, S.: An exploration into the process of requirements elicitation: A grounded approach. J. AIS 11(4) (2010)
8. De Marneffe, M.C., Manning, C.D.: The stanford typed dependencies representation. In: Coling 2008: Proceedings of the workshop on Cross-Framework and Cross-Domain Parser Evaluation. pp. 1–8 (2008)
9. Dijkman, R., Dumas, M., Van Dongen, B., Käärik, R., Mendling, J.: Similarity of business process models: Metrics and evaluation. Information Systems 36(2), 498–516 (2011)
10. Doan, A., Halevy, A.Y.: Semantic integration research in the database community: A brief survey. AI magazine 26(1), 83 (2005)
11. Dumas, M., Rosa, M., Mendling, J., Reijers, H.: Fundamentals of Business Process Management. Springer (2013)
12. Ehrig, M., Koschmider, A., Oberweis, A.: Measuring similarity between semantic business process models. In: Proceedings of the fourth Asia-Pacific conference on Comceptual modelling-Volume 67. pp. 71–80 (2007)
13. Euzenat, J., Shvaiko, P., et al.: Ontology matching, vol. 18. Springer (2007)
14. Friedrich, F., Mendling, J., Puhlmann, F.: Process model generation from natural language text. In: Advanced Information Systems Engineering. pp. 482–496. Springer (2011)
15. Giunchiglia, F., Shvaiko, P., Yatskevich, M.: Semantic matching. In: Encyclopedia of Database Systems, pp. 2561–2566. Springer (2009)
16. Gomez, F., Segami, C., Delaune, C.: A system for the semiautomatic generation of ER models from natural language specifications. Data & Knowledge Engineering 29(1), 57–81 (1999)
17. Jurafsky, D., Martin, J.H.: Speech & language processing. Pearson Education India (2000)

18. Kettinger, W., Teng, J., Guha, S.: Business Process Change: a Study of Methodologies, Techniques, and Tools. MIS quarterly pp. 55–80 (1997)
19. Klinkmüller, C., Weber, I., Mendling, J., Leopold, H., Ludwig, A.: Increasing recall of process model matching by improved activity label matching. In: Business Process Management, pp. 211–218. Springer (2013)
20. Kunze, M., Weidlich, M., Weske, M.: Behavioral similarity–a proper metric. In: Business Process Management, pp. 166–181. Springer (2011)
21. La Rosa, M., Dumas, M., Uba, R., Dijkman, R.: Business process model merging: An approach to business process consolidation. ACM Transactions on Software Engineering and Methodology (TOSEM) 22(2), 11 (2013)
22. Lavoie, B., Rambow, O., Reiter, E.: The modelexplainer. In: Eighth International Workshop on Natural Language Generation, Herstmonceux, Sussex (1996)
23. Leopold, H.: Natural language in business process models. Springer (2013)
24. Leopold, H., Mendling, J., Polyvyanyy, A.: Supporting process model validation through natural language generation. IEEE Transactions on Software Engineering 40(8), 818–840 (2014)
25. Lin, D.: An information-theoretic definition of similarity. In: ICML. vol. 98, pp. 296–304 (1998)
26. Manning, C.D., Raghavan, P., Schütze, H.: Introduction to information retrieval, vol. 1. Cambridge university press Cambridge (2008)
27. Manning, C.D., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S.J., McClosky, D.: The Stanford CoreNLP natural language processing toolkit. In: Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations. pp. 55–60 (2014)
28. Meziane, F., Athanasakis, N., Ananiadou, S.: Generating natural language specifications from UML class diagrams. Requirements Engineering 13(1), 1–18 (2008)
29. Mihalcea, R., Corley, C., Strapparava, C.: Corpus-based and knowledge-based measures of text semantic similarity. In: AAAI. vol. 6, pp. 775–780 (2006)
30. Noy, N.F.: Semantic integration: a survey of ontology-based approaches. ACM Sigmod Record 33(4), 65–70 (2004)
31. Rahm, E., Bernstein, P.A.: A survey of approaches to automatic schema matching. the VLDB Journal 10(4), 334–350 (2001)
32. Rosemann, M.: Potential Pitfalls of Process Modeling: Part A. Business Process Management Journal 12(2), 249–254 (2006)
33. Schumacher, P., Minor, M., Schulte-Zurhausen, E.: Extracting and enriching workflows from text. In: Information Reuse and Integration (IRI), 2013 IEEE 14th International Conference on. pp. 285–292. IEEE (2013)
34. Uba, R., Dumas, M., García-Bañuelos, L., La Rosa, M.: Clone detection in repositories of business process models. In: Business Process Management, pp. 248–264. Springer (2011)
35. Wang, Y.Y., Waibel, A.: Decoding algorithm in statistical machine translation. In: Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics. pp. 366–372 (1997)
36. Weidlich, M., Dijkman, R., Mendling, J.: The ICoP framework: Identification of correspondences between process models. In: Advanced Information Systems Engineering. pp. 483–498. Springer (2010)
37. Weidlich, M., Mendling, J., Weske, M.: Efficient consistency measurement based on behavioral profiles of process models. Software Engineering, IEEE Transactions on 37(3), 410–429 (2011)