A knowledge-intensive approach to process similarity calculation

Stefania Montani a, Giorgio Leonardi a,⇑, Silvana Quaglini b, Anna Cavallini c, Giuseppe Micieli c on behalf of the Stroke Unit Network (SUN) collaborating centers
a DISIT, Computer Science Institute, Università del Piemonte Orientale, Viale T. Michel 11, 15121 Alessandria, Italy b Department of Electrical, Computer and Biomedical Engineering, Università di Pavia, Via Ferrata 1, I-27100 Pavia, Italy c Istituto di Ricovero e Cura a Carattere Scientifico Fondazione ‘‘C. Mondino’’, Via Mondino 2, I-27100 Pavia, Italy

Article history:
Available online 24 January 2015

Keywords:
Process comparison
Graph edit distance
Process mining
Stroke management Process model comparison and similar processes retrieval are key issues to be addressed in many real world situations, and particularly relevant ones in some applications (e.g., in medicine), where similarity quantification can be exploited in a quality assessment perspective.
Most of the process comparison techniques described in the literature suffer from two main limitations: (1) they adopt a purely syntactic (vs. semantic) approach in process activity comparison, and/ or (2) they ignore complex control flow information (i.e., other than sequence). These limitations oversimplify the problem, and make the results of similarity-based process retrieval less reliable, especially when domain knowledge is available, and can be adopted to quantify activity or control flow construct differences.
In this paper, we aim at overcoming both limitations, by introducing a framework which allows to extract the actual process model from the available process execution traces, through process mining techniques, and then to compare (mined) process models, by relying on a novel distance measure.
The novel distance measure, which represents the main contribution of this paper, is able to address issues (1) and (2) above, since: (1) it provides a semantic, knowledge-intensive approach to process activity comparison, by making use of domain knowledge; (2) it explicitly takes into account complex control flow constructs (such as AND and XOR splits/joins), thus fully considering the different semantic meaning of control flow connections in a reliable way.
The positive impact of the framework in practice has been tested in stroke management, where our approach has outperformed a state-of-the art literature metric on a real world event log, providing results that were closer to those of a human expert. Experiments in other domains are foreseen in the future.

1. Introduction

Corresponding author. Tel.: +39 0131 360340.
E-mail addresses: stefania.montani@unipmn.it (S. Montani), giorgio.leonardi@ unipmn.it (G. Leonardi), silvana.quaglini@unipv.it (S. Quaglini), anna.cavallini@ mondino.it (A. Cavallini), giuseppe.micieli@mondino.it (G. Micieli).

Process model comparison is a key issue to be addressed in many real world situations. For example, when two companies are merged, process engineers need to compare processes originating from the two companies, in order to analyze their possible overlaps, and to identify areas for consolidation. Moreover, large companies build over time huge process model repositories, which serve as a knowledge base for their ongoing process management/ enhancement efforts. Before adding a new process model to the repository, process engineers have to check that a similar model does not already exist, in order to prevent duplication.
Particularly interesting is the case of medical process model comparison, where similarity quantification can be exploited in a quality assessment perspective. Indeed, the process model actually implemented at a given healthcare organization can be compared to the existing reference clinical guideline, e.g., to check conformance, or to understand the level of adaptation to local constraints that may have been required. As a matter of fact, the existence of local resource constraints may lead to differences between the models implemented at different hospitals, even when referring to the treatment of the same disease (and to the same guideline). A quantification of these differences (and maybe a ranking of the hospitals derived from it) can be exploited for several purposes, like, e.g., auditing purposes, performance evaluation and funding distribution.
Various process model comparison techniques are described in the literature (see Section 4). However, most of them suffer from two main limitations:

1. they adopt a purely syntactic approach in process activity comparison, ignoring the semantics of the activities being compared, often referring just to their names: activities with a different name are considered as not matching, while they could share very similar characteristics (e.g., have the same goal);
2. they ignore complex control flow information (other thansequence): in this way, a construct with, e.g., two parallel activities, can be matched to a construct involving the same activities, but in mutual exclusion.

Issues (1) and (2) above correspond to a strong simplification of the process model semantic meaning, and may lead to unreliable results in process comparison. This can be really unacceptable in many real world domains, like the already mentioned medical ones, where physicians and hospital managers need to guarantee the highest quality of service to patients.
In this paper, we aim at overcoming the limitations outlined above, by introducing a framework which allows to mine the actual process model from the available process execution traces, and then to compare (mined) process models.
While the framework, in its current version, relies on already published process mining techniques to extract the process model from traces, process comparison exploits a novel distance measure, which represents the main contribution of the paper.
Our distance measure is very innovative with respect to available literature approaches (see detailed discussion in Section 4). Indeed, it is able to address issues (1) and (2) above, since:

1. it provides a semantic approach to process activity comparison, by making use of domain knowledge. Indeed, it rates two activities as very similar, if they are connected through semantic (i.e., ontological) relations. Specifically, the metric can be properly adapted to operate with different knowledge representation formalisms (e.g., taxonomy vs. semantic network with different characteristics). Very interestingly, it also exploits all the information that can be extracted through process mining (e.g., temporal information), always in a semantic and knowledge-intensive perspective;
2. it explicitly takes into account complex control flow constructs (such as AND and XOR splits/joins – also called gateway nodes henceforth), thus considering the different semantic meaning of control flow connections in a reliable way.

Fully exploiting the semantics of process models in comparison and similarity quantification along the lines illustrated above represents a major development with respect to the literature in the field, as extensively discussed in Section 4. Such a development is likely to provide a significant impact in supporting the expert's work in quality assessment, particularly in those applications where domain knowledge is rich and well consolidated, as is often the case in medicine (Basu, Archer, & Mukherjee, 2012).

Indeed, the positive impact of the framework in practice has already been tested in stroke management (see Section 3), where our approach has outperformed a state-of-the-art metric (La Rosa, Dumas, Uba, & Dijkman, 2013) on a real world event log, providing results that were closer to those of a human expert.

The paper is organized as follows. Section 2 provides the details of our methodological approach. Section 3 showcases experimental results. Section 4 compares our contribution to related works. Section 5 illustrates our conclusions and future research directions.

2. Methods

As stated in the Introduction, our framework first extracts the actual process model from the execution traces, and then performs process model comparison by means of a novel metric. The methodological techniques supporting the first step (process mining) are briefly presented in subSection 2.1, while subSection 2.2 is devoted to the detailed description of our metric, which represents the main contribution of this paper.

2.1. Mining process models

Process mining describes a family of a posteriori analysis techniques (Van der Aalst et al., 2003) exploiting the information recorded in process execution trace repositories (also called event logs), to extract process related information (e.g., process models). Typically, these approaches assume that it is possible to sequentially record events such that each event refers to an activity (i.e., a well defined step in the process) and is related to a particular process instance. Furthermore, some mining techniques use additional information such as the timestamp of the event, or data elements recorded with the event.

Traditionally, process mining has been focusing on discovery, i.e., deriving process models and execution properties from event logs. It is important to mention that, in discovery, there is no a priori model, but, based on logs, some model, e.g., a Petri Net, is constructed. However, process mining is not limited to process models (i.e., control flow), and recent process mining techniques are more and more focusing on other perspectives, e.g., the organisational perspective, the performance perspective or the data perspective. Moreover, as well stated in the Process Mining Manifesto (IEEE Taskforce on Process Mining, 2011), process mining also supports conformance analysis and process enhancement. In this paper, however, we only deal with the process perspective.

In our work, we are currently relying on mining algorithms available within ProM (Van Dongen, Alves De Medeiros, Verbeek, Weijters, & Van der Aalst, 2005), an open source tool which supports a wide variety of process mining and data mining techniques.

In particular, we have mainly exploited ProM's heuristic miner (Weijters, Van der Aalst, & de Medeiros, 2006) for mining the process models. Heuristic miner takes in input the event log, and considers the order of the events within every single process instance execution. The time stamp of an activity is used to calculate this ordering. Heuristics miner can be used to express the main behavior registered in a log. Some abstract information, such as the presence of composite tasks (i.e., tasks semantically related to their constituent activities by means of the ''part-of'' relation), cannot be derived by heuristic miner, that will only build a model including ground (i.e., not further decomposable) activities. On the other hand, it can mine the presence of short distance and long distance dependencies (i.e., direct or indirect sequence of activities), and information

about parallelism, with a certain reliability degree (see also Section 2.2). The output of the mining process is provided as a graph, also called ‘‘dependency graph’’, where nodes represent activities, and arcs represent control flow information.
We have chosen to rely on heuristic miner because it is known to be tolerant to noise, a problem that may affect many real world event logs (e.g., in medicine sometimes the logging may be incomplete). Moreover, heuristic miner labels the output graph edges with several mined information, that we are explicitly considering in process comparison (such as reliability, see Section 2.2). The output of heuristic miner can also be automatically converted into a Petri Net, making its semantics very clear (clearer with respect to the output of other miners).
It is however worth noting that our approach also works with different choices of the mining algorithm: as an example, in Section 3 we will present some results obtained with ProM’s multiphase miner (Van Dongen & Van der Aalst, 2004).

2.2. Calculating process similarity

Since mined process models are represented in the form of graphs, we define a distance based on the notion of graph edit distance (Bunke, 1997). Such a notion calculates the minimal cost of transforming one graph into another by applying edit operations, i.e., insertions/deletions and substitutions of nodes, and insertions/deletions of edges. While string edit distance looks for an alignment that minimizes the cost of transforming one string into another by means of edit operations, in graph edit distance we have to look for a mapping. A mapping is a function that matches (possibly by substituting) nodes to nodes, and edges to edges. Unmatched nodes/edges have to be deleted (or, dually, inserted in the other graph). Among all possible mappings, we will select the one that leads to the minimal cost, having properly quantified the cost of every type of edit operation. We provide a normalized version of the approach in Bunke (1997).
With respect to the available literature approaches (see Section 4 for an extensive comparison), we have introduced two novel
contributions:

1. we operate in a knowledge-intensive way in calculating the cost of activity node substitution (see dt contribution in f subn, Definition 3 below). Most literature approaches simply use an overlap distance to provide the cost of node substitution (i.e., 0 if the nodes are identical, 1 otherwise; see Becker & Laue (2012)). Some others (Dijkman, Dumas, & Garca-Banuelos, 2009) exploit string edit distance on node names. On the other hand, we adopt a more semantic approach, in which domain knowledge is exploited. We allow for the use of different metrics to calculate the cost of activity node substitution, on the basis of the available knowledge representation formalisms. We also add a cost contribution related to edge substitution (f sube in Definition 3 below), able to exploit information learned through process mining. Namely, at the moment we consider: (i) the reliability of a given edge (learned by heuristic miner) – see Definition 1 below, (ii) the percentage of traces that cross a given edge in the mined model (learned, e.g., by some mining algorithms in ProM) – see Definition 2 below, and (iii) statistics about the temporal duration of a given edge. As for item (iii), we have directly calculated the mean and the standard deviation of the temporal duration of edges, by referring to the content of the event log. Different/additional information learned by a miner could be introduced as well in the future;
2. we consider complex control flow information (i.e., other than sequence) between the mined process activities. This information, in our approach, is made explicit in the form of gateway nodes (e.g., AND joins/splits) in the graph. In extending graph edit distance, we only map activity nodes to activity nodes, and gateway nodes to gateway nodes. Our metric is then able to explicitly take into account the cost of gateway node substitution (see dg in f subn, Definition 3 below). In this way, we consider the different semantic meanings of control flow connections.

Formally, the following definitions apply:
Definition 1 (Reliability). The reliability of the edge ei assessing that activity a directly follows activity b in sequence (i.e., ei is an arc from b to a) is calculated as Weijters et al. (2006):
ja > bj jb > aj
relðeiÞ ¼
ja > bj þ jb > aj þ 1
where ja > bj is the number of occurrences in which activity a directly follows activity b in the event log, and jb > aj is the number of occurrences in which activity b directly follows activity a. A negative reliability value means that we must conclude that the opposite pattern holds, i.e., activity b follows activity a. Indeed, the reliability of a relationship (e.g., activity a follows activity b) is not only influenced by the number of occurrences of this pattern in the logs, but is also (negatively) determined by the number of occurrences of the opposite pattern (b follows a). However, edges with a negative reliability do not appear in the output graph (due to threshold mechanisms and proper heuristics (Weijters et al., 2006), that rule them out). Therefore, we deal with reliability values 2 ð0; 1Þ.
Definition 2 (Percentage of traces). The percentage of traces that crossed edge ei, assessing that activity a directly follows activity b in sequence, is calculated as:
ja > bjt
ptðeiÞ ¼ jALLTRACEj
where ja > bjt is the number of traces in which activity a directly follows activity b in the event log, and jALLTRACEj is the total number of available traces in the event log.
With this definition, the percentage of traces 2 ½0; 1.
Definition 3 (Extended Graph Edit Distance). Let G1 ¼ ðN1; E1Þ and G2 ¼ ðN2; E2Þ be two graphs, where Ei and Ni represent the sets of edges and nodes of graph Gi. Let jNij and jEij be the number of nodes and edges of graph Gi. Let M be a partial injective mapping (see Dijkman et al. (2009)) that maps nodes in N1 to nodes in N2 and let subn; sube; skipn and skipe be the sets of substituted nodes, substituted edges, inserted or deleted nodes and inserted or deleted edges with respect to M. In particular, a substituted edge connects a pair of substituted nodes in M. The fraction of inserted or deleted nodes, denoted f skipn, the fraction of inserted or deleted edges, denoted f skipe, and the average distance of substituted nodes, denoted f subn, are defined as follows:
jskipnj
f skipn ¼ jN1j þ jN2j where jskipnj is the number of inserted or deleted nodes;
jskipej f skipe ¼ jE1j þ jE2j where jskipej is the number of inserted or deleted edges;
2 Pn;m2MAdtðn;mÞ þ Px;y2MGdgðx;yÞ f subn ¼
jsubnj
where MA represents the set of mapped activity nodes in the mapping M; MG represents the set of mapped gateway nodes in M; dtðn; mÞ is the distance between two activity nodes m and n in

MA, and dgðx; yÞ is the distance between two gateway nodes x and y in MG.
The average distance of substituted edges f sube is defined as follows:
2 Pðn1;n2Þ;ðm1;m2Þ2Mðjrelðe1Þ relðe2Þj þ jptðe1Þ ptðe2Þj þ jmtðe1Þ mtðe2Þj þ jstðe1Þ stðe2ÞjÞ
f sube ¼

4 jsubej
where edge e1 (connecting node n1 to node m1) and edge e2 (connecting node n2 to node m2) are two substituted edges in M; relðeiÞ is the reliability of edge ei (see Definition 1); ptðeiÞ is the

percentage of traces that crossed edge ei (see Definition 2); mtðeiÞ and stðeiÞ are statistical values (mean and standard deviation of the elapsed times) calculated over all the occurrences of the mi > ni pattern in the traces, and normalized in ½0; 1 dividing by the duration of the longest mi > ni pattern in the log. If one of these parameters is unavailable (e.g., reliability is unavailable because heuristic miner was not used), its contribution is simply set to 0. Different/additional parameters learned by a miner could be considered as well in f sube in the future.
The extended graph edit distance induced by the mapping M is:
wskipnfskipnþwskipefskipeþwsubnf subnþwsubef sube extedit ¼ wskipnþwskipeþwsubnþwsube
where wsubn; wsube; wskipn and wskipe are proper weights
2 ½0; 1.
The extended graph edit distance of two graphs is the minimal possible distance induced by a mapping between these graphs.
The distance dtðn; mÞ between two activity nodes m and n in MA (see Definition 3) is a proper knowledge-intensive distance definition, to be chosen on the basis of the available knowledge representation formalism in the domain at hand. In our experiments, we could rely on a complete, goal-based domain taxonomy on stroke management activities (see Section 3), and adopted Palmer's taxonomic distance (Palmer & Wu, 1995) for calculating dt. Other distance definitions can be relied upon if domain knowledge is available as a semantic network with different characteristics. As an example, the metric in Chiabrando, Likavec, Lombardi, Picardi, and Theseider-Dupré (2011) can be relied upon when dealing with an incomplete ontology, or with a ontology containing many dense sub-ontologies. Our framework is modular and easily adaptable to this end.
To calculate the distance dgðx; yÞ between two gateway nodes x and y we proceed as follows:
1. if x and y are nodes of different types (i.e., a XOR and an AND), their distance is set to 1;
2. if x and y are of the same type (e.g., two ANDs), we have to calculate the difference between: (a) the incoming gateway nodes;
(b) the incoming activity nodes;
(c) the outgoing gateway nodes;(d) the outgoing activity nodes.
As regards item 2. (b), let S1 be the sequence of incoming activity nodes of the first gateway node x; let S2 be the sequence for the second gateway node y. Without loss of generality, suppose that S2 is not longer than S1. In order to compare S1 ad S2, we try all possible permutations in the order of the activity nodes in S2, and take the one that leads to the minimal distance with respect to S1. The distance between the two sequences is the average of the distance between single elements (i.e., pairs of activities), over the length of the longest sequence. The distance between a pair of activities is calculated exploiting the knowledge-intensive approach and the distance dt described above. Every activity in S1

that cannot be mapped to any activity in S2 (because S1 is longer than S2) contributes with a distance of 1.
Item 2. (d) works analogously.
Items 2. (a) and 2. (c) are simpler: identical incoming (respectively, outgoing) gateway nodes in the two sequences (e.g., two ANDs) provide a contribution of 0; different gateway nodes (i.e., an AND and a XOR) provide a contribution of 1. As above, we then calculate the average over the length of the longest sequence of gateway nodes. This procedure is obviously a simplification, since incoming gateway nodes may have other gateway nodes in input as well, but we do not consider this (recursive) information. Similar considerations hold for outgoing gateway nodes. This choice was motivated by computational complexity issues, but could be reconsidered in the future.

The four contributions are then combined as a weighted average dgðx; yÞ in Definition 3 (in which, at the moment, we are setting all the weights to 1, but the choice can of course be differently set in other domains/experiments).
It can be easily verified that our metric, being an extension of the edit distance, preserves the metric properties of non-negativity, identity of indiscernibles, and symmetry. Some versions of the normalized edit distance may fail the triangle inequality in a few very specific experimental situations (see Marzal & Vidal (1993)), but the problem can be tackled, as discussed in Yujian and Bo (2007). Moreover, as clearly stated in Becker and Laue (2012), triangle inequality is not considered to be essential for measuring process distance.
To find the mapping that leads to the minimal distance we resort to a greedy approach, in order to contain computational costs. It can be shown that the algorithm works in cubic time on the number of nodes of the larger graph (Dijkman et al., 2009). As is well known, a greedy algorithm is an algorithm that follows the problem-solving heuristic of making the locally optimal choice at each stage, with the hope of finding a global optimum. A greedy strategy does not in general produce an optimal solution, but nonetheless a greedy heuristic may yield locally optimal solutions that approximate a global optimal solution in a reasonable time.

3. Results

In this section, we will present our experiments. SubSection 3.1 will introduce the testbed we realized for testing our approach. SubSection 3.2 will then provide the details of the application domain, the available data, and the results.

3.1. The experimental testbed

We have realized a testbed, which easily allows to set up different experimental configurations. Within the testbed, it is possible to select different mining algorithms (e.g., different miners available in ProM, or obtained from other sources, or implemented from scratch) and different similarity metrics.
Since not all the miners are supposed to provide their output in the same format, our testbed includes a translation module, that produces an XML output, specifying node and edge properties in an interoperable way. It is worth noting that not even the different ProM miners provide the same output format. Specifically, they can export output graphs in the DOT language, but the type of information, and the way it is coded, may differ from miner to miner. For instance, typically gateway nodes are not explicit in the heuristic miner output graph, but control flow information is provided by means of some node parameters. On the other hand, multi-phase miner provides explicit gateway nodes. It is therefore necessary to translate the output formats in a common format to guarantee interoperability, and XML represent a natural solution to this end.
The translation module exploits a java class, which is a parser (e.g., a parser of a DOT file in the case of ProM miners). All the parsers must implement a java interface, which establishes the parser structure. It is therefore easy to add a new parser (for a new miner), because the interface already provides the declarations of the methods to be implemented.
The output XML file contains elements (nodes and edges), and their associated attributes, such as node type (activity or gateway), node information and edge information (e.g., edge input nodes, edge output nodes, reliability, percentage of traces). By means of this translation, we then always make gateway nodes explicit, as required by our metric. Some attributes are optional (e.g., edge reliability, calculated by heuristic miner, may be not provided by other miners).
The testbed also includes a loading module, that loads two XML files (describing the two processes to be compared) into the data structure (an hash table) used for graph mapping and distance calculation. At this stage, the test is fully independent of the exploited miner.
The testbed allows to choose a proper similarity metric (e.g., the distance described in this paper, the distance in La Rosa et al. (2013), or another distance defined by users). Not necessarily the

distance must use all the node or edge parameters provided by the miner (i.e., some XML attributes may be ignored in graph comparison).
In our experiments (see Section 3.2), we have tested four different configurations: (a) heuristic miner + the new distance described in this paper; (b) heuristic miner + the distance in La Rosa et al. (2013), (c) multi-phase miner + the new distance described in this paper; (d) multi-phase miner + the distance in La Rosa et al. (2013).
ProM's multi-phase miner (Van Dongen & Van der Aalst, 2004) provides in output an Event-driven Process Chain (EPC), i.e., a graph that contains three types of nodes: activities, gateway nodes, events. Events describe the situation before/after the execution of an activity; they do not provide additional information about the process control flow. We have therefore ignored events in distance calculation. By means of the testbed translation module, the EPCs have been converted into standard XML files, as described above.
The distance in La Rosa et al. (2013) has been chosen because it is one of the very few literature contributions (see Section 4) that somehow considers gateway nodes (but not domain knowledge, nor additional mined information reported on edges) in graph comparison.
Results are provided in the next section.

3.2. Testing the framework in stroke management

We have applied our framework to stroke management processes.
A stroke is the rapidly developing loss of brain function(s) due to disturbance in the blood supply to the brain. This can be due to ischemia (lack of glucose and oxygen supply) caused by a thrombosis or embolism, or to a hemorrhage. As a result, the

affected area of the brain is unable to function, leading to inability to move one or more limbs on one side of the body, inability to understand or formulate speech, or inability to see one side of the visual field. A stroke is a medical emergency and can cause permanent neurological damage, complications, and death. It is the leading cause of adult disability in the United States and Europe and the number two cause of death worldwide.
In our experiments, we could rely on a database of 9929 traces, collected at 16 stroke units of the Stroke Unit Network (SUN) of Regione Lombardia, Italy (Micieli, Cavallini, Quaglini, Fontana, & Duè, 2010). Such stroke units are all equipped with similar human and instrumental resources. The number of traces varies from 266 to 1149. Traces are composed of 13 activities on average. Data refer to the period 2009–2012.
We also could exploit domain knowledge, in the form of a taxonomy of stroke management activities. In such a taxonomy, classes are defined on the basis of their goal. In our distance calculation (contribution dt in f subn, see Definition 3), Palmer's taxonomic distance was exploited (Palmer & Wu, 1995). This distance allows us to exploit the hierarchical structure, since the distance between two activities is set to the normalized number of arcs on the path between the two activities themselves in the taxonomy. The underlying idea is that two different activities are more or less distant on the basis of their goal.
We asked a stroke management expert other than our medical co-authors (i.e., Dr. I. Canavero, see Acknowledgments) to provide a ranking of the SUN stroke units (see Table 1, column 2), on the basis of the quality of service they provide. The top level unit will be referred as H0 in the experiments. The expert identified 5 hospitals (H1–H5) with a high similarity level with respect to H0; 5 hospitals (H6–H10) with a medium similarity level with respect to H0; and 5 hospitals (H11–H15) with a low similarity level with respect to H0. The ordering of the hospitals within one specific similarity level is not very relevant. It is instead important to distinguish between different similarity levels.

The medical expert also provided the following values for distance weights: wsubn ¼ 1; wsube ¼ 0:2; wskipn ¼ 1; wskipe ¼ 0:6. The rationale behind this choice is the following: in stroke management, for most of the processes, activities are more important than their sequential connection, therefore a node substitution (see wsubn) or deletion (see wskipn) have the highest weights. Edge deletion (see wskipe) is more important than edge substitution (see wsube), because a change in the activity execution sequence must still be strongly penalized (even if not as strongly as a change in the activities themselves). The penalty for edge substitution is the lowest, because it refers to situations in which the activity sequence is identical in the models; only, information (e.g., reliabilities or times) associated to the edges at hand may be different. It is important to take into account these differences, but they do not impact as much as a change in the model control flow.

It is however worth noting that a sensitivity analysis can be conducted to automate weight setting, when expert knowledge is not available.

As explained in Section 3.1, thanks to our testbed we were able to set up four different experimental configurations. Namely, we could mine the process models according to heuristic miner, and to multi-phase miner. We then ordered the two available process model sets with respect to H0, resorting to the new distance defined in this paper (see Section 2.2), and to the distance in La Rosa et al. (2013), globally obtaining four rankings. Results are shown in Table 1. Column 1, in Table 1, shows the levels of similarity with respect

to the reference hospital. Column 2 shows the ranking according to the human medical expert; columns 3 and 4 show the results obtained by mining the process models by means of heuristic

Table 1

Ordering of 15 hospitals, 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 ranking	New dist. heuristic	LaRosa dist. heuristic	New dist. M-phase	LaRosa dist. M-phase
High	H1	H14	H14	H9	H14
High	H2	H3	H3	H2	H3
High	H3	H2	H9	H3	H1
High	H4	H1	H1	H1	H7
High	H5	H11	H12	H11	H8
Medium		H6	H10	H6	H12	H5
Medium		H7	H4	H11	H7	H6
Medium		H8	H7	H10	H4	H13
Medium		H9	H9	H2	H10	H11
Medium		H10	H6	H4	H15	H2
Low	H11	H8	H13	H8	H10
Low	H12	H12	H8	H13	H15
Low	H13	H15	H15	H6	H4
Low	H14	H13	H7	H14	H12
Low	H15	H5	H5	H5	H9

miner, relying on the distance defined in this paper and on the one defined by La Rosa et al. (2013), respectively. Similarly, columns 5 and 6 show the results obtained by mining the process models by means of multi-phase miner.

When exploiting heuristic miner, the distance defined in this paper correctly rates three process models in the high similarity group (60%), four process models in the medium similarity group

(80%), and three process models in the low similarity group (60%, column 3). The distance in La Rosa et al. (2013), on the other hand, correctly rates only two process models in every group (40%, column 4).
When exploiting multi-phase miner, the distance defined in this paper correctly rates three process models in the high similarity group (60%), two process models in the medium similarity group (40%), and two process models in the low similarity group (40%, column 5). The distance in La Rosa et al. (2013), correctly rates only two process models in the high similarity group (40%), one process model in the medium similarity group (20%), and two process models in the low similarity group (40%, column 6).
Thus, our distance produces results that are closer to the qualitative ranking provided by the human expert. Very interestingly, this situation holds both when relying on heuristic miner, and when relying on multi-phase miner. However, our metric works particularly well when adopting heuristic miner, probably because it mines more information (e.g., reliability), that is later exploited by the metric. These data are simply unavailable when using multi-phase miner, therefore in this last case distance calculation is less knowledge-intensive.
In conclusion, our knowledge-intensive approach to distance calculation has proved to be able to provide a high quality process model comparison in practice. As such, it could be confidently used for comparing medical processes in a quality evaluation perspective, at least when comparing hospitals that are equipped with similar resources, as it was the case in our experiments.

4. Related work

Similarity-based graph comparison is a very active research area, which is giving birth to different methodological approaches and software tools. Graph databases, like, e.g., HypergraphDB (Iordanov, 2010) and DEX (Martínez-Bazan, Gómez-Villamor, & Escale-Claveras, 2011), are gaining popularity, for working in emerging linked data such as social network data and biological data. However, in this section we will focus on contributions that are more closely related to graph similarity in process/workflow management research.
As stated in Dijkman, Dumas, Van Dongen, Kaarik, and Mendling (2011), Becker and Laue (2012), three classes of similarity metrics can be considered to deal with process model comparisons: (i) node matching similarity, which compares the labels attached to process model nodes; (ii) structural similarity, which compares node labels, as well as graph topology; (iii) behavioral similarity, which compares node labels, as well as the behavioral/ causal relations captured in the process models.
While class (i) somehow oversimplifies the problem, class (iii) requires causal information, which we do not currently mine. Indeed, our work is related to class (ii). Therefore, in the following we will focus on structural similarity approaches.
The goal of comparing objects with a complex structure (i.e., graphs) entails the definition of a nontrivial notion of distance. The issue of providing a proper graph distance definition has been afforded in the literature, following three main directions, i.e.,:

1. relying on a local notion of similarity (two subgraphs are similarif their neighboring nodes are similar), as in the similarity flooding algorithm (Melnik, Garcia-Molina, & Rahm, 2002);
2. relying on subgraph isomorphism, e.g., to find maximum common sub-graphs (Valiente, 2002), and
3. adapting the edit distance notion to graphs (Bunke, 1997).

We are currently following direction (3), but directions (1) and (2) could be considered in our future work for comparison.
The SAI toolkit (Kendall-Morwick & Leake, 2011) is transversal with respect to the three directions (1)–(3) outlined above, since it is a framework for workflow representation and comparison that allows different similarity measures to be used.

The work in Madhusudan, Zhao, and Marshall (2004), on the other hand, describes an approach specifically related to direction (1). In Madhusudan et al. (2004), a retrieval system for supporting incremental workflow modeling is presented. The system proposes a similarity-based reuse of workflow templates using a planner that employs an inexact graph matching algorithm based on similarity flooding. For computing similarities, the algorithm relies on the idea that elements of two distinct graphs are similar, when their adjacent elements are similar. The algorithm propagates the similarity from a node to its respective neighbors based on the topology in the two graphs. However, edge similarity is not considered.

The work in Kapetanakis, Petridis, Knight, Ma, and Bacon (2010) belongs to direction (2), as it exploits a maximum common subgraph approach for similarity-based process retrieval, in a retrieval system for supporting business process monitoring. Interestingly, the metric in Kapetanakis et al. (2010) takes into account temporal information, since it combines a contribution related to activity similarity, and a contribution related to delays between activities.

The approach in Goderis, Li, and Goble (2006) relies on graph isomorphism (direction (2)) for retrieving scientific workflows (e.g., pipelines for bioinformatics experiments). Unlike business workflows, scientific workflows have a strong focus on the data flow, typically restricting the control flow to a partial ordering of the tasks. The work in Ma, Zhang, and Lu (2014) focuses on data oriented workflows as well. It defines a formal structure called Time Dependency Graph (TDG), and exploits it as a representation model of data oriented workflows with variable time constraints. A distance measure is proposed for computing workflow similarity by their normalization matrices, established based on their TDGs. The peculiarity of data oriented workflows, however, make these contributions less closely related to our approach.

The works following direction (3), on the other hand, extend the notion of graph edit distance (Bunke, 1997), which calculates the minimal cost of transforming one graph into another, by applying insertions/deletions and substitutions of nodes, and insertions/ deletions of edges.

The work in Minor, Tartakovski, Schmalen, and Bergmann (2008) makes use of a normalized version of the graph edit distance. The approach is used to support workflow modification in an agile workflow system, and takes into account control flow information as well as activity information. However, Minor et al. (2008) only makes use of syntactical information in the definition of the edit operation costs. Moreover, the work is limited to considering (small) changes with respect to a running process instance.

The work in Kunze and Weske (2011) relies on graph edit distance, and exploits string edit distance on node names to determine the cost of node substitutions. The work in Li, Reichert, and Wombacher (2008) encapsulates a set of edit operations into the so-called ‘‘high-level change operations’’, and measures distance on the basis of the number of high-level change operations needed to transform one graph into another. The work in Bae, Caverlee, Liu, and Yan (2006) transforms a graph into an ordered tree, and then exploits tree edit distance.

With respect to Minor et al. (2008), Kunze and Weske (2011), Li et al. (2008) and Bae et al. (2006), we make use of semantic information in activity comparison. We also make explicit use of the information mined/learned from the data in the mapped edges contribution.

The use of semantic information in similarity calculation is a very active research area in text understanding, where several approaches have been proposed (see the survey in Sánchez, Batet, Isern, & Valls (2012)), that compute the information content of concepts from the knowledge provided by ontologies; the work in Sánchez and Batet (2013), for example, proposes a similarity measure that considers multiple ontologies in an integrated way.

In the field of business process management, process semantics have been exploited in the literature to accomplish various tasks. The work in Jung (2009), for instance, proposes a framework based on aligning business ontologies for integrating heterogeneous business

processes, in order to provide efficient collaboration (i.e., communication and sharing) between them.
However, these contributions are only loosely related to our work.
Focusing more specifically on our research problem, the use of semantic information in structured process model comparison and retrieval is proposed in Bergmann and Gil (2014), a system working on workflows represented as semantically labeled graphs. The work in Bergmann and Gil (2014) adopts a graph edit distancebased approach, which is particularly suitable for scientific workflows. The paper proposes to use a metric in which the similarity between two mapped nodes or arcs makes explicit use of their semantic description. However, the framework is presented in a general, high-level way, and the specific costs of edit operations are not provided. With respect to our work, Bergmann and Gil (2014) is much more focused on the data flow, which was not considered in our current application. As already observed, this makes this work less related to ours.
The closest works with respect to our approach are Dijkman et al. (2009) and La Rosa et al. (2013) (which extends Dijkman et al. (2009)). Specifically, Dijkman et al. (2009) provides a normalized version of the graph edit distance (Bunke, 1997) for comparing business process models, and defines syntactical edit operation costs for activity node substitution, activity node insertion/deletion, and edge insertion/deletion.
With respect to Dijkman et al. (2009), we have introduced several novel contributions:
(a) we have moved towards a more semantic and knowledgeintensive approach in activity node substitutions, by allowing the exploitation of domain knowledge. The work in Dijkman et al. (2009), on the other end, relies on edit distance between activity node names;
(b) always in the knowledge-intensive perspective, we have explicitly considered edge substitutions, which was disregarded in Dijkman et al. (2009). Indeed, some miners label edges with information that can be relevant in graph comparison. Moreover, statistical temporal information can be mined from the event log. All these data are exploited in
our metric;
(c) the work in Dijkman et al. (2009) does not take into account control flow elements other than sequence, so that gateway nodes are not represented in the graph, and not used in distance calculation. On the contrary, we have considered this issue as well in our contribution.
The work in La Rosa et al. (2013) extends the work in Dijkman et al. (2009) specifically by dealing with issue (c) (but not with (a) and (b)): indeed, the authors explicitly represent gateway nodes, in order to describe, e.g., parallelism and mutual exclusion. With respect to our approach, La Rosa et al. (2013) simplifies the treatment of incoming/outgoing activity nodes with respect to a gateway node: in comparing two gateway nodes, it only calculates the fraction of their incoming (respectively, outgoing) activity nodes that were mapped; it does not consider the cost of their substitution, i.e., how similar this mapped activity nodes are. On the other hand, we explicitly use domain knowledge in this phase of distance calculation as well, as described in Section 2.2. The work in La Rosa et al. (2013) also considers activity nodes that are connected to the gateway node at hand indirectly, i.e., through a path of nodes that can also include gateway nodes. On the contrary, we limit our comparison to incoming/outgoing activity nodes that are directly connected to the gateway node we want to examine. In La Rosa et al. (2013) incoming/outgoing gateway nodes are completely disregarded.
Despite these differences, La Rosa et al. (2013) is still the closest literature contribution with respect to our work. This justifies the choice of comparing our results to the ones that can be obtained by the metric in La Rosa et al. (2013), on the stroke dataset (see Section 3). As observed, our metric outperformed the one in La Rosa et al. (2013), probably thanks to the use of domain knowledge and edge information (including temporal information). Indeed, when domain

knowledge is available, rich and well consolidated, as is often the case in medicine, its exploitation can surely improve the quality of any automated support to the expert's work – including process comparison (see e.g., Basu et al. (2012)). Moreover, time is a very important parameter in medical application, particularly when referring to emergency medicine, as it is in the case of stroke.
Most of the approaches following directions (1)–(3) above typically suffer from problems related to their high computational complexity, which is sometimes mitigated by resorting to greedy techniques (see, e.g., Dijkman et al. (2009), and our own approach). To avoid these problems, however, some previous works have investigated structureless workflow retrieval, where workflow representation is a plain textual description, a set of tags (Goderis et al., 2006), or a set of abstract workflow features (Bergmann, Freßmann, Maximini, Maximini, & Sauer, 2006). Recently, a probabilistic similarity model for workflow execution paths was also proposed (Becker, Bergener, Breuker, & Räckers, 2011). Other approaches have suggested the use of a two-step procedure, which combines an initial and comparatively inexpensive retrieval step, to winnow the instances to be considered, with a more expensive strategy that ranks the remaining instances (as in the well-known MAC/FAC system (Forbus, Gentner, & Law, 1995)) (KendallMorwick & Leake, 2011, 2012). In Bergmann and Gil (2014), some procedures for non-exhaustive search, based on the A∕ algorithm, are provided. More work on computational performances will be considered in our future research as well.

5. Discussion, conclusions and future work

In this paper, we have described a novel framework for process mining and process comparison. The main research contribution of our work is represented by the novel metric we have defined to support process comparison. The strength of such a metric is twofold. First, it provides a semantic approach to process activity comparison, by making use of domain knowledge. In detail, it calculates activity similarity on the basis of activity connections through semantic (i.e., ontological) relations in the available domain knowledge representation formalism. Interestingly, it also exploits all the information that can be extracted through process mining (e.g., temporal information), always in a semantic and knowledge-intensive perspective. As a second development from the methodological viewpoint, our metric explicitly takes into account complex control flow constructs (such as AND and XOR splits/joins), often ignored or oversimplified in available literature contributions.
Practical implications of the adoption of the new metric in process similarity calculation are basically related to an increase in the reliability of results. Indeed, out metric fully captures the semantic meaning of process activities and of their connections, and process semantics are explicitly resorted to in difference quantification. A reliable comparison is of course the first step towards a reliable conformance checking activity, a reliable performance evaluation, or a reliable analysis of local adaptation needs. This impact is particularly significant in medical domains, where patient's health is addressed, and best practices must be correctly identified.
Indeed, the positive impact of the framework in practice has already been tested in stroke management (see Section 3), where experimental results have favored our contribution, in comparison to the distance definition reported in La Rosa et al. (2013), the most similar already published work with respect to our approach. Indeed our metric, that could take advantage of domain knowledge, in the form of a taxonomy, outperformed the work in La Rosa et al. (2013) on a real world stroke management event log, and provided results that were closer to those of a human expert. This held both when relying on heuristic miner to learn process models, and when relying on multi-phase miner. However, our metric worked particularly well when adopting heuristic miner, probably because it mines more information, that are simply unavailable when using multi-phase miner; therefore in this last case distance calculation is less knowledge-intensive.

We believe that our metric could therefore be confidently used for comparing medical processes in a quality evaluation perspective. However, our framework is modular enough to be adapted and tested in very different application domains, and when dealing with different knowledge representation formalisms; we would like to plan further experiments in different applications in the near future.

Besides this experimental future work, we plan to address more theoretical and technological issues as well. Indeed, in its current implementation, our framework still suffers from some limitations, that need to be addressed. Specifically:

currently, we have mainly focused on defining a proper metric, able to overcome the limitations encountered in the process comparison literature; as for the process mining step, we just exploited some of available mining algorithms (chosen among the most ''popular'' miners in ProM). In the future, we would like to test different process mining algorithms as well, to increase the practical applicability of our approach. The availability of the testbed described in Section 3.1 will make these additional experiments easily configurable. Notably, the exploitation of a different algorithm might impact on distance calculation: adjustments may be required, in order to take into account specific outputs that were not provided by heuristic miner or multiphase miner;

as a further methodological enhancement in the process mining step, we would also like to define a novel mining approach ourselves, able to directly take into account temporal information (see, e.g., Burattin & Sperduti (2010)), and to improve model precision, reducing the number of mined paths that do not correspond to any trace in the log (Canensi, Montani, Leonardi, & Terenziani, 2014). We believe that a greater model precision will enhance the reliability of adopting the whole framework
in practice, especially in the medical field;

from a more technological point of view, we plan to integrate our distance calculation as a plug-in in the ProM 6 environment. This will allow us to make our work available to the process mining community, facilitating the collection of feedback from other users, and the testing also in very different application domains.

We believe that these enhancements could represent a relevant added value in our work, by making process comparison even more versatile, reliable and useful in practice.

Acknowledgments

We would like to thank Dr. I. Canavero for her independent work in the experimental phase.
This research is partially supported by the GINSENG Project, Compagnia di San Paolo.

References

Bae, J., Caverlee, J., Liu, L., & Yan, H. (2006). Process mining by measuring process block similarity. In Proceedings of the 2006 international conference on business process management workshops BPM'06 (pp. 141–152). Berlin, Heidelberg: Springer-Verlag.

Basu, R., Archer, N., & Mukherjee, B. (2012). Intelligent decision support in healthcare. Analytics, Jan–Feb 2012, 33–38.

Becker, J., Bergener, P., Breuker, D., & Räckers, M. (2011). On measures of behavioral distance between business processes. In Proceedings Wirtschaftsinformatik (pp. 665–674).

Becker, M., & Laue, R. (2012). A comparative survey of business process similarity measures. Computers in Industry, 63, 148–167.

Bergmann, R., Freßmann, A., Maximini, K., Maximini, R., & Sauer, T. (2006). Casebased support for collaborative business. In Advances in case-based reasoning. In T. Roth-Berghofer, M. Gker, & H. Gvenir (Eds.). Lecture notes in computer science (Vol. 4106, pp. 519–533). Berlin Heidelberg: Springer.

Bergmann, R., & Gil, Y. (2014). Similarity assessment and efficient retrieval of semantic workflows. Information Systems, 40, 115–127.
Bunke, H. (1997). On a relation between graph edit distance and maximum common subgraph. Pattern Recognition Letters, 18(8), 689–694.
Burattin, A., & Sperduti, A. (2010). Heuristics miner for time intervals. In ESANN 2010, 18th European symposium on artificial neural networks, Bruges, Belgium, April 28–30 (pp. 41–46).
Canensi, L., Montani, S., Leonardi, G., & Terenziani, P. (2014). Chapman: A context aware process miner. In Proceedings of workshop on synergies between case-based reasoning and data mining, international conference on case based reasoning, September 9 (pp. 202–212).
Chiabrando, E., Likavec, S., Lombardi, I., Picardi, C., & Theseider-Dupré, D. (2011). Semantic similarity in heterogeneous ontologies. In P. D. Bra & K. Grønbæk (Eds.), HT'11, Proceedings of the 22nd ACM conference on hypertext and hypermedia, Eindhoven, The Netherlands, June 6–9 (pp. 153–160). ACM.
Dijkman, R., Dumas, M., & Garca-Banuelos, R. (2009). Graph matching algorithms for business process model similarity search. In Proceedings of the 7th international conference on business process management BPM'09 (pp. 48–63). Berlin, Heidelberg: Springer-Verlag.
Dijkman, R., Dumas, M., Van Dongen, B., Kaarik, R., & Mendling, J. (2011). Similarity of business process models: metrics and evaluation. Information Systems, 36, 498–516.
Forbus, K. D., Gentner, D., & Law, K. (1995). Mac/fac: A model of similarity-based retrieval. Cognitive Science, 19, 141–205.
Goderis, A., Li, P., & Goble, C. A. (2006). Workflow discovery: The problem, a case study from e-science and a graph-based solution. In Proceedings of international conference on web services, ICWS'06, September 18–22 (pp. 312–319).
IEEE Taskforce on Process Mining (2011). Process mining manifesto. Retrieved from <http://www.win.tue.nl/ieeetfpm>. (last accessed on 4/11/2013).
Iordanov, B. (2010). Hypergraphdb: A generalized graph database. In Web-Age information management. In H. Shen, J. Pei, M. zsu, L. Zou, J. Lu, & T.-W. Ling, et al. (Eds.). Lecture notes in computer science (Vol. 6185, pp. 25–36). Berlin Heidelberg: Springer.
Jung, J. (2009). Semantic business process integration based on ontology alignment. Expert Systems with Applications, 36, 11013–11020.
Kapetanakis, S., Petridis, M., Knight, B., Ma, J., & Bacon, L. (2010). A case based reasoning approach for the monitoring of business workflows. In Case-based reasoning. Research and development. In I. Bichindaritz & S. Montani (Eds.). Lecture notes in computer science (Vol. 6176, pp. 390–405). Berlin Heidelberg: Springer.
Kendall-Morwick, J., & Leake, D. (2011). A toolkit for representation and retrieval of structured cases. In B. Diaz-Agudo & A. Cordier (Eds.), Proceedings of the ICCBR 2011 workshop on process-oriented case-based reasoning, PO-CBR'11, September 12–15 (pp. 111–120). University of Greenwich.
Kendall-Morwick, J., & Leake, D. (2012). On tuning two-phase retrieval for structured cases. In L. Lamontagne & J. A. Recio-García (Eds.), Proceedings of the ICCBR 2012 workshop on process-oriented case-based reasoning, PO-CBR'12, September 3–6 (pp. 25–34). University of Lyon.
Kunze, M., & Weske, M. (2011). Metric trees for efficient similarity search in large process model repositories. In Business process management workshops. In M. zur Muehlen & J. Su (Eds.). Lecture notes in business information processing (Vol. 66, pp. 535–546). Berlin Heidelberg: Springer.
La Rosa, M., Dumas, M., Uba, R., & Dijkman, R. (2013). Business process model merging: An approach to business process consolidation. ACM Transactions on Software Engineering and Methodology, 22, 937–948.

Li, C., Reichert, M., & Wombacher, A. (2008). On measuring process model similarity based on high-level change operations. In Conceptual modeling – ER 2008. In Q. Li, S. Spaccapietra, E. Yu, & A. Oliv (Eds.). Lecture notes in computer science (Vol. 5231, pp. 248–264). Berlin Heidelberg: Springer.
Madhusudan, T., Zhao, J., & Marshall, B. (2004). A case-based reasoning framework for workflow model management. Data and Knowledge Engineering, 50, 87–115.
Martínez-Bazan, N., Gómez-Villamor, S., & Escale-Claveras, F. (2011). Dex: A highperformance graph database management system. In IEEE 27th international conference on data engineering workshops, ICDEW'11, April 11–16 (pp. 124–127).
Marzal, A., & Vidal, E. (1993). Computation of normalized edit distance and applications. IEEE Transactions on Pattern Analysis and Machine Intelligence, 15, 926–932.
Ma, Y., Zhang, X., & Lu, K. (2014). A graph distance based metric for data oriented workflow retrieval with variable time constraints. Expert Systems with Applications, 41, 1377–1388.
Melnik, S., Garcia-Molina, H., & Rahm, E. (2002). Similarity flooding: A versatile graph matching algorithm and its application to schema matching. In Proceedings of 18th international conference on data engineering, February 26– March 1 (pp. 117–128).
Micieli, G., Cavallini, A., Quaglini, S., Fontana, G., & Duè, M. (2010). The Lombardia stroke unit registry: 1-Year experience of a web-based hospital stroke registry. Neurological Sciences, 31(5), 555–564.
Minor, M., Tartakovski, A., Schmalen, D., & Bergmann, R. (2008). Agile workflow technology and case-based change reuse for long-term processes. International Journal of Intelligent Information Technologies, 4, 80–98.
Palmer, M., & Wu, Z. (1995). Verb semantics for english-chinese translation. Machine Translation, 10, 59–92.
Sánchez, D., & Batet, M. (2013). A semantic similarity method based on information content exploiting multiple ontologies. Expert Systems with Applications, 40, 1393–1399.
Sánchez, D., Batet, M., Isern, D., & Valls, A. (2012). Ontology-based semantic similarity: A new feature-based approach. Expert Systems with Applications, 39, 7718–7728.
Valiente, G. (2002). Algorithms on trees and graphs. Berlin: Springer.
Van der Aalst, W., Van Dongen, B., Herbst, J., Maruster, L., Schimm, G., & Weijters, A. (2003). Workflow mining: A survey of issues and approaches. Data and Knowledge Engineering, 47, 237–267.
Van Dongen, B., Alves De Medeiros, A., Verbeek, H., Weijters, A., & Van der Aalst, W. (2005). The prom framework: A new era in process mining tool support. In Applications and theory of petri nets 2005. In G. Ciardo & P. Darondeau (Eds.). Lecture notes in computer science (Vol. 3536, pp. 444–454). Berlin Heidelberg: Springer.
Van Dongen, B., & Van der Aalst, W. (2004). Multi-phase process mining: Building instance graphs. In Conceptual modeling ER 2004. In P. Atzeni, W. Chu, H. Lu, S. Zhou, & T.-W. Ling (Eds.). Lecture notes in computer science (Vol. 3288, pp. 362–376). Berlin Heidelberg: Springer.
Weijters, A., Van der Aalst, W., & de Medeiros, A. A. (2006). Process mining with the heuristic miner algorithm, WP 166. Eindhoven University of Technology, Eindhoven.
Yujian, L., & Bo, L. (2007). A normalized Levenshtein distance metric. IEEE Transactions on Pattern Analysis and Machine Intelligence, 29, 1091–1095.