

A Semantic Approach for Business Process Model Abstraction

Sergey Smirnov¹, Hajo A. Reijers², and Mathias Weske¹

¹ Hasso Plattner Institute, University of Potsdam, Germany

{sergey.smirnov,mathias.weske}@hpi.uni-potsdam.de

² Eindhoven University of Technology, The Netherlands

h.a.reijers@tue.nl

Abstract. Models of business processes can easily become large and difficult to understand. Abstraction has proven to be an effective means to present a readable, high-level view of a business process model, by showing aggregated activities and leaving out irrelevant details. Yet, it is an open question how to combine activities into high-level tasks in a way that corresponds to such actions by experienced modelers. In this paper, an approach is presented that exploits *semantic* information within a process model, beyond *structural* information, to decide on which activities belong to one another. In an experimental validation, we used an industrial process model repository to compare this approach with actual modeling decisions, showing a strong correlation between the two. As such, this paper contributes to the development of modeling support for the application of effective process model abstraction, easing the use of business process models in practice.

Key words: business process modeling, model management, business process model abstraction, activity clustering

1 Introduction

Business process models are used within a range of organizational initiatives [19]. However, human readers are limited in their cognitive capabilities to make sense of large and complex business process models [2, 33]. One well-known way to address this issue is by applying *abstraction*, the act of retaining essential properties of a process model on a particular level of analysis while hiding insignificant process details. Indeed, in a recent empirical investigation into the need for business process model abstraction [32], we found that its most prominent use case is the need for gaining a *quick overview* of the process. In such a situation, the user wants to familiarize herself with a business process but has only a large process model of many detailed activities at her disposal. To deal with such a demand, the process model can then be displayed as a partially ordered set of coarse-grained activities, each of which aggregates a number of lower-level activities. As an example, an abstraction of a process model that captures the creation of a forecasting report is shown in Fig. 1. In this figure, m is the initial model and m_a is the abstract model of the same process.

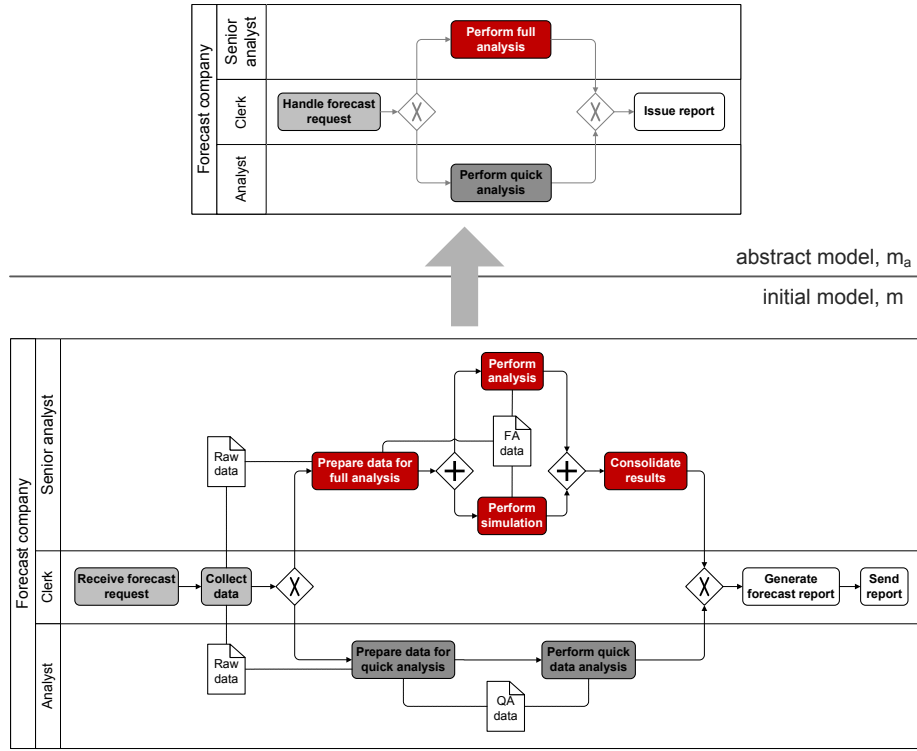


Fig. 1. Motivating example: initial model and its abstract counterpart

While it has been empirically shown that abstraction can significantly improve the sense-making of large process models [25], a limited insight exists into the criteria that experienced modelers use to decide on which activities to aggregate into new ones. A number of techniques has been proposed that exploit *structural* properties of a process model to arrive at abstract models [5, 24]. It seems likely that experienced process modelers take a wider range of properties into account rather than just a model’s control flow. For example, the fact that two activities use the same document and are executed by the same role may be used as relevant inputs in deciding to cluster these two into an aggregated activity. This situation applies, for example, to the activities *Prepare data for quick analysis* and *Perform quick data analysis* in Fig. 1.

In this paper, we complement the existing streams of work with respect to process model abstraction by proposing an abstraction technique that incorporates semantic aspects contained within a process model. We rely on the observation that industrial process models are often enriched with non-control flow model elements. Examples are: data that is being processed within an activity, IT systems invoked within particular activities, and roles assigned to activities. The central idea in this paper is that activities associated with the *same* non-control

flow elements are semantically *related* and, therefore, more appropriate candidates to be aggregated into the same activity than activities without shared elements.

A number of recent contributions exist that consider *semantic* aspects for aggregation, e.g., [8, 31]. However, their assumptions, e.g., the existence of an activity ontology [31], are too strict for generic use. Our approach is based on the application of the vector space model, an algebraic model popular in information retrieval [28]. As we will discuss in this paper, the use of vector spaces allows to determine the degree of similarity between activities according to several information types available in process models. We have validated the proposed technique applying it to a process model repository that is in use by a large European telecommunication organization. The repository incorporates hierarchical relations between high-level activities and the activities that they aggregate. Also, the process models contain various types of semantic information. The validation suggests that our approach closely approximates the decisions of the involved modelers to cluster activities.

The main contribution of this paper is a technique that may assist novice process modelers in the abstraction of complex process models by mimicking the abstraction decisions of more experienced modelers, as discovered from existing models. In this way, the technique allows to reuse activity aggregation principles for future aggregation decisions. Since the lack of experienced process modelers is a noted issue in many large modeling projects [26], this is a valuable asset to improve the process model quality. Meanwhile, the designed technique can also support experienced modelers enabling process model abstraction in conformance to their specific abstraction style. Hence, experts can accelerate their modeling routine configuring this technique, while staying in control over the modeling outcome. Finally, the technique can also be used to safeguard a particular “fingerprint” of a process model collection with respect to abstraction choices.

The paper is structured accordingly. We continue in Section 2 explaining the proposed algorithm, along with providing the required background knowledge. Section 3 empirically validates the proposed approach, using an industrial set of process models from the telecommunication sector. Finally, Section 4 contrasts our contribution with the related research, while Section 5 concludes the paper.

2 Activity Aggregation

This section elaborates on the proposed activity aggregation algorithm. After the introduction of the main concepts, we argue how activity aggregation can be interpreted as a clustering problem. We discuss a suitable clustering algorithm and alternative activity distance measures. The section focuses on one specific measure that enables the tuning of an activity aggregation. We explain how the aggregation setup is realized and show how the setup information can be mined from an existing process model collection.

2.1 Foundations

The designed aggregation algorithm inspects an activity environment, i.e., process model elements that are related to activities in a process model. Examples of such elements are data objects accessed by activities and roles supporting activity execution, e.g., see model in Fig. 1. The list of such model element types varies depending on the process modeling language, the tool at hand, modeling procedures taken into account, and the modeler's style. Each of the model element types can be considered as an activity property that has a specific value. Definition 1 formalizes the activity property concept.

Definition 1 (Activity Property Value and Activity Property Type).

Let \mathcal{P} be a finite nonempty set of activity property values. Alongside, \mathcal{T} is a finite nonempty set of activity property types. Mapping $type : \mathcal{P} \rightarrow \mathcal{T}$ assigns a type to each value.

The process model in Fig. 1 illustrates Definition 1. *Raw data*, *FA data*, and *Analyst* are examples of activity property values. The process model presents two activity property types: *Role* and *Data object*. For instance, $type(Raw\ data) = Data\ object$, $type(FA\ data) = Data\ object$, and $type(Analyst) = Role$. Further, we define a process model as follows.

Definition 2 (Process Model). A tuple $m_i = (A_i, G_i, F_i, P_i, props_i)$ is a *process model*, where:

- A_i is a finite nonempty set of activities;
- G_i is a finite set of gateways;
- $N_i = A_i \dot{\cup} G_i$ is the set of nodes, where $\dot{\cup}$ denotes a disjoint union of sets;
- $F_i \subseteq N_i \times N_i$ is the flow relation;
- $P_i \subseteq \mathcal{P}$ is a set of activity property values;
- $props_i : A_i \rightarrow 2^{P_i}$ is a mapping that assigns property values to an activity.

Definition 2 does not make a distinction between different gateway types, since the future discussion does not make use of them. Mapping $props_i$ assigns activity property values to model activities. Referring to model m in the motivating example of Fig. 1, mapping $props_i$ can be illustrated as $props_i(Collect\ data) = \{Clerk, Raw\ data\}$. Notice that Definitions 1 and 2 allow to manage the considered activity property types in a flexible fashion: it is enough to introduce a new activity property type to set \mathcal{T} , the values to \mathcal{P} , and respectively update mapping $type$. Thereafter, new activity properties can be easily considered within the activity aggregation. Finally, we postulate the concept of a process model collection.

Definition 3 (Process Model Collection). A tuple $c = (M, A, P, \sigma)$ is a *process model collection*, where:

- M is a nonempty finite set of n process models with elements $m_i = (A_i, G_i, F_i, P_i, props_i)$, where $i = 1, 2, \dots, n$;
- $A = \dot{\cup}_{i=1,2,\dots,n} A_i$ is a set of collection activities;
- $P = \cup_{i=1,2,\dots,n} P_i$ is a set of collection activity property values;

- $\sigma \subseteq M \times M$ is a subprocess relation refining a process model with subprocess models, such that $\forall m_i, m_j \in M$, where $j = 1, 2, \dots, n$ and $i \neq j$, if $(m_i, m_j) \in \sigma$ then $(m_j, m_i) \notin \sigma^+$, where σ^+ is a transitive reflexive closure of σ .

Definition 3 explicitly enumerates the model collection activities and property value types. The relation σ formalizes the subprocess relation that exists between models. Note that according to the definition, σ enables only a process model hierarchy without loops. Without loss of generality in the remainder of this paper we discuss abstraction of process models within a process model collection. Indeed, a process model m_i can be seen as a trivial process model collection $c = (\{m_i\}, A_i, P_i, \emptyset)$.

2.2 Activity Aggregation as Cluster Analysis Problem

In this paper we interpret activity aggregation as a problem of cluster analysis. Consider process model $m_i = (A_i, G_i, F_i, P_i, props_i)$ from process model collection $c = (M, A, P, \sigma)$. The set of objects to be clustered is the set of activities A_i . The objects are clustered according to a distance measure: objects that are “close” to each other according to this measure are put together. The distance between objects is evaluated through analysis of activity property values P . The cluster analysis outcome, activity clusters, correspond to coarse-grained activities of the abstract process model. While cluster analysis provides a large variety of algorithms, e.g., see [29], we focus on one algorithm that suits the business process model abstraction use case in focus.

In the considered scenario, the user demands control over the number of activities in the abstract process model. For example, a popular practical guideline is that five to seven activities are displayed on each level in the process model [30]. Provided a fixed number, e.g. 6, the clustering algorithm has to assure that the number of clusters equals the request by the user. We turn to the use of k-means clustering algorithm, as it is simple to implement and typically exhibits good performance [16]. K-means clustering partitions an activity set into k clusters. The algorithm assigns an activity to the cluster, which centroid is the closest to this activity. To evaluate an activity distance, we analyze activity property values P . We foresee a number of alternative activity distance measures and elaborate on them in the next section.

2.3 Activity Distance Measures

To introduce the distance measure among activities we represent activities as vectors in a vector space. Such an approach is inspired by the vector space model, an algebraic model widely used in information retrieval [28]. The space dimensions correspond to activity property values P and the vector space can be captured as vector $(p_1, \dots, p_{|P|})$, where $p_j \in P$ for $j = 1, \dots, |P|$. Consider an example set of property values $P' = \{FA\ data, QA\ data, Raw\ data\}$ and the corresponding vector space presented in Fig. 2. A vector \mathbf{v}_a representing an activity $a \in A_i$ in process model $m_i = (A_i, G_i, F_i, P_i, props_i)$ is constructed as follows. If activity a

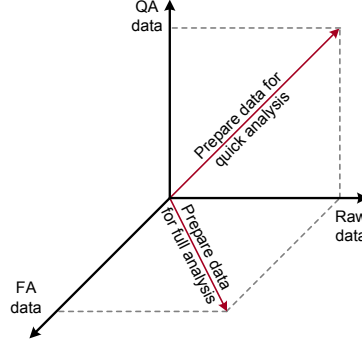


Fig. 2. Example of a vector space formed by dimensions *FA data*, *QA data*, *Raw data*

is associated with a property value $p_j \in P_i$, the corresponding vector dimension $\pi_j(\mathbf{v}_a)$ has value 1; otherwise, the dimension $\pi_j(\mathbf{v}_a)$ has value 0:

$$\pi_j(\mathbf{v}_a) = \begin{cases} 1, & \text{if } p_j \in \text{props}_i(a); \\ 0, & \text{otherwise.} \end{cases}$$

For process model m in Fig. 1, activities *Prepare data for quick analysis* and *Prepare data for full analysis* correspond, respectively, to vectors $\mathbf{v}_1 = (0, 1, 1)$ and $\mathbf{v}_2 = (1, 0, 1)$ in the vector space with dimensions *FA data*, *QA data*, *Raw data*, see Fig. 2.

Similarity of two vectors in the space is defined by the angle between these vectors: the larger the angle, the more distant the activities are. Typically, the cosine of the angle between two vectors is used as a vector similarity measure:

$$\text{sim}(a_1, a_2) = \cos(v_{a_1}, v_{a_2}) = \frac{v_{a_1} \cdot v_{a_2}}{\|v_{a_1}\| \|v_{a_2}\|} \quad (1)$$

Then, the distance between two activities is:

$$\text{dist}(a_1, a_2) = 1 - \text{sim}(a_1, a_2) \quad (2)$$

By construction the vector dimension values are non-negative. Hence, the activity similarity and activity distance measures vary within the interval $[0, 1]$.

For a process model collection $c = (M, A, P, \sigma)$ we distinguish two types of vector spaces. On the one hand, a vector space can be formed by the dimensions corresponding to the activity property values disregard their type, i.e., all elements of P . We reference such spaces as *heterogeneous vector spaces*. An example of a heterogeneous vector space is a space with 6 dimensions *Analyst*, *Clerk*, *FA data*, *QA data*, *Raw data*, and *Senior analyst*. On the other hand, a vector space can be formed by the dimensions corresponding to the activity property values of a particular type. Given an activity property type t , such a space is formally defined by the set $P_t = \{\forall p \in P : \text{type}(p) = t\}$. We refer to such spaces as *homogeneous vector spaces*. Fig. 2 provides an example of a homogeneous vector

space formed by activity properties of type *Data object*. We denote the activity distance in a heterogeneous space with $dist_h(a_1, a_2)$ and in a homogeneous vector space with $dist_t(a_1, a_2)$, where t is the respective activity property type. Both distance measures can be employed for activity aggregation. If the user wants to make use of one activity property type t only, the distance is defined by $dist_t$. To cluster activities according to several activity property types, $dist_h$ can be employed. In addition, we introduce an alternative distance measure $dist_{agg}$ that aggregates multiple homogeneous distance measures $dist_t$:

$$dist_{agg}(a_1, a_2) = \frac{1}{|T|} \sum_{\forall t \in T} w_t \cdot dist_t(a_1, a_2) \quad (3)$$

In Equation 3, the set T corresponds to the activity property types that appear in process model collection c . Then, function $dist_{agg}$ is the weighted average value of distance measures in the vector spaces corresponding to the available activity property types. Coefficient w_t is the weight of $dist_t$ indicating the impact of the activity distance according to property type t . We reference all the weights in Equation 3 as $\mathbf{W} = (w_{t_1}, \dots, w_{t_n})$, where $n = |T|$. In the remainder of this section we will explain the role of vector \mathbf{W} .

2.4 Process Model Collection Abstraction Fingerprint

The application of different abstraction operations to one process model leads to various abstract representations of the modeled business process. The differences between abstraction operations are explained by their pragmatics, i.e., various abstraction purposes. If the abstraction is realized by a human, the modeling habits of the designer are reflected in the abstraction operation as well. Hence, abstraction pragmatics and modeling habits of the designer are inherent properties of the abstraction operation and together form an *abstraction style*. We use vector \mathbf{W} in Equation 3 to model an abstraction style.

From the user perspective vector \mathbf{W} is the tool to express the desired abstraction style. We foresee two scenarios how vector \mathbf{W} can be obtained. In the first scenario, the user explicitly specifies \mathbf{W} . This approach is useful if the user wants to introduce a new abstraction style. However, coming up with an appropriate value for \mathbf{W} may be challenging. Hence, the second scenario implies that vector \mathbf{W} is mined from a process model collection enriched with subprocess relation (formalized with σ in Definition 3). The discovered vector is a “fingerprint” of the process model collection with respect to the used abstraction style. We will now describe an approach how vector \mathbf{W} can be discovered from such a process model collection.

The discovery process of a model collection’s abstraction fingerprint is driven by the following argumentation. Activities of a process model collection are aggregated into aggregated activities, i.e. subprocess placeholders, by the model designer. We aim to achieve an activity clustering algorithm that approximates this aggregation behavior of a human. This is possible if an activity distance measure employed by the algorithm resembles the criteria that a human designer

uses to aggregate activities into a subprocess. The exact criteria are unknown. Yet, for each pair of activities we can observe the outcome: Either the activities belong to different subprocesses or to the same one. For a process model collection $c = (M, A, P, \sigma)$ function $diff$ formalizes this observation:

$$diff(a_k, a_l) = \begin{cases} 0, & \text{if } a_k, a_l \in A_i; \\ 1, & \text{otherwise.} \end{cases} \quad (4)$$

To mine the process model collection fingerprint \mathbf{W} we select its value in such a way that the behavior of function $dist_{agg}$ approximates the behavior of $diff$. The discovery of vector \mathbf{W} is realized by means of linear regression. In our setting, the values $dist_t$ are considered independent variables and the value of function $diff$ the dependent variable. Components of vector \mathbf{W} are the regression coefficients. The standardized coefficients indicate the impact of each activity property type on the abstraction style. Hence, it is possible to reveal criteria employed by the human designer during abstraction. Furthermore, the regression's coefficient of determination R^2 allows to judge how well the obtained statistical model explains the observed behavior. For our purposes, R^2 suggests if the discovered statistical model can be used for business process model abstraction.

3 Empirical Validation

The proposed activity aggregation mechanism calls for validation. The goal of the validation is to learn how well the proposed operation approximates the abstraction style of human modelers. We performed an empirical validation of the approach by conducting an experiment with a real world business process model collection. This section provides a detailed discussion of the validation; it describes in detail the explored process model collection, explains the experiment design, and discusses the validation results.

3.1 Validation Setup

As a research object we choose a set of business process models from a large telecommunication service provider. This organization is currently in the process of setting up a repository with high-quality process models, which are brought together for the purpose of consultation and re-use by business users. The model set includes 30 elaborate models, enriched with activity properties of the following types: *roles*, *responsible roles*, *IT systems*, and *data objects*. It is noticeable that a special type of roles, i.e., *responsible roles*, is also distinguished in these models. In addition to these non-control flow types of information, we also study the impact that activity labels and activity neighboring control flow elements have on the decision to aggregate activities into the same subprocess. To compare activities with respect to their labels, the corresponding vector space is formed by the words that appear in the labels. Against this background, finding the distance between activities becomes an information retrieval task as labels can

	Nodes	Activities	Role	Responsible	role	IT system	Data object
Average	15.5	6.3	2.1	0.76		1.5	0.76
Minimum	5	1	0	0		0	0
Maximum	48	20	5	2		7	17

Table 1. Properties of business process models used in the validation

be treated as documents in information retrieval. The comparison of activities with respect to their neighbors shows whether the neighborhoods of the two activities intersect, i.e., contain the same flow elements. Table 1 outlines the relevant properties of the process models. In the existing repository, the models are hierarchically organized using a subprocess relation. Within the model set, we have identified 8 subprocess hierarchies. Each hierarchy contains a root process model refined with subprocesses, allowing for several levels of refinement.

To formally validate how good the designed activity aggregation approximates the behavior of modelers clustering a set of activities into the same subprocess, we selected the following approach. For each pair of activities that belong to the same process hierarchy, we have evaluated two values in the process model collection: *diff* and *dist*. Here, *diff* describes the human abstraction style, which indicates whether the activities have been decided to be placed in the same subprocess or not. The value of *dist* represents the vector space distance between the two activities in accordance with our approach. To discover if the two approaches yield similar results, we study the correlation between the two variables. A strong correlation of two variables implies that *dist* is a good distance measure in the clustering algorithm. In this case, the inclusion of two activities within the same subprocess is mirrored by a close positioning of the corresponding vectors in the vector space. Given the nature of the observed variables, we employ Spearman’s rank correlation coefficient.

In the following, we first investigate the human abstraction style in the model collection as a whole. Then, we verify the results organizing a K-fold cross validation. We partition the model sample into 4 subsamples, i.e., $k = 4$ and perform four tests. In each test, three subsamples are used to discover vector \mathbf{W} , while the fourth subsample is used to evaluate the correlation values between the *diff* and *dist* measures in different vector spaces. In this way, a more reliable insight is developed into the question whether the human abstraction style can be mimicked in contrast to using the whole process model collection for both the discovery and the evaluation of this correlation.

3.2 Validation Results

Table 2 outlines the validation’s results. The columns in the table correspond to distance measures. While the first 6 columns correspond to distances in homogeneous spaces, the last three columns reflect the distance measure taking into account multiple activity properties. All three distance measures make use of the activity property types in columns 1–6. The distance $dist_h$ is measured

Experiment	$\rho(dist_t, diff)$						$\rho(dist_h, diff)$	$\rho(dist_{avg}, diff)$	$\rho(dist_{agg}, diff)$
	Role	Responsible role	IT system	Data object	Label	Neighbor			
All models	0.70	0.61	0.60	—	0.34	0.58	0.74	0.65	0.77
Test ₁	0.79	0.76	0.75	—	0.42	0.60	0.79	0.69	0.79
Test ₂	0.64	0.56	0.56	—	0.43	0.62	0.68	0.70	0.70
Test ₃	0.68	0.58	0.58	—	0.53	0.64	0.68	0.72	0.71
Test ₄	0.61	0.47	0.45	—	0.20	0.48	0.70	0.56	0.52
Average ₁₋₄	0.68	0.59	0.58	—	0.39	0.59	0.71	0.67	0.68

Table 2. Correlation values observed in the K-fold cross validation

in heterogeneous vector space, where dimensions are activity property values of types listed in columns 1–6. The distance measure $dist_{avg}$ is the average value of distances in columns 1–6. The distance measure $dist_{agg}$ is evaluated according to Equation 3. Vector \mathbf{W} used in $dist_{agg}$ is obtained using linear regression as described in the previous section. Rows of Table 2 correspond to experiments. The first row describes the study of the whole model collection. Rows 2–5 describe the results of 4 tests along the K-fold cross validation we explained earlier, while the last row provides the average correlations observed in the 4 separate tests.

The correlation values that are presented in Table 2 are all significant using a confidence level of 99%, i.e., all p values are lower than 0.01. However, no statistically significant results were obtained for the distance in the homogeneous vector space that corresponds to *Data objects*. Overall, the presented correlation values range around 0.7. This level is generally considered to indicate a strong correlation [11, 12], particularly in situations where human decision making is involved. Therefore, we can speak of a strong relation between the $dist$ and $diff$ measures.

Among the distance measures in homogeneous spaces, one can point out the distance in the *Role* space that overall displays the highest correlation values for the different studies (0.61–0.79). In contrast, correlation values for *Label* are the lowest (0.20–0.53). Another observation is that distances taking into account multiple activity property types tend to have higher correlations. From these, $dist_{agg}$ outperforms all other distance measures with a value arriving at 0.77 when all models are considered. For the average values of the K-fold cross validations, however, $dist_h$, $dist_{avg}$, and $dist_{agg}$ demonstrate a similar performance, with correlation values of 0.71, 0.67, and 0.68 respectively. This observation can be explained by the fact that $dist_{agg}$ is parameterized by vector \mathbf{W} —the abstraction fingerprint of a particular model set. Thus, the distance measure $dist_{agg}$ “trained” on one model set may never excel $dist_{avg}$, once the set of models is changed. Tests 1–4 support this argumentation. Note that this result does not restrict the

applicability of the approach: in a real world setting, the goal is to transfer the abstraction style from one model set to another. The average values in the lower row should, therefore, be seen as most important from the ones displayed.

A careful inspection of the linear regression results associated with parameterizing vector \mathbf{W} provides additional insights. In particular, we are interested in the observed R^2 values and the beta coefficients (also known as “standardized coefficients”). The R^2 for the whole model set, as well as the average value for the K-fold cross validation is 0.52. This value shows the explained level of variation in abstraction style as explained by the various distance measures under consideration and can be considered as moderately strong. The beta coefficients of the distance measures in various spaces reveal their impact on the activity aggregation. The beta coefficients for activity property types *Role* and *Responsible role* have average values of 0.55 and 0.37, respectively. At the same time, the standardized coefficients of *Neighbor* and *Label* property types fluctuate around 0. The average value for *IT systems* is in between, with a beta coefficient of 0.19. The provided numbers illustrate that the activity property types *Role* and *Responsible role* have a big impact on the abstraction style of the considered process model collection. *IT systems* also contributes to the activity aggregation, but the influence of activity labels and activity neighborhood is insignificant. Clearly, such insights may differ from one process model to the other.

The validation indicates that the suggested distance measures can be used in a close approximation of the abstraction style of human modelers. Among the introduced measures, $dist_{agg}$ is of great interest, as it takes into account the abstraction style of a particular process model collection. Furthermore, the validation revealed activity property types, *Role* and *Responsible role*, that have the highest impact on the abstraction style for this particular collection.

4 Related Work

The topic of business process model abstraction can be related to several research streams. We identify these streams looking both from the perspective of the disciplines of software engineering and business process management.

Model properties and relations are thoroughly investigated in the software engineering area. For instance, in [21, 22] Kühne elaborates on the concepts of model, metamodel, model types, and model relations. These works systematically describe and organize relations, e.g., generalization and classification, which are seminal for the problem of model abstraction. Closely related are also the studies that cover model granularity. In [17], the authors investigate model and metamodel granularity. The authors compare several metamodels and come up with best practices with respect to granularity. One can observe that the relation between a coarse-grained activity in an abstract model and its counterparts in the initial model is the meronymy, or part-of, relation. Meronymy has been studied in depth in the software engineering domain [3, 13]. Although the referenced papers do not provide concrete techniques for the implementation of abstraction

within process models, they facilitate a better problem understanding and help to identify the main concepts in this domain.

Business process management is the discipline concerned with using methods, techniques, and software to design, enact, control, and analyze operational processes. A large body of knowledge corresponds to process model analysis based on model transformations. Model transformations can be reused in the context of the abstraction problem. An example of such a transformation consists of reduction rule sets for Petri nets, e.g., see [4, 23, 27]. Each reduction rule explicitly defines a structural fragment to be discovered in the model and a method of this fragment transformation. Hence, reduction rule sets enable process model abstraction through iterative rule application. As the transformed process fragments are explicitly defined, each reduction rule set handles only a particular model class. Thereby, each reduction rule set requires an argument about the model class reducible with the given rules. The model class limits the application of abstraction approaches based on reduction rules [5, 10]. Process model decomposition approaches are free of this limitation: they seek for process fragments with particular properties. An example of such a decomposition is presented in [34], where single entry single exit fragments are discovered. The result of process model decomposition is the hierarchy of process fragments according to the containment relation, i.e., the process structure tree. Such a tree can be used for abstraction in process models [24]. Finally, one can distinguish model transformations that preserve process behavior properties. In [1], van der Aalst and Basten introduce three notions of behavioral inheritance for WF-nets and study inheritance properties. The paper suggests model transformations, such that the resulting model inherits the behavior of the initial model. An approach for process model abstraction can exploit such transformations as basic operations. While the outlined model transformations can support solving the general problem of process model abstraction, they all focus on structural and behavioral aspects of models and model transformations, leaving the semantic aspect out of scope.

Many tasks in the management of large process model collections can be traced back to the problem of *activity matching*, which is closely related to the problem of business process model abstraction. Examples of such management tasks are: the search for a particular process model over a process model set or ensuring the consistency of models capturing one and the same process from different perspectives. Activity matching is realized through analysis of activity properties: activity labels, referenced data objects and neighboring activities. In [9, 35] the authors suggest activity matching algorithms and evaluate them. While the named works explore the existing process models and do not directly address the problem of process model abstraction, their results have a potential of being applied in business process model abstraction. Semantic aggregation of activities relates to research on semantic business process management. Notice that process models enriched with semantic information facilitate many process analysis tasks, see [18]. Along this line of research, several authors argue how to use activity ontologies to realize activity aggregation [6, 7]. It should be noticed,

however, that such works imply the existence of a semantic description for model elements and their relations, which is a restriction that rarely holds in real world settings.

Establishing an activity’s granularity level is also a recurrent challenge in process mining, where logs contain records that are often very fine-granular. As such, the process models directly mined from the logs can be overloaded with information making them hard to comprehend. Activity clustering is an efficient means to raise the abstraction level for the mined models. In [14, 15] Günther and van der Aalst propose activity aggregation mechanisms based on clustering algorithms. The mechanisms extensively use information present in process logs, but which are less common for process models, i.e., timestamps of activity starts and stops, activity frequencies, and transition probabilities. Thus, in contrast to the activity aggregation approach proposed in this paper, process mining considers other activity property types for clustering and utilize other clustering algorithms.

5 Conclusions and Future Work

Despite business process model abstraction has been addressed in a number of research endeavors, this paper proposes a novel approach in this area. Specifically, it exploits semantic aspects—beyond the control-flow perspective—to determine a similarity between different activities for the purpose of simplifying process model abstraction. Relevant levels of similarity can be determined on the basis of existing process models in which abstraction was already applied.

Our main contribution is a method to discover sets of related activities, where each set corresponds to a coarse-grained activity of an abstract process model. As a second contribution, we propose an approach to discover an abstraction style inherent to a given process model collection, which is reusable for abstraction of new process models. Both contributions are of practical interest, as they addresses model management issues recurrently appearing in process model projects. The experimental validation provides strong support for the applicability and effectiveness of the presented ideas.

Our approach is characterized by a number of limitations and assumptions. First of all, it builds on the assumption that all kinds of semantic information, such as data objects, roles, and resources, can be observed within the descriptions of process models in industrial collections. The process model collection we obtained through our cooperation with a large telecommunication company clearly confirms this idea, but this also applies to other industrial repositories, such as the SAP Reference Model [20]. Secondly, in our validation we have merely focused on the appearance or not of two activities being within the same subprocess or process model, although it can be imagined that a more fine-grained correspondence measure could yield even more useful results.

These and other limitations guide our future research plans. The direct next step for us is the use of advanced vector space models reflecting the relations between different activity property values. Such models enable activity clustering

algorithms to consider the *structure* of organigrams and data object relations. Meanwhile, it can also be beneficial to consider other clustering algorithms and compare the outcome with the solution introduced in this paper. From a practical perspective, it is important to suggest names for coarse-grained activities that are products of activity aggregation. Finally, we would like to improve the validation method for activity aggregation. On the one hand, this implies replacing correlation with an alternative metric for activity aggregation quality. On the other hand, the validation will require an empirical study involving human modelers and stakeholders, who can evaluate the proposed activity aggregation.

References

1. W. M. P. van der Aalst and T. Basten. Life-Cycle Inheritance: A Petri-Net-Based Approach. In *ICATPN 1997*, volume 1248 of *LNCS*, pages 62–81. Springer, 1997.
2. E.R. Aguilar, F. Ruiz, F. García, and M. Piattini. Evaluation Measures for Business Process Models. In *SAC 2006*, pages 1567–1568. ACM, 2006.
3. F. Barbier, B. Henderson-Sellers, A. Le Parc-Lacayrelle, and J.-M. Bruel. Formalization of the Whole-Part Relationship in the Unified Modeling Language. *IEEE TSE*, 29(5):459–470, 2003.
4. G. Berthelot. Transformations and Decompositions of Nets. In *Advances in Petri nets 1986*, volume 254 of *LNCS*, pages 359–376. Springer, 1987.
5. R. Bobrik, M. Reichert, and T. Bauer. View-Based Process Visualization. In *BPM 2007*, volume 4714 of *LNCS*, pages 88–95. Springer, 2007.
6. F. Casati and M.-Ch. Shan. Semantic Analysis of Business Process Executions. In *EDBT 2002*, pages 287–296. Springer, 2002.
7. A.K. Alves de Medeiros, W. M. P. van der Aalst, and C. Pedrinaci. Semantic Process Mining Tools: Core Building Blocks. In *ECIS 2008*, pages 1953–1964, Galway, Ireland, 2008.
8. C. Di Francescomarino, A. Marchetto, and P. Tonella. Cluster-based Modularization of Processes Recovered from Web Applications. *Journal of Software Maintenance and Evolution: Research and Practice*, 2010.
9. R. M. Dijkman, M. Dumas, and L. García-Bañuelos. Graph Matching Algorithms for Business Process Model Similarity Search. In *BPM 2009*, volume 5701 of *LNCS*, pages 48–63. Springer, 2009.
10. R. Eshuis and P. Grefen. Constructing Customized Process Views. *Data Knowl. Eng.*, 64(2):419–438, 2008.
11. A.N. Franzblau. *A Primer of Statistics for Non-statisticians*. Harcourt, Brace & World New York, 1958.
12. B. B. Gerstman. StatPrimer – Version 6.4. Technical report, San Jose State University, 2010. <http://www.sjsu.edu/faculty/gerstman/StatPrimer/>.
13. G. Guizzardi. Modal Aspects of Object Types and Part-Whole Relations and the *de re/de dicto* Distinction. In *CAiSE 2007*, volume 4495 of *LNCS*, pages 5–20. Springer, 2007.
14. C. W. Günther and W. M. P. van der Aalst. Mining Activity Clusters from Low-Level Event Logs. *BETA Working Paper Series*, WP 165, 2006.
15. C. W. Günther and W. M. P. van der Aalst. Fuzzy Mining—Adaptive Process Simplification Based on Multi-perspective Metrics. In *BPM 2007*, volume 4714 of *LNCS*, pages 328–343. Springer, 2007.

16. J. Hartigan. *Clustering Algorithms*. John Wiley and Sons, New York, USA, 1975.
17. B. Henderson-Sellers and C. Gonzalez-Perez. Granularity in Conceptual Modelling: Application to Metamodels. In *ER 2010*, volume 6412 of *LNCS*, pages 219–232. Springer, 2010.
18. M. Hepp, F. Leymann, J. Domingue, A. Wahler, and D. Fensel. Semantic Business Process Management: A Vision Towards Using Semantic Web Services for Business Process Management. In *ICEBE*, pages 535–540. IEEE Computer Society, 2005.
19. M. Indulska, J. Recker, M. Rosemann, and P. Green. Business Process Modeling: Current Issues and Future Challenges. In *CAiSE*, volume 5565 of *LNCS*, pages 501–514. Springer, 2009.
20. G. Keller and T. Teufel. *SAP R/3 Process Oriented Implementation*. Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA, 1998.
21. Th. Kühne. Matters of (Meta-) Modeling. *Software and Systems Modeling*, 5(4):369–385, 2006.
22. Th. Kühne. Contrasting Classification with Generalisation. In *APCCM 2009*, volume 96 of *CRPIT*, January 2009.
23. J. Mendling, H. Verbeek, B. van Dongen, W. M. P. van der Aalst, and G. Neumann. Detection and Prediction of Errors in EPCs of the SAP Reference Model. *Data Knowl. Eng.*, 64(1):312–329, 2008.
24. A. Polyvyanyy, S. Smirnov, and M. Weske. The Triconnected Abstraction of Process Models. In *BPM 2009*, volume 5701 of *LNCS*, pages 229–244. Springer, 2009.
25. H. A. Reijers and J. Mendling. Modularity in Process Models: Review and Effects. In *BPM 2008*, pages 20–35. Springer, 2008.
26. M. Rosemann. Potential Pitfalls of Process Modeling: Part A. *Business Process Management Journal*, 12(2):249–3254, 2006.
27. W. Sadiq and M. E. Orlowska. Analyzing Process Models Using Graph Reduction Techniques. *Information Systems*, 25(2):117–134, 2000.
28. G. Salton, A. Wong, and C. S. Yang. A Vector Space Model for Automatic Indexing. *Communications of the ACM*, 18(11):613–620, 1975.
29. S. E. Schaeffer. Graph Clustering. *Computer Science Review*, 1(1):27–64, 2007.
30. A. Sharp and P. McDermott. *Workflow Modeling: Tools for Process Improvement and Applications Development*. Artech House Publishers, 2008.
31. S. Smirnov, R. M. Dijkman, J. Mendling, and M. Weske. Meronymy-Based Aggregation of Activities in Business Process Models. In *ER 2010*, volume 6412 of *LNCS*, pages 1–14. Springer, 2010.
32. S. Smirnov, H. A. Reijers, T. Nugteren, and M. Weske. Business Process Model Abstraction: Theory and Practice. Technical Report 35, Hasso Plattner Institute, 2010. <http://bpt.hpi.uni-potsdam.de/pub/Public/SergeySmirnov/abstractionUseCases.pdf>.
33. I. T. P. Vanderfeesten, H. A. Reijers, J. Mendling, W. M. P. van der Aalst, and J. Cardoso. On a Quest for Good Process Models: The Cross-Connectivity Metric. In *CAiSE 2008*, volume 5074 of *LNCS*, pages 480–494. Springer, 2008.
34. J. Vanhatalo, H. Völzer, and J. Koehler. The Refined Process Structure Tree. In *BPM 2008*, pages 100–115. Springer, 2008.
35. M. Weidlich, R. M. Dijkman, and J. Mendling. The ICoP Framework: Identification of Correspondences between Process Models. In *CAiSE 2010*, volume 6051 of *LNCS*, pages 483–498. Springer, 2010.