Towards semantic process mining through knowledge-based trace abstraction

G. Leonardi (1), M. Striani (2), S. Quaglini (3), A. Cavallini (4), S. Montani (1)

(1) DISIT, Computer Science Institute, Università del Piemonte Orientale, Alessandria, Italy
(2) Department of Computer Science, Università di Torino, Italy
(3) Department of Electrical, Computer and Biomedical Engineering, Università di Pavia, Italy
(4) I.R.C.C.S. Fondazione "C. Mondino", Pavia, Italy - on behalf of the Stroke Unit Network (SUN) collaborating centers

Abstract. Many information systems nowadays record data about the process instances executed at the organization in the form of *traces* in a log. In this paper we present a framework able to convert actions found in the traces into higher level concepts, on the basis of domain knowledge. Abstracted traces are then provided as an input to semantic process mining.
The approach has been tested in the medical domain of stroke care, where we show how the abstraction mechanism allows the user to mine process models that are easier to interpret, since unnecessary details are hidden, but key behaviors are clearly visible.

Keywords: Semantic process mining; knowledge-based trace abstraction; medical applications

1 Introduction

Most commercial information systems, including those adopted by many health care organizations, record information about the executed process instances in a log [29]. The log stores the sequences (*traces* [29] henceforth) of actions that have been executed at the organization, typically together with key execution parameters, such as times, cost and resources. Logs can be provided in input to **process mining** [29, 30] algorithms, a family of a-posteriori analysis techniques able to extract non-trivial knowledge from these historic data; within process mining, *process model discovery* algorithms, in particular, take as input the log traces and build a process model, focusing on its control flow constructs. Classical process mining algorithms, however, provide a purely syntactical analysis, where actions in the traces are processed only referring to their names. Action names are strings without any semantics, so that identical actions, labeled by synonyms, will be considered as different, or actions that are special cases of other actions will be processed as unrelated.

Relating *semantic structures*, such as ontologies, to actions in the log, not only can solve the synonyms issue, but also can enable trace comparison and process mining techniques to work at *different levels of abstraction* (i.e., at the level of instances and/or concepts) and, therefore, to mask irrelevant details, to promote reuse, and, in general, to make process analysis much more flexible and reliable.

In fact, it has been observed that human readers are limited in their cognitive capabilities to make sense of large and complex process models [1, 33], while it would be often sufficient to gain a quick overview of the process, in order to familiarize with it in a short amount of time. Of course, deeper investigations can still be conducted, subsequently, on the detailed (ground) process model.

Interestingly, **semantic process mining**, defined as the integration of semantic processing capabilities into classical process mining techniques, has been recently proposed in the literature (see Section 5). However, while more work has been done in the field of semantic *conformance checking* (another branch of process mining) [10, 13], to the best of our knowledge semantic *process model discovery* needs to be further investigated.

In this paper, we present a **knowledge-based abstraction mechanism** (see Section 2), able to operate on log traces. In our approach:

- actions in the log are mapped to the ground terms of an ***ontology***;
- a ***rule base*** is exploited, in order to identify which of the multiple ancestors of an action should be considered for abstracting the action itself. ***Medical knowledge and contextual information*** are resorted to in this step;
- when a set of consecutive actions on the trace abstract as the same ancestor, they are merged into the same abstracted *macro-action*, labeled as the common ancestor at hand. This step requires a proper treatment of delays and/or actions in-between that descend from a different ancestor.

Our abstraction mechanism is then provided as an input to **semantic process mining** (see Section 3). In particular, we rely on classical *process model discovery* algorithms embedded in the open source framework ProM [32], made semantic by the exploitation of domain knowledge in the abstraction phase.

We also describe our experimental work (see Section 4) in the field of stroke care, where the application of the abstraction mechanism on log traces has allowed us to mine simpler and more understandable (from the clinical point of view) process models.

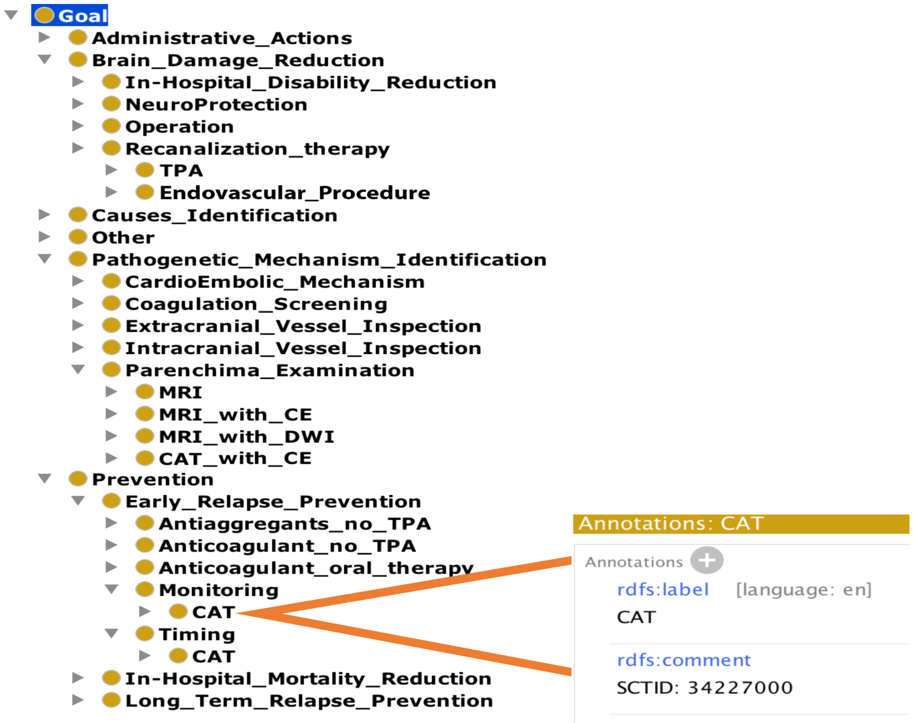
2 Knowledge-based trace abstraction

In our framework, trace abstraction has been realized as a multi-step mechanism. The following subsections describe the various steps.

2.1 Ontology mapping

As a first step, every action in the trace to be abstracted is mapped to a ground term of an **ontology**, formalized resorting to domain knowledge.

In our current implementation, we have defined an ontology related to the field of stroke management, where ground terms are patient management actions, while abstracted terms represent medical goals. Figure 1 shows an excerpt of the stroke domain ontology, formalized through the Protègè editor.



- Goal
  - Administrative_Actions
  - Brain_Damage_Reduction
    - In-Hospital_Disability_Reduction
    - NeuroProtection
    - Operation
    - Recanalization_therapy
      - TPA
      - Endovascular_Procedure
  - Causes_Identification
  - Other
  - Pathogenetic_Mechanism_Identification
    - CardioEmbolic_Mechanism
    - Coagulation_Screening
    - Extracranial_Vessel_Inspection
    - Intracranial_Vessel_Inspection
    - Parenchima_Examination
      - MRI
      - MRI_with_CE
      - MRI_with_DWI
      - CAT_with_CE
  - Prevention
    - Early_Relapse_Prevention
      - Antiaggregants_no_TPA
      - Anticoagulant_no_TPA
      - Anticoagulant_oral_therapy
      - Monitoring
        - CAT
      - Timing
        - CAT
    - In-Hospital_Mortality_Reduction
    - Long_Term_Relapse_Prevention

Annotations: CAT

Annotations

rdfs:label [language: en]

CAT

rdfs:comment

SCTID: 34227000

Fig. 1. An excerpt from the stroke domain ontology. The annotation shown for CAT (computer assisted tomography) reports its SNOMED code

In particular, a set of classes, representing the main goals in stroke management, have been identified, namely: “Administrative Actions” (such as hospital admission and discharge), “Brain Damage Reduction”,“Causes Identification”, “Pathogenetic Mechanism Identification”, “Prevention”, and “Other”. These main goals can be further specialized into subclasses, according to more specific goals (e.g., “Parenchima Examination” is a subgoal of “Pathogenetic Mechanism Identification”, while “Early Relapse Prevention” is a subgoal of “Prevention”), down to the ground actions, that will implement the goal itself.

Some actions in the ontology can be performed to implement different goals. For instance, a Computer Assisted Tomography (CAT) can be used to monitor therapy efficacy, or to assess therapy starting time (see class “Prevention” in figure 1). The proper goal to be used in the abstraction phase will be selected on the basis of the context of execution, as formalized in the rule base, described in the following subsection.

All ground actions are being mapped to SNOMED concepts[1]. The ontology partially showed in Figure 1 is therefore being integrated with the most comprehensive and precise clinical health terminology product in the world, accepted as a common global language for health terms. The figure, as an example, reports the CAT SNOMED code, which is an attribute of the CAT action.

2.2 Rule-based reasoning for ancestor selection

As a second step in the trace abstraction mechanism, a **rule base** is exploited to identify which of the multiple ancestors of an action in the ontology should be considered for abstracting the action itself. The rule base encodes medical knowledge. Contextual information (i.e., the actions that have been already executed on the patient at hand, and/or her/his specific clinical conditions) is used to activate the correct rules. The rule base has been formalized in Drools [20].

As an example, referring to the CAT action mentioned above, the rules reported below state that, if the patient has experienced a severe brain damage and suffers from atrial fibrillation, s/he must initiate a proper therapy. Such a therapy starts with ASA (a class of anti-inflammatory drugs), and continues with daily AC (anti-coagulant drug) administration. Before the first AC, a CAT is required, to assess AC starting time, which could be delayed in case CAT detects a hemorrhagic transformation. After a few days of AC administration, another CAT is needed, to monitor therapeutic results. Therefore, depending on the context, CAT can implement the "Timing" or the "Monitoring" goal (see Figure 1). Forward chaining on the rules below (showed as a simplified pseudocode with respect to the system internal representation, for the sake of simplicity) allows to determine the correct ancestor for the CAT action.

```
rule "SevereDamage"
    when
      (
         Damage(value > threshold) &&
         AtrialFibrillation(value=true)
       )
     then
        logicalInsertFact (DamFib);
end

rule "Fibrillation1"
    when
        existInLogical(DamFib) &&
        isBefore("CAT", "AC")
     then
        setAncestorName("CAT", "Timing");
end
```

[1] http://www.snomed.org/snomed-ct, last accessed on 02/08/2018

```
rule "Fibrillation2"
    when
        existInLogical(DamFib) &&
       isAfter("CAT", "AC")
     then
      setAncestorName("CAT", "Monitoring");
end
```

2.3 Trace abstraction

Once the correct ancestor of every action has been identified, trace abstraction can be completed.

In this last step, when a set of consecutive actions on the trace abstract as the same ancestor, they have to be merged into the same abstracted ***macro-action***, labeled as the common ancestor at hand. This procedure requires a proper treatment of *delays*, and of actions in-between that descend from a different ancestor (*interleaved actions* henceforth).

Trace abstraction has been realized by means of the procedure described in Algorithm 1 below.

The function *abstraction* takes in input a log *trace*, the domain ontology *onto*, and the *level* in the ontology chosen for the abstraction (e.g., $level = 1$ corresponds to the choice of abstracting the actions up to the sons of the ontology root). It also takes in input three thresholds (*delay_th*, *n_inter_th* and *inter_th*). These threshold values have to be set by the domain expert in order to limit the total admissible delay time within a macro-action, the total number of interleaved actions, and the total duration of interleaved actions, respectively. In fact, it would be hard to justify that two ground actions share the same goal (and can thus be abstracted to the same macro-action), if they are separated by very long delays, or if they are interleaved by many/long different ground actions, meant to fulfill different goals (where the term “long” may correspond to different quantitative values in different application domains).

The function outputs an abstracted trace.

For every action i in *trace*, an iteration is executed (lines 3-27). First, a macro-action m_i, initially containing just i, and sharing its starting and ending times, is created. m_i is labeled referring to the ancestor of i (the one identified by the rule based reasoning procedure) at the abstraction *level* provided as an input. Accumulators for this macro-action ($total - delay$, $num - inter$ and $total - inter$, commented below) are initialized to 0 (lines 4-10). Then, a nested cycle is executed (lines 11-25): it considers every element j following i in the trace, where a trace element can be an action, or a delay between a pair of consecutive actions. Different scenarios can occur:

- if j is a delay, $total - delay$ is updated by summing the length of j (lines 12-14).

Algorithm 1: Multi-level abstraction algorithm

```
$abs\_trace = abstraction(trace, onto, level, delay\_th, n\_inter\_th, inter\_th)$;
$abs\_trace = \emptyset$;
for $every\ i \in actions\ in\ trace$ do
    if $(i.startFlag = yes)$ then
        $create : m_i\ as\ ancestor(i, level)$;
        $m_i.start = i.start$;
        $m_i.end = i.end$;
        $total\_delay = 0$;
        $num\_inter = 0$;
        $total\_inter = 0$;
        for $(every\ j \in elements\ in\ trace)$ do
            if $(j\ is\ a\ delay)$ then
                $total\_delay = total\_delay + j.length$;
            else
                if $(ancestor(j, level){=}ancestor(i, level))$ then
                    if $(total\_delay < delay\_th \wedge num\_inter <$
                    $n\_inter\_th \wedge total\_inter < inter\_th)$ then
                        $m_i.end = max(m_i.end, j.end)$;
                        $j.startFlag = no$;
                    end
                else
                    $num\_inter = num\_inter + 1$;
                    $total\_inter = total\_inter + j.length$;
                end
            end
        end
    $append\ m_i\ to\ abs\_trace$;
end
return $abs\_trace$;
```

- if j is an action, and j shares the same ancestor of i at the input abstraction *level*, then j is incorporated into the macro-action m_i. This operation is always performed, provided that $total - delay$, $number - inter$ and $total - inter$ do not exceed the threshold passed as an input (lines 15-19). j is then removed from the actions in *trace* that could start a new macro-action, since it has already been incorporated into an existing one (line 18). This kind of situation is described in Figure 2 (a).
- if j is an action, but does not share the same ancestor of i, then it is treated as an interleaved action. In this case, $num - inter$ is increased by 1, and $total - inter$ is updated by summing the length of j (lines 20-23). This situation, in the end, may generate different types of temporal constraints between macro-actions, as the ones described in Figure 2 (b) (Allen's *during* [2]) and Figure 2 (c) (Allen's *overlaps* [2]).

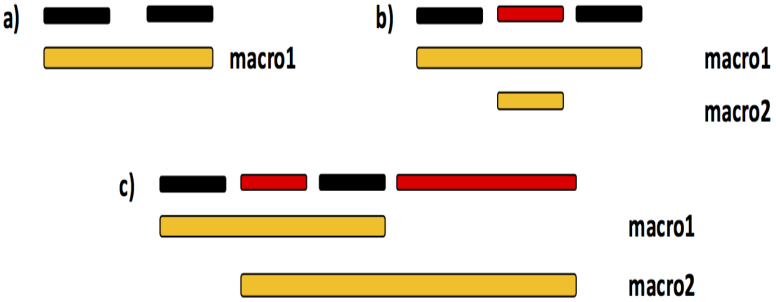


Fig. 2. Different trace abstraction situations: (a) two actions are abstracted to a single macro-action $macro1$, with a delay in between; (b) two actions are abstracted to a macro-action $macro1$, with an interleaved action in between, resulting in a different macro-action $macro2$ during $macro1$; (c) two actions are abstracted to a macro-action $macro1$, with an interleaved action in between, which is later aggregated to a fourth action, resulting in a macro-action $macro2$ overlapping $macro1$.

Finally, the macro-action m_i is appended to *abs_trace*, that, in the end, will contain the list of all the macro-actions that have been created by the procedure (line 26).

Complexity. The cost of abstracting a trace is $O(actions * elements)$, where *actions* is the number of actions in the input trace, and *elements* is the number of elements (i.e., actions + delay intervals) in the input trace.

3 Semantic process mining

In our approach, process mining, made semantic by the exploitation of the abstraction mechanism illustrated above, is implemented resorting to the well-known process mining tool ProM, extensively described in [32]. ProM (and specifically its newest version ProM 6) is a platform-independent open source framework that supports a wide variety of process mining and data mining techniques, and can be extended by adding new functionalities in the form of plug-ins.

For the work described in this paper, we have exploited ProM's Heuristic Miner [35]. Heuristic Miner is a plug-in for process model discovery, able to mine process models from logs. It receives in input the log, and considers the order of the actions within every single trace. It can mine the presence of short-distance and long-distance dependencies (i.e., direct or indirect sequence of actions), with a certain degree of reliability. The output of the mining process is provided as a graph, known as the "dependency graph", where nodes represent actions, and edges represent control flow information. The output can be converted into other formalisms as well.

Currently, we have chosen to rely on Heuristics Miner, because it is known to be tolerant to noise, a problem that may affect medical logs (e.g., sometimes the logging may be incomplete). Anyway, testing of other mining algorithms available in ProM 6 is foreseen in our future work.

4 Experimental results

In this section, we describe the experimental results we have conducted, in the application domain of stroke care.

First, we will present a case study, meant to showcase in an immediate fashion the effects of abstraction of the quality of the mined process model.

Then, we will discuss a more complete validation work, able to properly quantify the abstraction effects themselves.

In both studies the available log was composed of more than 15000 traces, collected at the Stroke Unit Network (SUN) collaborating centers of the Lombardia region, Italy. The number of traces varied from 266 to 1149. Traces were composed of 13 actions on average.

4.1 Case study

In the case study, we wanted to test whether our capability to abstract the log traces on the basis of their semantic goals allowed to obtained process models where unnecessary details are hidden, but key behaviors are clear. Indeed, if this hypothesis holds, in our application domain it becomes easier to compare process models of different stroke units (SUs), highlighting the presence/absence of common paths, regardless of minor action changes (e.g., different ground actions that share the same goal) or irrelevant different action ordering or interleaving (e.g., sets of ground actions, all sharing a common goal, that could be executed in any order)[2].

[2] It is however worth noting that, within our framework, it is still possible to mine the process models from ground traces, and investigate them in detail as a further analysis step, if needed.

Figure 3 compares the process models of two different SUs (SU-A and SU-B), mined by resorting to Heuristic Miner, operating on ground traces. Figure 4, on the other hand, compares the process models of the same SUs as figure 3, again mined by resorting to Heuristic Miner, but operating on traces abstracted according to the goals of the ontology in figure 1. In particular, abstraction was conducted up to level 2 in the ontology (where level 0 is the root, i.e.. "Goal").

Generally speaking, a visual inspection of the two graphs in figure 3 is very difficult. Indeed, these two ground processes are "spaghetti-like" [29], and the extremely large number of nodes and edges makes it hard to identify commonalities in the two models.

The abstract models in figure 4, on the other hand, are much more compact, and it is possible for a medical expert to analyze them.

In particular, the two graphs in figure 4 are not identical, but in both of them it is easy to a identify the macro-actions which corresponds to the treatment of a typical stroke patient.

However, the model for SU-A at the top of figure 4 exhibits a more complex control flow (with the presence of loops), and shows three additional macro-actions with respect to the model of SU-B, namely "Extracranial Vessel Inspection", "Intracranial Vessel Inspection" and "Recanalization". This finding can be explained, since SU-A is a better equipped SU, where different kinds of patients, including some atypical/more critical ones, can be managed, thanks to the availability of different skills and instrumental resources. These patients may require the additional macro-actions reported in the model, and/or the repetition of some procedures, in order to better characterize and manage the individual patient's situation.

On the other hand, SU-B is a more generalist SU, where very specific human knowledge or technical resources are missing. As a consequence, the overall model control flow is simpler, and some activities are not executed at all.

Interestingly, our abstraction mechanism, while hiding irrelevant details, allows to still appreciate these differences.

4.2 Validation

For the validation study, we asked a SUN stroke management expert to provide a ranking of some SUN stroke units (see Table 1, column 2), on the basis of the quality of service they provide, with respect to the top level SU (referred as H0 in the experiments). Such a ranking was based on her personal knowledge of the SUs human and instrumental resource availability (not on the process models); therefore, it was qualitative, and *coarse-grained*, in the sense that more than one SU could obtain the same qualitative evaluation. The expert identified 6 SUs (H1-H6) with a high similarity level with respect to H0; 5 SUs (H7-H11) with a medium similarity level with respect to H0; and 4 SUs (H12-H15) with a low similarity level with respect to H0. The ordering of the SUs within one specific similarity level is not very relevant, since, as observed, the expert's ranking is coarse-grained. It is instead important to distinguish between different similarity levels.

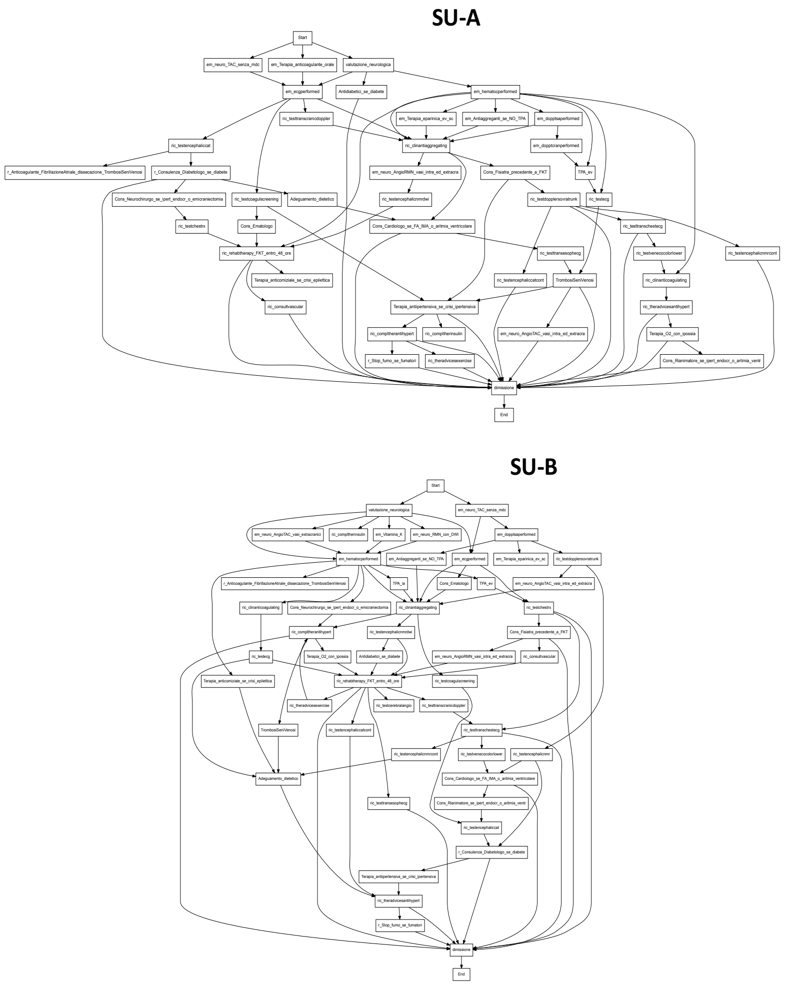

SU-A
Start
em_neuro_TAC_senza_mdc
em_Terapia_anticoagulante_orale
valutazione_neurologica
em_ecgperformed
Antidiabetici_se_diabete
em_hematocperformed
ric_testtranscranicdoppler
em_Terapia_eparinica_ev_sc
em_Antiaggreganti_se_NO_TPA
em_dopptsaperformed
ric_testencephaliccat
ric_clinantiaggregating
em_dopptcranperformed
r_Anticoagulante_FibrillazioneAtriale_dissecazione_TrombosiSeniVenosi
r_Consulenza_Diabetologo_se_diabete
em_neuro_AngioRMN_vasi_intra_ed_extracra
Cons_Fisiatra_precedente_a_FKT
TPA_ev
Cons_Neurochirurgo_se_ipert_endocr_o_emicraniectomia
ric_testcoagulscreening
Adeguamento_dietetico
ric_testencephalicnmrdwi
ric_testdopplersovratrunk
ric_testecg
ric_testchestrx
Cons_Ematologo
Cons_Cardiologo_se_FA_IMA_o_aritmia_ventricolare
ric_testtranschestecg
ric_rehabtherapy_FKT_entro_48_ore
ric_testtransesophecg
ric_testvenecocolorlower
ric_testencephalicnmrcont
Terapia_anticomiziale_se_crisi_epilettica
ric_testencephaliccatcont
TrombosiSeniVenosi
ric_clinanticoagulating
ric_consultvascular
Terapia_antiipertensiva_se_crisi_ipertensiva
ric_theradvicesantihypert
ric_compltherantihypert
ric_compltherinsulin
em_neuro_AngioTAC_vasi_intra_ed_extracra
Terapia_O2_con_ipossia
r_Stop_fumo_se_fumatori
ric_theradvicesexercise
Cons_Rianimatore_se_ipert_endocr_o_aritmia_ventr
dimissione
End
SU-B
Start
valutazione_neurologica
em_neuro_TAC_senza_mdc
em_neuro_AngioTAC_vasi_extracranici
ric_compltherinsulin
em_Vitamina_K
em_neuro_RMN_con_DWI
em_dopptsaperformed
em_hematocperformed
em_Antiaggreganti_se_NO_TPA
em_ecgperformed
em_Terapia_eparinica_ev_sc
ric_testdopplersovratrunk
r_Anticoagulante_FibrillazioneAtriale_dissecazione_TrombosiSeniVenosi
TPA_ia
Cons_Ematologo
TPA_ev
em_neuro_AngioTAC_vasi_intra_ed_extracra
ric_clinanticoagulating
Cons_Neurochirurgo_se_ipert_endocr_o_emicraniectomia
ric_clinantiaggregating
ric_testchestrx
ric_compltherantihypert
ric_testencephalicnmrdwi
Cons_Fisiatra_precedente_a_FKT
ric_testecg
Terapia_O2_con_ipossia
Antidiabetici_se_diabete
em_neuro_AngioRMN_vasi_intra_ed_extracra
ric_consultvascular
Terapia_anticomiziale_se_crisi_epilettica
ric_rehabtherapy_FKT_entro_48_ore
ric_testcoagulscreening
ric_theradvicesexercise
ric_testcerebralangio
ric_testtranscranicdoppler
TrombosiSeniVenosi
ric_testencephaliccatcont
ric_testtranschestecg
ric_testencephalicnmrcont
ric_testvenecocolorlower
ric_testencephalicnmr
Adeguamento_dietetico
Cons_Cardiologo_se_FA_IMA_o_aritmia_ventricolare
ric_testtransesophecg
Cons_Rianimatore_se_ipert_endocr_o_aritmia_ventr
ric_testencephaliccat
r_Consulenza_Diabetologo_se_diabete
Terapia_antiipertensiva_se_crisi_ipertensiva
ric_theradvicesantihypert
r_Stop_fumo_se_fumatori
dimissione
End

Fig. 3. Comparison between two process models, mined by resorting to Heuristic Miner, operating on ground traces. The figure is not intended to be readable, but only to give an idea of how complex the models can be

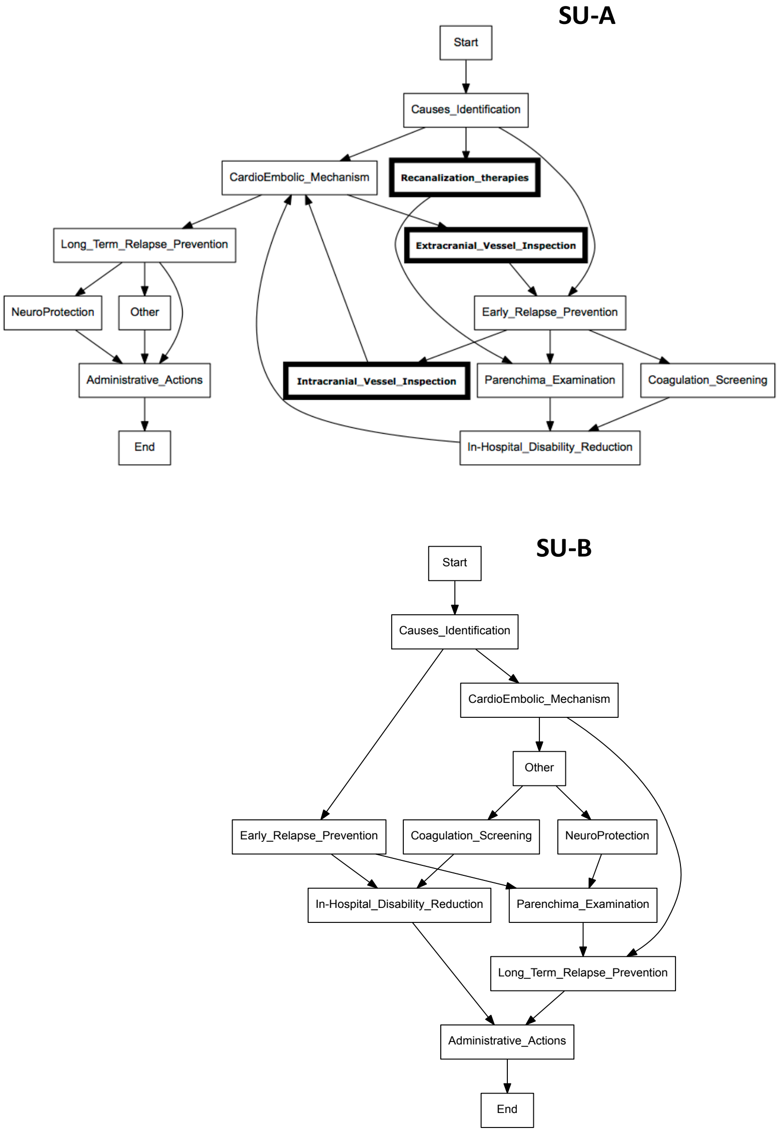


Fig. 4. Comparison between the two process models of the same SUs as figure 3, mined on abstracted traces. Additional macro-actions executed at SU-A are highlighted in bold

We mined the process models of the 16 SUs by resorting to Heuristic Miner, both working on ground traces, and working on abstracted traces. We then ordered the two available process model sets with respect to H0, resorting to two different process model distances, i.e., the one described in [11], and the one presented in [22], globally obtaining four rankings.

These two distance measures are quite dissimilar. In particular, the distance in [11] provides a normalized version of the graph edit distance [5] for comparing business process models, and defines syntactical edit operation costs for node substitution (relying on edit distance between action names), node insertion/deletion, and edge insertion/deletion. On the other hand, the distance in [22] moves towards a more semantic and knowledge-intensive approach: when actions are represented in an ontology, as it is our case, it can adopt Palmer's distance [23] for calculating action node substitution costs. Palmer's distance between two actions is set to the normalized number of arcs on the path between the two actions themselves in the ontology. So, given the semantics of our ontology, two different actions can be considered as more or less distant on the basis of their goal. Moreover, the distance in [22] also considers edge substitution, which is disregarded in [11]. Indeed, Heuristic Miner labels edges with information that can be relevant in graph comparison, such as reliability [35], while statistical temporal information can be easily extracted from the log and saved as arc properties.

We exploited both these metrics because we wished to verify the effect of trace abstraction independently of the simplicity or completeness of the distance definition.

Results are shown in table 1.

Column 1 in table 1 shows the levels of similarity with respect to the reference SU. Column 2 shows the ranking according to the human medical expert; columns 3 and 4 show the ranking obtained by relying on the distance in [11], mining the process models on ground and abstracted traces, respectively. Similarly, columns 5 and 6 show the results obtained by relying on the distance in [22], mining the process models on ground and abstracted traces, respectively. In particular, abstraction was conducted at level 1 in the ontology (where level 0 is the root).

When working on ground traces, the distance in [11] correctly rates two process models in the high similarity group (33%), one process model in the medium similarity group (20%), and one process model in the low similarity group (25%, column 3). When working on abstracted traces, on the other hand, the distance in [11] correctly rates three process models in the high similarity group (50%), one process models in the medium similarity group (20%), and one process model in the low similarity group (25%, column 4).

When working on ground traces, the distance in [22] correctly rates two process models in the high similarity group (33%), zero process model in the medium similarity group (0%), and one process model in the low similarity group (25%, column 5). When working on abstracted traces, on the other hand, the distance in [22] correctly rates four process models in the high similarity group (66%), two process models in the medium similarity group (40%), and two process model in the low similarity group (50%, column 6).

In summary, when working on abstracted traces, both distances lead to rankings that are closer to the qualitative ranking provided by the human expert, but the improvement is larger when adopting the distance in [22], probably because it is able to take into account semantic information. We plan to verify this statement by means of further experiments in the future.

For the sake of completeness, we have also tested the effect of abstraction by evaluating the capability of the two distances in locating the high level SUs (i.e., the SUs with a high similarity with respect to H0) as the first six items of the overall ranking. To this end, we calculated the $nDCG_6$[7] index on the rankings of Table 1. Also in this experiment, we were able to verify that the use of abstraction leads to results that are closer to the (ideal) ones provided by the human expert (see Table 2), with a larger improvement when adopting the distance in [22].

Table 1. Ordering of 15 SUs, with respect to a given query model. Correct positions in the rankings with respect to the expert's qualitative similarity levels are highlighted in bold.

Similarity	**Medical expert**	**[11] ground**	**[11] abs**	**[22] ground**	**[22] abs**
High	H1	H15	**H6**	**H3**	**H1**
High	H2	H8	H13	**H2**	**H6**
High	H3	**H5**	**H2**	H15	**H5**
High	H4	**H6**	**H4**	H10	**H3**
High	H5	H12	H10	H8	H8
High	H6	H11	H8	H11	H10
Medium	H7	H4	H3	H4	**H9**
Medium	H8	H3	H5	H12	H4
Medium	H9	H14	**H7**	H6	H14
Medium	H10	H2	H14	H1	**H7**
Medium	H11	**H10**	H15	H13	H12
Low	H12	**H13**	H9	H9	**H13**
Low	H13	H9	H1	**H14**	H2
Low	H14	H1	H11	H7	H11
Low	H15	H7	**H12**	H5	**H15**

Table 2. $nDCG_6$ index calculation on the rankings provided by the two metrics.

[11] ground	**[11] abs**	**[22] ground**	**[22] abs**
0.621	0.798	0.781	0.925

Table 3, on the other hand, reports our results on the calculation of *fitness* [29] on the process models mined for our 16 SUs, at different levels of abstraction. Fitness evaluates whether a process model is able to reproduce all execution sequences that are in the log. If the log can be replayed correctly, fitness evaluates to 1. In the worst case, it evaluates to 0. Fitness calculation is available in ProM.

Table 3. Fitness values calculated on the mined process models, when operating at different levels of abstraction

SU	**Ground**	**Abs. level 2**	**Abs. level 1**
H0	0.58	0.83	0.87
H1	0.43	0.73	0.97
H2	0.47	0.72	0.87
H3	0.40	0.85	0.95
H4	0.42	0.91	0.93
H5	0.42	0.83	0.87
H6	0.52	0.76	0.95
H7	0.52	0.79	0.92
H8	0.75	0.78	0.92
H9	0.46	0.85	0.92
H10	0.41	0.83	0.87
H11	0.44	0.61	0.90
H12	0.49	0.67	0.89
H13	0.46	0.84	0.91
H14	0.43	0.85	0.90
H15	0.54	0.78	0.90

As it can be observed from the table, the more the traces are abstracted, the more the fitness values increase in the corresponding mined models.

In conclusion, our abstraction mechanism, while hiding irrelevant details, allows to still appreciate relevant differences between models, such as, e.g., the presence/absence of important actions, as commented in section 4.1. Moreover, when working on abstracted traces, the adoption of different distance definitions leads to SU rankings that are closer to the qualitative ranking provided by the human expert. Finally, very interestingly, abstraction proves to be a means to significantly increase the quality of the mined models, measured in terms of fitness, which is a well known and largely adopted indicator.

5 Related works

The use of semantics in business process management, with the aim of operating at different levels of abstractions in process discovery and/or analysis, is a relatively young area of research, where much is still unexplored.

One of the first contributions in this field was proposed in [6], which introduces a process data warehouse, where taxonomies are exploited to add semantics to process execution data, in order to provide more intelligent reports. The work in [14] extends the one in [6], presenting a complete architecture that allows business analysts to perform multidimensional analysis and classify process instances, according to flat taxonomies (i.e., taxonomies without subsumption relations between concepts).

Hepp et al. [17] propose a framework able to merge semantic web, semantic web services, and business process management techniques to build semantic business process management, and use ontologies to provide machine-processable semantics in business processes [18]. The work in [26] develops in a similar context, and extends OLAP tools with semantics (exploiting ontologies rather than (flat) taxonomies).

The topic was studied in the SUPER project [25], within which several ontologies were created, such as the process mining ontology and the event ontology [24]; these ontologies define core terminologies of business process management, usable by machines for task automation. However, the authors did not present any concrete implementations of semantic process mining or analysis.

Ontologies, references from elements in logs to concepts in ontologies, and ontology reasoners (able to derive, e.g., concept equivalence), are described as the three essential building blocks for semantic process mining in [10]. This paper also shows how to use these building blocks to extend ProM's LTL Checker [31] to perform semantic auditing of logs.

The work in [8] focuses on the use of semantics in business process monitoring, an activity that allows to detect or predict process deviations and special situations, to diagnose their causes, and possibly to resolve problems by applying corrective actions. Detection, diagnosis and resolution present interesting challenges that, on the authors' opinion, can strongly benefit from knowledge-based techniques.

In [8, 9] the idea to explicitly relate (or annotate) elements in the log with the concepts they represent, linking these elements to concepts in ontologies, is addressed. This "semantic lifting" approach, to use a term borrowed from the Web scenario, is also investigated in [3], to discover the process that is actually being executed.

In [9] an example of process discovery at different levels of abstractions is presented. It is however a very simple example, where a couple of ground actions are abstracted according to their common ancestor. However, the management of interleaved actions or delays is not addressed, and multiple inheritance is not considered. A more recent work [19] introduces a methodology that combines domain and company-specific ontologies and databases to obtain multiple levels of abstraction for process mining. In this paper data in databases become instances of concepts at the bottom level of a taxonomy tree structure. If consecutive tasks in the discovered model abstract as the same concepts, those tasks are aggregated. However, also in this work we could find neither a clear description of the abstraction algorithm, nor the management of interleaved actions or delays.

Moreover, most of the papers cited above (including [10, 9]) present theoretical frameworks, and not yet a detailed technical architecture nor a concrete implementation of all their ideas.

Referring to medical applications, the work in [13] proposes an approach, based on semantic process mining, to verify the compliance of a Computer Interpretable Guideline with medical recommendations. In this case, semantic process mining refers to conformance checking rather than to process discovery (as it is also the case in [10]). These works are thus only loosely related to our contribution.

Besides the contributions listed above, most of which can be categorized as normative/deductive approaches, it is worth citing some interesting emergent approaches as well [27, 4]. The work in [27] affords the problem of dealing with very big log files, proposing solutions for scalable process discovery and extending XES [34], a log file format introduced to solve problems with the semantics of additional attributes; the work in [4] moves even forward, presenting a methodology designed to implement consistent process mining algorithms in a Big Data context, managing semantic heterogeneity.

Another interesting research direction adopts abstraction as a way to establish a relationship between the events typically recorded in the log by the information system, and the high-level actions which are of interest when mining business process models. In particular the work in [12] looks for mappings between events and actions. The set of possible mappings can be large, and one should focus on the mapping which has the highest coverage, meaning that the chosen mapping should be applicable to the largest possible number of traces in the log. The range of a mapping, i.e., the set of actions that appear in the mapping, is also a very important factor. In particular, the authors adopt a greedy approach, where they try to merge those mappings which, by themselves, already have the largest range and highest coverage among their peers. The work in[28], on the other hand, shows that supervised learning (namely Condition Random Fields) can be leveraged for the event abstraction task when annotations with high-level interpretations of the low-level events are available for a subset of the traces. Conditional Random Fields are trained on the annotated traces to create a probabilistic mapping from low-level events to high-level actions. This mapping, once obtained, can be applied to the unannotated traces as well. In [21], the authors align the behavior defined by activity patterns with the observed behavior in the log. Activity patterns encode assumptions on how high-level actions manifest themselves in terms of recorded low-level events. The work in [16] exploits trace segmentation. Trace segmentation is based on the idea that subsequences of events, which are supposed to be the product of a higher-level action, are identified. From the co-occurrence of events in the log, the authors derive the relative correlation between their event classes. All event classes found in the log are successively combined into clusters, representing higher-level types. In this hierarchy of event classes, an arbitrary level of abstraction can be chosen to process the log.

Most of these constributions (see, e.g., [12, 16]), however, adopt non-semantic approaches. Moreover, as stated in the Introduction, in our work we assume that traces are sequences of actions (i.e., with respect to event traces, they have already been pre-processed). Therefore, this line of research is only loosely related to ours.

In conclusion, in the current research panorama, our work appears to be very innovative, for several reasons:

- many approaches, illustrating very interesting and sometimes ambitious ideas, just provide pure theoretical frameworks, which can be very important to inspire more engineering-style work. However, concrete implementations of algorithms and complete architectures of systems are often missing, leaving open research opportunities for contributions like the one we have presented;
- in semantic process mining, more work has been done in the field of conformance checking (also in medical applications), while process discovery still deserves attention (also because many approaches are still at the theoretical level, as commented above);
- as regards trace abstraction, it is often proposed as a very powerful means to obtain better process discovery and analysis results, but technical details of the abstraction mechanism are usually not provided, or are illustrated through very simple examples, where the issues related to the management of interleaved actions or delays do not emerge.

6 Concluding remarks and future work

In this paper, we have presented a framework for knowledge-based abstraction of log traces. In our approach, abstracted traces are then provided as an input to semantic process mining. Semantic process mining relies on ProM algorithms; indeed, the overall integration of our approach within ProM is foreseen in our future work.

Our case study in the field of stroke management suggests that the capability of abstracting the log traces on the basis of their semantic goal allows to mine clearer process models, where unnecessary details are hidden, but key behaviors are clear.

Moreover, in our validation study we mined the process models of some SUs by resorting to Heuristic Miner, both working on ground traces, and working on abstracted traces. We then ordered the two available process model sets with respect to the model of the best equipped SU in the SUN network, resorting to two different process model distances, i.e., the one described in [11], and the one presented in [22], globally obtaining four rankings. We verified that, when working on abstracted traces, both distances lead to rankings that are closer to the qualitative ranking provided by a domain expert, but the improvement is much larger when adopting the distance in [22], probably because it is able to take into account semantic information. Finally, abstraction proves to be a means to significantly increase the quality of the mined models, when measured in terms of fitness.

In the future, we would like to test the approach in different application domains as well, even if we know that the task will require time and may present some issues. Indeed, domain knowledge acquisition is typically time consuming: our work on formalizing the stroke ontology and the rules required multiple sessions of work with physicians. Moreover, the domain of stroke management can count on internationally recognized guidelines, which are well detailed: therefore, the acquired knowledge can be considered as comprehensive and accurate; this may not hold in different domains. The domain choice will therefore be critical, and possibly we will select a non medical application.

We also plan to apply the abstraction mechanism to subsets of the log, obtained by clustering techniques or possibly by domain expert rating, in order to test the effect of abstraction on a more homogeneous input.

The quality of the mined process models could also be evaluated by resorting to conformance checking techniques.

We will also consider how to cobine our approach with the pattern-based abstraction described in [21].

Finally, an abstraction mechanism directly operating on process models (i.e., on the graph, instead of the log), may be considered, possibly along the lines described in [15], and abstraction results will be compared to the ones currently enabled by our framework.

References

1. E. Rolón Aguilar, F. Ruiz, F. García, and M. Piattini. Evaluation measures for business process models. In H. Haddad, editor, *Proceedings of the 2006 ACM Symposium on Applied Computing (SAC), Dijon, France, April 23-27, 2006*, pages 1567–1568. ACM, 2006.
2. J.F. Allen. Towards a general theory of action and time. *Artificial Intelligence*, 23:123–154, 1984.
3. A. Azzini, C. Braghin, E. Damiani, and F. Zavatarelli. Using semantic lifting for improving process mining: a data loss prevention system case study. In R. Accorsi, P. Ceravolo, and P. Cudré-Mauroux, editors, *Proceedings of the 3rd International Symposium on Data-driven Process Discovery and Analysis*, volume 1027 of *CEUR Workshop Proceedings*, pages 62–73. CEUR-WS.org, 2013.
4. A. Azzini and P. Ceravolo. Consistent process mining over big data triple stores. In *IEEE International Congress on Big Data, BigData Congress 2013*, pages 54–61. IEEE Computer Society, 2013.
5. H. Bunke. On a relation between graph edit distance and maximum common subgraph. *Pattern Recognition Letters*, 18(8):689694, 1997.
6. F. Casati and M.C. Shan. Semantic analysis of business process executions. In C. S. Jensen, K. G. Jeffery, J. Pokorný, S. Saltenis, E. Bertino, K. Böhm, and M. Jarke, editors, *Advances in Database Technology - EDBT 2002, 8th International Conference on Extending Database Technology, Prague, Czech Republic, March 25-27, Proceedings*, volume 2287 of *Lecture Notes in Computer Science*, pages 287–296. Springer, 2002.
7. W.B. Croft, D. Metzler, and T. Strohman. *Search Engines: Information Retrieval in Practice*. Alternative Etext Formats. Addison-Wesley, 2010.

8. A. K. Alves de Medeiros, C. Pedrinaci, W. M. P. van der Aalst, J. Domingue, M. Song, A. Rozinat, B. Norton, and L. Cabral. An outlook on semantic business process mining and monitoring. In R. Meersman, Z. Tari, and P. Herrero, editors, *On the Move to Meaningful Internet Systems 2007: OTM 2007 Workshops, OTM Confederated International Workshops and Posters, AWeSOMe, CAMS, OTM Academy Doctoral Consortium, MONET, OnToContent, ORM, PerSys, PPN, RDDS, SSWS, and SWWS 2007, Vilamoura, Portugal, November 25-30, 2007, Proceedings, Part II*, volume 4806 of *Lecture Notes in Computer Science*, pages 1244–1255. Springer, 2007.
9. A. K. Alves de Medeiros and W. M. P. van der Aalst. Process mining towards semantics. In T. S. Dillon, E. Chang, R. Meersman, and K. P. Sycara, editors, *Advances in Web Semantics I - Ontologies, Web Services and Applied Semantic Web*, volume 4891 of *Lecture Notes in Computer Science*, pages 35–80. Springer, 2009.
10. A. K. Alves de Medeiros, W. M. P. van der Aalst, and C. Pedrinaci. Semantic process mining tools: Core building blocks. In W. Golden, T. Acton, K. Conboy, H. van der Heijden, and V. K. Tuunainen, editors, *16th European Conference on Information Systems, ECIS 2008, Galway, Ireland, 2008*, pages 1953–1964, 2008.
11. R. Dijkman, M. Dumas, and R. Garca-Banuelos. Graph matching algorithms for business process model similarity search. In U. Dayal, J. Eder, J. Koehler, and H. Reijers, editors, *Proc. International Conference on Business Process Management*, volume 5701 of *Lecture Notes in Computer Science*, pages 48–63. Springer, Berlin, 2009.
12. D. R. Ferreira, F. Szimanski, and C. Ghedini Ralha. Improving process models by mining mappings of low-level events to high-level activities. *J. Intell. Inf. Syst.*, 43(2):379–407, 2014.
13. M. A. Grando, M. H. Schonenberg, and W. M. P. van der Aalst. Semantic process mining for the verification of medical recommendations. In V. Traver, A. L. N. Fred, J. Filipe, and H. Gamboa, editors, *HEALTHINF 2011 - Proceedings of the International Conference on Health Informatics, Rome, Italy, 26-29 January, 2011*, pages 5–16. SciTePress, 2011.
14. D. Grigori, F. Casati, M. Castellanos, U. Dayal, M. Sayal, and M.C. Shan. Business process intelligence. *Computers in Industry*, 53(3):321–343, 2004.
15. C. Günther and W. van der Aalst. Fuzzy mining - adaptive process simplification based on multi-perspective metrics. In G. Alonso, P. Dadam, and M. Rosemann, editors, *Business Process Management, 5th International Conference, BPM 2007, Brisbane, Australia, September 24-28, 2007, Proceedings*, volume 4714 of *Lecture Notes in Computer Science*, pages 328–343. Springer, 2007.
16. C. W. Günther, A. Rozinat, and W. van der Aalst. Activity mining by global trace segmentation. In S. Rinderle-Ma, S. W. Sadiq, and F. Leymann, editors, *Business Process Management Workshops, BPM 2009 International Workshops, Ulm, Germany, September 7, 2009. Revised Papers*, volume 43 of *Lecture Notes in Business Information Processing*, pages 128–139. Springer, 2010.
17. 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 F. C. M. Lau, H. Lei, X. Meng, and M. Wang, editors, *2005 IEEE International Conference on e-Business Engineering (ICEBE 2005), 18-21 October 2005, Beijing, China*, pages 535–540. IEEE Computer Society, 2005.
18. M. Hepp and D. Roman. An ontology framework for semantic business process management. In A. Oberweis, C. Weinhardt, H. Gimpel, A. Koschmider,

V. Pankratius, and B. Schnizler, editors, *eOrganisation: Service-, Prozess-, Market-Engineering: 8. Internationale Tagung Wirtschaftsinformatik - Band 1, WI 2007, Karlsruhe, Germany, February 28 - March 2, 2007*, pages 423–440. Universitaetsverlag Karlsruhe, 2007.

19. W. Jareevongpiboon and P. Janecek. Ontological approach to enhance results of business process mining and analysis. *Business Proc. Manag. Journal*, 19(3):459–476, 2013.
20. M. N. De Maio M. Salatino, E. Aliverti. *Mastering JBoss Drools 6 for Developers*. Packt Publishing, 2016.
21. F. Mannhardt, M. de Leoni, H. A. Reijers, W. van der Aalst, and P. J. Toussaint. From low-level events to activities - A pattern-based approach. In M. La Rosa, P. Loos, and Oscar Pastor, editors, *Business Process Management - 14th International Conference, BPM 2016, Rio de Janeiro, Brazil, September 18-22, 2016. Proceedings*, volume 9850 of *Lecture Notes in Computer Science*, pages 125–141. Springer, 2016.
22. S. Montani, G. Leonardi, S. Quaglini, A. Cavallini, and G. Micieli. Improving structural medical process comparison by exploiting domain knowledge and mined information. *Artificial Intelligence in Medicine*, 62(1):33–45, 2014.
23. M. Palmer and Z. Wu. Verb Semantics for English-Chinese Translation. *Machine Translation*, 10:59–92, 1995.
24. C. Pedrinaci and J. Domingue. Towards an ontology for process monitoring and mining. In M. Hepp, K. Hinkelmann, D. Karagiannis, R. Klein, and N. Stojanovic, editors, *Proceedings of the Workshop on Semantic Business Process and Product Lifecycle Management SBPM 2007, held in conjunction with the 3rd European Semantic Web Conference (ESWC 2007), Innsbruck, Austria, June 7, 2007*, volume 251 of *CEUR Workshop Proceedings*, 2007.
25. C. Pedrinaci, J. Domingue, C. Brelage, T. van Lessen, D. Karastoyanova, and F. Leymann. Semantic business process management: Scaling up the management of business processes. In *Proceedings of the 2th IEEE International Conference on Semantic Computing (ICSC 2008), August 4-7, 2008, Santa Clara, California, USA*, pages 546–553. IEEE Computer Society, 2008.
26. D. Sell, L. Cabral, E. Motta, J. Domingue, and R. C. dos Santos Pacheco. Adding semantics to business intelligence. In *16th International Workshop on Database and Expert Systems Applications (DEXA 2005), 22-26 August 2005, Copenhagen, Denmark*, pages 543–547. IEEE Computer Society, 2005.
27. A. Syamsiyah, B. van Dongen, and W. van der Aalst. DB-XES: enabling process discovery in the large. In *Proceedings of the 6th International Symposium on Data-driven Process Discovery and Analysis (SIMPDA*.
28. N. Tax, N. Sidorova, R. Haakma, and w. van der Aalst. Event abstraction for process mining using supervised learning techniques. *CoRR*, abs/1606.07283, 2016.
29. W. van der Aalst. *Process Mining. Data Science in Action.* Springer, 2016.
30. W. van der Aalst, B. van Dongen, J. Herbst, L. Maruster, G. Schimm, and A. Weijters. Workflow mining: a survey of issues and approaches. *Data and Knowledge Engineering*, 47:237–267, 2003.
31. W. M. P. van der Aalst, H. T. de Beer, and B. F. van Dongen. Process mining and verification of properties: An approach based on temporal logic. In R. Meersman, Z. Tari, M. Hacid, J. Mylopoulos, B. Pernici, Ö. Babaoglu, H. Jacobsen, J. P. Loyall, M. Kifer, and S. Spaccapietra, editors, *On the Move to Meaningful Internet Systems 2005: CoopIS, DOA, and ODBASE, OTM Confederated International Conferences CoopIS, DOA, and ODBASE 2005, Agia Napa, Cyprus, October 31 -*

November 4, 2005, Proceedings, Part I, volume 3760 of *Lecture Notes in Computer Science*, pages 130–147. Springer, 2005.

32. B. van Dongen, A. Alves De Medeiros, H. Verbeek, A. Weijters, and W. van der Aalst. The proM framework: a new era in process mining tool support. In G. Ciardo and P. Darondeau, editors, *Knowledge Mangement and its Integrative Elements*, pages 444–454. Springer, Berlin, 2005.
33. I. T. P. Vanderfeesten, H. A. Reijers, J. Mendling, W. M. P. van der Aalst, and J. S. Cardoso. On a quest for good process models: The cross-connectivity metric. In Z. Bellahsene and M. Léonard, editors, *Advanced Information Systems Engineering, 20th International Conference, CAiSE 2008, Montpellier, France, June 16-20, 2008, Proceedings*, volume 5074 of *Lecture Notes in Computer Science*, pages 480–494. Springer, 2008.
34. H. M. W. Verbeek, J. C. A. M. Buijs, B. F. van Dongen, and W. M. P. van der Aalst. Xes, xesame, and prom 6. In P. Soffer and E. Proper, editors, *Information Systems Evolution - CAiSE Forum 2010, Hammamet, Tunisia, June 7-9, 2010, Selected Extended Papers*, volume 72 of *Lecture Notes in Business Information Processing*, pages 60–75. Springer, 2011.
35. A. Weijters, W. van der Aalst, and A. Alves de Medeiros. *Process Mining with the Heuristic Miner Algorithm, WP 166*. Eindhoven University of Technology, Eindhoven, 2006.