Workflow Mining : Discovering Generalized Process Models from Texts

Ahmed Halioui, Petko Valtchev and Abdoulaye Baniré Diallo
Computer Science Department, Laboratoire Bioinformatique
Université du Québec à Montréal, Québec, Canada
halioui.ahmed@uqam.ca, valtchev.petko@uqam.ca, diallo.abdoulaye@uqam.ca

Abstract

Contemporary workflow management systems are driven by explicit process models specifying the inter-dependencies between tasks. Creating these models is a challenging and time-consuming process. Existing approaches for mining concrete workflows into models tackle design aspects with no regard for the diverging abstraction levels of the tasks. Concrete workflow logs represent tasks and cases of concrete events –partially or totally ordered– grounding hidden multi-level semantics and contexts. Relevant generalized events could be rediscovered within processes. Here, we propose an ontology-based workflow mining system generating patterns from sequences of events which are themselves extracted from texts. Our system T-GROWLer (GeneRalized Ontology-based Workflow minER within Texts) is based on two ontology-based modules: a workflow extractor and a pattern miner. To this end, it uses two different ontologies: a domain one and a process one. The first ontology supports workflow extraction from texts. Extracted concrete knowledge is enhanced using the process ontology and then fed into the generalized workflow pattern miner.

1 Introduction

The goal of data mining workflows is to extract patterns about processes from event logs [1] [2]. Depending on the process mining technique used, information presented in such logs may vary. The challenge is to extract such data from a variety of data sources, *e.g.* e-mails, databases and texts. Different types of workflow representations are proposed in the literature of process mining [1] [3] [4]. Petri Networks are the most popular design representing the dynamic aspect of processes. The Alpha-algorithm [1], is one of the first process discovery algorithms. An example of discovered patterns within shopping baskets web pages is: $\langle$ if activity $a : FillInDeliveryInfo$ is followed by $b : CancelOrder$ but b is never followed by $a \rangle$, also used in other models, like HMM, YAWL, BPMN, etc. [1] [5]. However, the representational bias of each model should be considered while mining patterns. Workflow representations delimit the search space of patterns and circumscribe the expressive power of models. Moreover, depending on the questions one seeks to answer, different views on the available data could be presented. In reality, there's no "real" unique model, but different ones. There is a huge gap of knowledge between concrete workflows and models (*i.e.* experience and skills) [1]. Bridging this gap amounts to represent different levels of knowledge. Models should represent the expertise within workflows: which is in some level between concrete (actual) and generalized (abstract) workflows.

In this paper, we adopt an ontological approach to represent the different levels of expertise. Ontologies offer shared vocabularies to facilitate tasks of knowledge integration and interpretation via triples of concepts and relations. Building ontologies is a proliferating research topic in artificial intelligence, bioinformatics and computational medicine [6] [7]. Today, ontologies are used to represent data via concepts and relationsfor semantic enrichment. We propose to use ontological representations for two purposes: (1) the extraction of workflow terms and relations from texts, and (2) the extraction of generalized (abstract) workflow patterns. Our approach of pattern mining from scientific texts follows an ontology-based representation of experience within scientific texts. In this study, we distinguish two types of knowledge in order to represent experience and skills: theoretical knowledge and processual knowledge. The former is used to build a domain ontology representing concepts and relations of one domain analysis basics (theory). While a processual ontology is used to represent another dimension of knowledge which is domain practices and software experience. It is specifically used to extract abstract but relevant patterns from texts. Concepts and relations of processual ontology represent explicitly elements of workflows (events and activities). Here, we distinguish two types of workflows: (1) concrete workflows which elements are terms and links between them and (2) abstract workflows representing more general concepts and relations. Only the processual ontology is used in the extraction steps of these two kinds of workflows.

The rest of the paper is organized as follows: the next section presents related work and background information on workflow pattern-based mining. In section 3, we define the pipeline of our approach and the models representing the different kinds of knowledge. We also describe the different processes of concrete workflow extraction from texts and gener-

alized pattern mining. The proposed model is applied to the phylogenetic workflows where processes describe tools and methods used in the phylogenetic inference. Section 4 shows the phylogenetic domain knowledge. Section 5 presents the experimental evaluation and results.

2 Background and related work

Given a universe of items (events) O, a database $\mathcal{D}$ of records (processes) that combine items from O, and a frequency threshold σ, pattern mining's goal is to extract the family $\mathcal{F}_\sigma$ of patterns present in at least σ records. Two binary relations are generally used in a pattern mining problem combining hierarchies: the *generality* [1] between patterns $\sqsubseteq_\Gamma$ (where Γ is a pattern language) and the instantiation $\preccurlyeq$ between a data record and a pattern. In the simplest settings, both data records and patterns are sets of items (itemsets). Hence, the mining goal is to find all the frequent subsets of a family of sets $\mathcal{D} \subseteq \Delta$ where Δ is the data language. We find in the literature more elaborate data structures such as sequences, episodes and graphs [8]. In the first studies on sequential pattern mining, several work, about including *hierarchies* in the mining process in order to enrich semantically generated patterns, are proposed. This research axis looks for generalized pattern languages built on the top of concepts C and abstracting objects from O. Concepts from a taxonomy H represent *is-a* relationships. In fact, a taxonomy is a simple domain ontology. The second wave of studies on generalized pattern mining uses ontologies as a Directed Acyclic Graphs (DAG). Furthermore, we find that ontologies are also used in the different steps of data mining: (a) in the preprocessing step to filter infrequent items, (b) while generating patterns (in the joining step) or (c) in the postprocessing step in order to prune irrelevant patterns.

Among the first researches on using knowledge representations in pattern mining domain, we find the work of [9]. Authors propose three data structures: HG-trees (generalized hierarchical trees), AR-rules (attribute-relation rules) and constraints on the environment EBC. Inspired from this work, in [10], the authors used hierarchies in pattern mining process. Their algorithm GSP (Generalized Sequential Patterns) searches for $\mathcal{F}_\sigma$ through the pattern space $\langle \Gamma, \sqsubseteq_\Gamma \rangle$ by exploring the monotony in frequency with regard to the generality operator $\sqsubseteq_\Gamma$. Their *Apriori* miner performs a level-wise top-down traversal of the pattern space. On itemsets, it examines patterns at level k - that is, of size k - on two points: the method computes the frequency of candidate patterns by matching them against the records in $\mathcal{D}$ whereas it generates $k+1$ candidates by combining pairs of frequent k-patterns. However, their approach is expensive in terms of memory and computation. Their representation of $\mathcal{D}$ becomes huge while exploring deeper levels on the hierarchy. Han in [11] uses associations in one level at a time. While Fortin *et al.* [12] used an oriented object representation to generate association rules of different levels of abstraction.

Zhou *et al.* in [13] represent apriori information in a DAG in order to calculate numerical dependencies (probabilities) between concepts C and items from O. Furthermore, Philips *et al.* in [14] proposed Prolog ontologies in an inference rule-based system using the first logic order language. Cevspivova *et al.* integrated ontology in the different steps of mining in the model CRIPS-DM. They publish a new miner afterwards called 4ft-Miner used in mining medical processes [15]. In [13], authors proposed a Generalization Impact measure based on support in order to estimate the sufficient level of abstraction to generate interesting generalized patterns.

Finally, in [16], a fully-blown ontology is used. First, the data sequences are enriched with domain properties from the ontology. The resulting structures are akin to labeled digraphs, in which vertices are objects and edges represent links between objects or the induced sequential order. Patterns are generalized ontological digraphs, i.e., made of sets of ontology classes connected by properties (plus sequencing). Matching between data sequences and patterns is defined on-top of the standard ontological instanciation between objects and classes and represents an order preserving injective graph morphism. The proposed *xPMiner* method applies an Apriori-like level-wise mining whereby levels in the pattern space are defined w.r.t. the overall depth of the pattern elements within the ontology. Candidates of level $k+1$ are generated from k-level frequent patterns by a *refinement operator* (see for instance [17]) made of four types of primitives: 1) add (append) a root concept to the sequence, 2) replace a concept from the sequence by a direct specialization thereof, 3) add a root property between two concepts from the sequence, and 4) replace a property by a direct specialization. A major shortage of the proposed approach is the cost of the plain patter-to-data matching operation. Indeed, as no optimization is proposed, the large number of labeled digraph morphism computations take a significant toll on the method's performances. In our own study, we choose to enhance that approach (see below).

Overall, in our own approach, data records are extracted from a unstructured sources, e.g., scientific texts. To the best of our knowledge, no ontology-based pattern mining system exists using data extracted from texts. Only text mining systems based on patterns have been proposed [6]. In what follows, we present the T-GROWLer system elicits experience in the form of generalized workflows which are mined from concrete ones, themselves acquired by text analysis of textual sources.

3 T-GROWLer, an ontology-based workflow mining system

We describe in this section the proposed methods in two modules: (A) workflow extraction from texts and (B) workflow mining (see Fig. 1). The first module requires plain text articles, a domain ontology and a processual ontology. Ontologies representing abstract concepts and relations are described to guide the information extraction processes (see Fig. 1). The main goal of module A is to extract concrete workflows from texts. In order to preserve processes order in workflows, abstract workflows are used. Abstract workflows are sequences of abstract concepts ordered by *isInputOf* and *genInputTo* relations. The first process (1 - A) extracts named entities from texts using the domain ontology schema. Re-

[1] also know as the raising operation

lation extraction is afterwards executed to create RDF triples between the elements of the processual ontology. These semantic associations enrich concrete workflows with different multi-level links. Once the processual ontology and concrete workflows are constructed, the second module (1 - B) deals with the generation of frequent patterns.

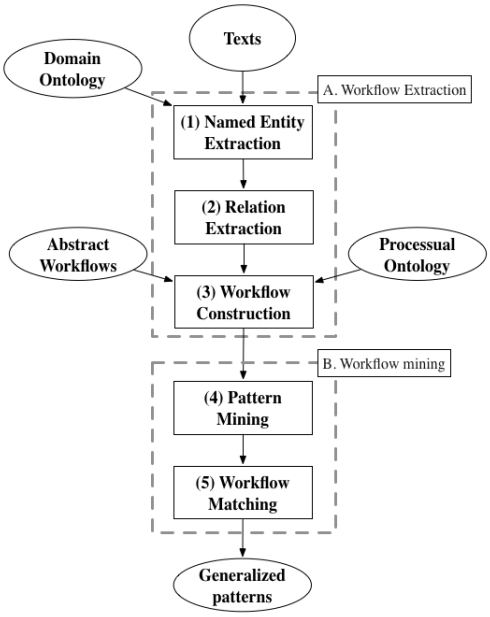

Figure 1: Generalized-workflow mining from texts.

We discuss the different steps of the workflow mining from texts in the following sections.

3.1 Workflow extraction from texts

The first module of our T-GROWLer system extracts automatically information from texts and organizes them in two ontologies: a domain ontology and a processual ontology. The first database is used in order to extract terms (named entities and relations) from texts. The second one reorganizes extracted data in a processual schema. Formally, an ontology is a four-tuple $\Omega = \langle C, \mathcal{R}, \sqsubseteq_\Omega, \rho \rangle$. C is a set of concepts, R is the set of domain relations. The generality order $\sqsubseteq_\Omega$ holds both for concepts and relations. ρ is a ternary relation $C_1 \times R \times C_2$ which connects two concepts with one relation. C_1 is called the *domain* concept and C_2 is the *range*. Our workflow extraction method takes place 3 different steps: named entity extraction, relation extraction and workflow reconstruction. We recognize and annotate automatically terms in texts with different statistical models.

We use a supervised approach in order to recognize and learn concepts from texts [18] . A handwriting specific grammar is described (in JAPE [2] language) for each ontological concept to extract and filter the right instances. For example, when a program's term is a noun (such as IDEA, NETWORK or PHASE), different semantic ambiguities are presented. We can solve this problem be modeling rules to verify if the concerned word is written in uppercase. A learning set is then constructed. In addition, we use morpho-syntactic techniques in Natural Language Processing domain such as tokenization, stemming and POS (Part Of Speech) tags to annotate texts. Two sets of features have been calculated on terms in order to learn them. These features represent term's contexts in POS annotations and surrounded concepts in a window of five next and previous concepts and tokens. A PAUM model [19] is used to learn terms with the extracted features. PAUM (Perceptron algorithm with uneven margins) was designed especially for imbalanced data and has successfully been applied to named entity recognition problems [19]. For the relation extraction module, we propose a distant supervision approach taking tuples from domain ontology as input and annotate text in a semi-supervised way [20]. We adopt this hypothesis for the relation extraction process: a link (an instance of a relation) between two concepts should exist in the same sentence evoking instances of these concepts. Such link is represented in the verb located between concepts' terms. In order to recognize tuples in texts, we designed morpho-syntactic rules for each relation. For example the relation $isAlignedWithHomologsIn$ is defined with specific terms in the context of the verb *align*. This relation should have as a domain a $DataType$ term and as a range a $Database$ term (see Fig. 5). A homologous alignment relation verb (for instance *align*) should be followed or preceded by the term *homologs* or *homologous* in the same sentence. From recognized links between named entities in text, we build a concrete workflow knowledge base where items are named entities ordered by the *isInputOf* and *genInputTo* properties (relations). Workflows as traceable graphs represent sequences of events. From the most abstract workflows, we ordered these events in order to construct a workflow database. For example, from the tuples: (ρ_1) (16sGene, "isInputOf",ClustalW), (ρ_2) (ClustalW, "genInputTo", PHYLIP) and (ρ_3) (PHYLIP, "genInputTo", TreeView), we construct the sequence $\langle$ 16sGene, ClustalW, PHYLIP, TreeView $\rangle$. This sequence is then enhanced with ontological properties and concepts from the ontology to construct a generalized workflows (see Fig. 5).

3.2 Mining generalized workflows

The second module of our system develops two major aspects of mining patterns. The first is the definition itself of a generalized workflow mining technique. The second aspect treats the definition of *frequent* patterns in the search space of workflows as directed acyclic graphs.

We developed an Aprioi approach inspired from [16] to generate generalized sub-graphs of workflows. The authors in [16] proposed two descriptive languages to represent the pattern space. Data language Δ_Ω is derived from sequences translated into objects IDs from the knowledge base Ω. The second language Γ_Ω is used to represent a pattern S in a canonical structure. Both languages represent workflows in a pair of a sequence and a triple set $S = (\zeta, \theta)$ where ζ is a sequence of concepts (or objects) and θ is a set of relations. Level-wise descent relies on the generation of $k+1$-level pat-

[2]JAPE is a Java Annotation Patterns Engine. JAPE provides nite state transduction over annotations based on regular expressions. https://gate.ac.uk/sale/tao/splitch8.html

terns from k-level ones, which amounts to the computation of four canonical operations in a depth-first search strategy. To generate candidates, we apply four different canonical operations on the root concepts from the ontology: (1) AddCLS - adds a concept to a pattern, (2) AddREL - adds a relation between two concepts, (3) SplCLS - instantiates a concept and (4) SplREL - instantiates a relation in a pattern. For instance, AddREL adds a property between two concepts (a domnain and a range) in a workflow pattern. Concepts subject and object could be in different positions in the pattern S. Specializing a concept/relation replaces the concerned entity with a lower abstract event. For each pattern, 4 different canonical operations are used. Although these operations constitute an effective generation mechanism, their unrestricted use is redundancy-prone and hence would harm the efficiency of the global mining process. In the previous work of [16], calculating the support of a pattern S amounts to match this pattern with all the workflow concrete sequences in the database.

For this purpose, we introduce a novel data structure to prune patterns while calculating their supports. Calculating frequent patterns S begins from the last visited concept/relation of a parent pattern. Then, we don't need to recalculate all the matching possibilities for each concept and relation previously matched.

We remind that a sequence in a workflow is obtained from the order of *isInputOf* and *genInputTo* links. The non-strict partial order of events is obtained by following the most abstract pattern relations in the ontology Ω (see Fig. 5). For instance, the sequence s in Fig. 2 is a poset $s.\zeta$ of events $i \in 1,..7$ and a set $s.\theta$ of links $l \in 1,..6$. A pattern S is represented as well with a poset of concepts in $S.\zeta$ and a set of relations $S.\theta$. For an iteration k, S is matched with s in ζ: $i1 \sqsubseteq C10$, $i4 \sqsubseteq C11$, $i6 \sqsubseteq C14$ and $i7 \sqsubseteq C15$. S is matched as well with s in θ.

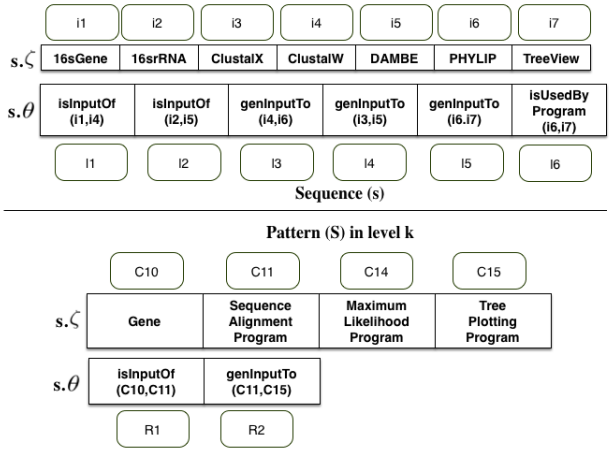

Figure 2: Workflow sequence S and pattern S representations.

The proposed matching structure MS is a "stack" results of the canonical operations (adding/specializing concepts/relations) while extending a pattern. For example, the arrows in Fig .3 shows the way how we obtain a pattern S. The result of the last operation in this case is *genInputTo*.

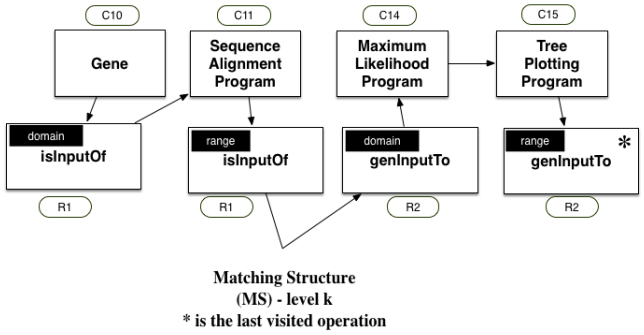

Figure 3: Matching structure of a pattern S.

If the next operation in the $k + 1^{th}$ iteration is adding the relation $R3(C14, C15)$ (see Fig. 4), then the new data structure of S should put the relation in the proper position. Our matching data structure adds the domain and the range properties by linking the concerned concepts.

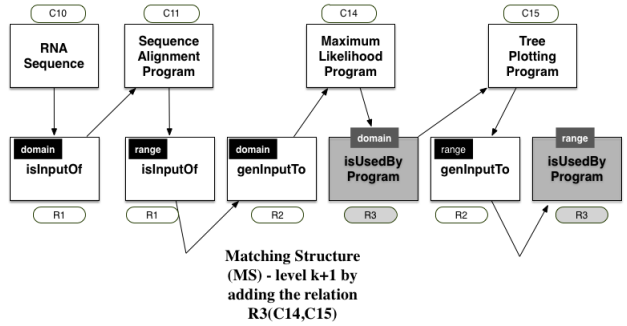

Figure 4: Extension of a pattern S by adding a relation $R3$ (grey boxes).

Precedence relation within patterns considers matching solutions structure to calculate pattern's supports. Our proposed data structure considers pattern generation order and then we don't have to recalculate all the matched concrete workflows in each iteration. With the anti-monotony characteristic of the Apriori approach, our algorithm guarantees that all frequent patterns are generated.

4 Case study: phylogenetic workflows

Phylogeny represent evolutionary history of a group of organisms having a common ancestor. Inferred from nucleic acid (DNA or RNA) or protein sequences, phylogenetic relations are discovered with tools detecting different evolutionary phenomena such as duplications, insertions and deletions [7]. Furthermore, during phylogenetic tasks, multiple bioinformatics tools and methods are used [21] which complicates the comparison of different studies even for the same species. Platforms like Phylogeny.fr, Taverna or Armadillo [22] propose different tools and web services. However each platform represents a completely different level of phylogenetic

knowledge: from workflow based-systems to web services and programming language packages. However, the complex nature of bioinformatics data (as heterogeneous, spread and dynamic) makes the knowledge discovery task very difficult.

Phylogenetic workflows follow a general pipeline of processes: (1) data collection, (2) homology search and (3) phylogenetic inference. In the first step, a molecular data (sequences or alignments) of a set of species (taxa) is collected. In the second step, sequences are aligned with homologs ("similar sequences"). In the third step, phylogenetic trees describing evolutionary history of the concerned taxa are inferred. This general pipeline have been used in the construction of workflows from texts. Extracted relation instances should be ordered following this general workflow. We downloaded scientific articles from the digital repository PubMed Central (PMC) [3] database (articles published from 2013 until April 2015) in a XHTML format. Over 2,000 texts have been extracted. In addition, we constructed a terminology gazetteer from different well-known databases in the phylogenetic literature such as NCBI Unigene, Uniprot, Taxonomy Browser and KEGG Disease [4]. The terminology is then reorganized in a domain ontology in order to recognize terms and relations in texts like presented in module A of the proposed model (see Fig. 1).

Next, we built a processual ontology to discover workflow patterns within annotated terms in texts. We present in Fig. 5 a sample from the proposed processual phylogenetic ontology.

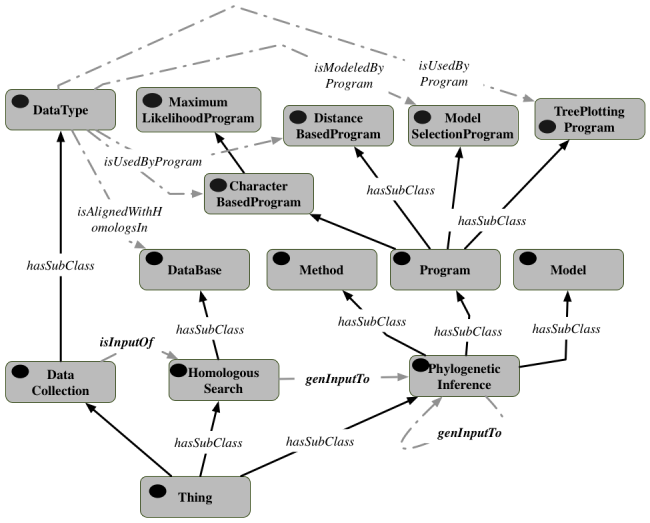

Figure 5: A portion from the phylogenetic processual ontology (PhylOntology). On the bottom, we see abstract concepts ordered by the *isInputOf* and the *genInputTo* properties. We describe the phylogenetic inference in 3 big steps: $DataCollection$, $HomologousSearch$ and $PhylogeneticInference$. Each class/concept contains different sub-classes related to each other with different levels of relations. For example, a $DataType$ (*e.g.* DNA sequence) is aligned in a $Database$ used by a $CharacterBasedProgram$, a $TreePlottingProgram$ and modeled by a $ModelSelectionProgram$.

5 Experiments and results

5.1 Named entity learning results

We used a PAUM model to recognize the different phylogenetic concepts in texts. The Gate implementation of PAUM (with default parameters) is used to evaluate our supervised machine learning annotation system. The learning set is constructed from our gazetteer terminology. A specific grammar for each concept is also modeled. A cross validation of 10 folds presents results in terms precision and recall (see table 1). Our proposed model with context features has well learned and applied 7 different domain concepts: DataType, Taxon, Gene, Protein, Disease, Method and Program. DataType is a concept including different data type sequences: DNA, RNA, protein and alignments. 9366 terms have been extracted from the phylogenetic analyses sections (with 95% of F-measure). The overall F-measure for both sections (abstracts and phylogenetic analyses) is 96%. Indeed, with only context POS annotations and concept neighbors, we could recognize the 7 different classes. Specific vocabularies and gazetteers are able to distinguish different concepts in the texts.

A gold standard list of terms and a corresponding acquisition corpus were built. An expert has extracted manually about 1,000 terms (and classified them into the different concepts) from 100 phylogenetic sections to build a reference terminology. Each extracted term has been compared to the reference entities in order to validate its domain's precision and recall (see figure 6).

Evaluating extracted terms with the expert reference terminology shows that only 6 classes have been well extracted by our PAUM model. Extracted diseases terms didn't recognized by the expert. This is expected, since diseases terms aren't frequent in phylogenetic analyses sections. Only 6 terms have been extracted with our model and 3 are recognized by the expert. However, for the other concept's instances, our model did recognize terms with 73.95 % of F-measure.

We didn't implement yet our relation extraction system. Thus, only 50 workflows have been manually built in order to execute the generalized pattern mining phase. Concepts and properties terms are then organized in the processual ontology. This latter presents 13 different object properties (links), 68 concepts (with 3 concept roots) and 560 individuals (terms). Our workflow database is composed from 50 workflows with an average of 11 items (terms) and 9 links per workflow. Our workflow database PhyloFlows and the processual ontology represent the input of the next module (B) of the proposed model (see Fig. 1). We present in the next section the different results of the workflow-pattern mining process.

5.2 Generalized-pattern results

In order to evaluate our pattern mining system T-GROWLer, we present different experimentations. We tested the new

[3] http://www.ncbi.nlm.nih.gov/pmc/

[4] More than 1,800,000 terms have been extracted.

Table 1: Evaluation results of the automatic annotation system on phylogenetic analyses sections - extracted from articles published from 2013 to April 2015

Concepts	N.Instances	Precision	Recall
DataType	9366	0.97	0.92
Taxon	8867	0.98	0.94
Gene	922	1.00	0.95
Protein	485	0.99	0.97
Disease	6	0.30	0.30
Method	2671	0.99	0.96
Program	2132	0.99	0.90
Overall	24449	0.98	0.93

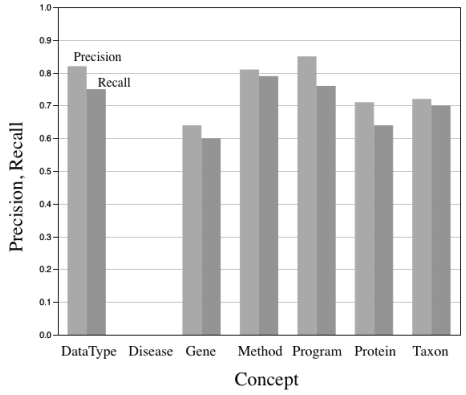

Figure 6: Gold standard evaluation results.

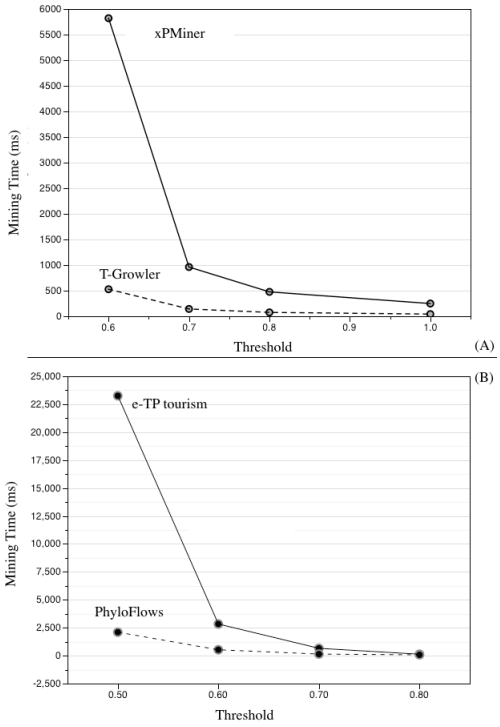

Figure 7: (A) Comparing the mining time of T-Growler and xPMiner. (B) Varying support for PhyloFlows and e-TP tourism databases.

matching algorithm in terms of speed (mining time) and scalability (see Fig. 7). T-GROWLer outperforms the xPMiner system [16] with 2 times order of magnitude in high threshold supports (>0.8). With 60% of minimum support, our approach executes 10 times faster the mining algorithm than the xPMiner algorithm.

We also tested our method with a different database. eTP-tourism (Electronic Tourism Platform) worfklow-base is a portion of a log file describing web pages navigation of 56 different users in a Web travel site [16]. This database has an average of 19 items and 6 links per sequence and enhanced by an ontology of 157 concepts and 35 object properties. Fig. 7- B shows the difference in mining time between PhyloFlows and eTP-tourism. Running time does eventually increase when more concepts and links are used. Our system spends about 10 times the mining time to generate patterns with 500 items and 100 links more. While eTP-tourism database presents more dense graphs, matching sub-graphs and mining processes become heavier to handle. However, taking 23 seconds to do the whole process of matching and generating about 6,000 graph patterns is still a reasonable time.

We present, next, detailed results obtained from the PhyloFlows database. We remind that a pattern S is a set of concepts and relations. Relations are couples of domain and range concepts. For instance:$(1, 2) = [isAlignedIn]$ represents the relation $isAlignedIn$ between the 1^{st} and the 2^{nd} concept in the pattern S. We present next, a sample of 2 different generated patterns.

1. Concepts=[DataCollection, MultipleSequenceAlignment, PhylogeneticInference] ; Relations={(1, 3)=[isInferredBy], (1, 2)=[genInputTo], (2, 3)=[genInputTo]} ; Threshold = 80%
2. Concepts=[DNASequence, HomologySearch, GTR-Model, CharacterBasedProgram, TreePlottingProgram ; Relations={(1, 2)=[isAlignedIn], (1, 3)=[isModeledBy], (1, 4)=[isUsedByProgram], (1, 5)=[isUsedByProgram], (1, 2)=[isInputOf], (2, 3)=[genInputTo], (3, 4)=[genInputTo], (4, 5)=[genInputTo]} ; Threshold = 40%

All generated patterns present different levels of concepts and relations.. For example, in 80% of the cases, we find that a phylogenetic analysis should have at least three ordered steps: data collection, multiple sequence alignment and a phylogenetic inference program. However, these kinds of patterns are generally well known by the community. While decreasing the threshold support, we find more interesting patterns like when selecting DNA sequences, a homology search is done with a GTR model for a character based inference method and a visualization tool (pattern 2). This sequence is enhanced with different kinds of links like which alignment program is used or which kind of inference method has been adopted in these kinds of studies.

These kinds of patterns could be used for a better recommendation considering their abstract but semantically

richer meanings. For example, if we know that the $16s\ rRNA$ gene is used by the $ClustalW$ alignment program in 10% of the cases, then a concrete recommendation could offer $MrBayes$ as phylogenetic reconstruction program. However, offering any $Character - based : Inference\ Program$ is a more interesting recommendation due to its higher frequency (40%) in the database as it is a more "general" concept.

6 Conclusion

Our proposed approach of mining generalized patterns from texts proves that different kinds of knowledge could be extracted from the phylogenetic literature. Expertise and domain knowledge from articles show interesting patterns to mine. Different ontologies have been used to represent the different types of knowledge into concepts and relations. Extracted workflows are enhanced with ontologies' annotations to produce generalized frequent patterns. These patterns, covering a wider range of values then specific terms, could be used to assist practitioners with a richer vocabulary during their tasks. Our proposed model could also be applied to different domains when expertise is confronted to domain knowledge. Video games, for example, is a good domain application to our approach when gamers are constantly faced to cognitive gestures. Workflows of events could be extracted from games sessions and mined in order to recommend and predict generalized patterns like what type of event is going to happen during a game or what kind of item a player needs to buy to accomplish a mission.

References

[1] W. van der Aalst, T. Weijters, and L. Maruster. Workflow mining: Discovering process models from event logs. *IEEE Trans. on Knowl. and Data Eng.*, 16(9):1128–1142, September 2004.

[2] C. A. Ellis, A. J. Rembert, K.-H. Kim, and J. Wainer. Beyond workflow mining. In S. Dustdar, J. L. Fiadeiro, and A. P. Sheth, editors, *Business Process Management*, volume 4102 of *Lecture Notes in Computer Science*, pages 49–64. Springer, 2006.

[3] W. M. P. van der Aalst and A. Weijters. Process mining: a research agenda. *Computers in Industry*, 53:231–244, 2004.

[4] N. Hashmi. Abstracting workflows: unifying bioinformatics task conceptualization and specification through semantic web services. *on Semantic Web for Life.*, (October):27–28, 2004.

[5] W. Dong, W. Fan, L. Shi, C. Zhou, and X. Yan. A general framework to encode heterogeneous information sources for contextual pattern mining. *Proceedings of the 21st ACM int. conference on Information and knowledge management - CIKM '12*, page 65, 2012.

[6] Y. Tsuruoka, Y. Tateishi, J.-D. Kim, T. Ohta, J. McNaught, S. Ananiadou, and J. Tsujii. Developing a robust part-of-speech tagger for biomedical text. In *Advances in informatics*, pages 382–392. Springer, 2005.

[7] M. Anisimova, D. a. Liberles, H. Philippe, J. Provan, T. Pupko, and A. von Haeseler. State-of the art methodologies dictate new standards for phylogenetic analysis. *BMC evolutionary biology*, 13:161, January 2013.

[8] C. C. Aggarwal and J. Han, editors. *Frequent Pattern Mining*. Springer, 2014.

[9] S. S. A. Bamshad Mobasher. The role of domain knowledge in data mining. In *Proceedings of the 4th int. conference on Information and knowledge management*, pages 37–43. ACM, 1995.

[10] R. Srikant and R. Agrawal. Mining sequential patterns: Generalizations and performance improvements. In *Proceedings of the 5th int. Conference on Extending Database Technology: Advances in Database Technology*, EDBT '96, pages 3–17, London, UK, UK, 1996. Springer-Verlag.

[11] J. Han and Y. Fu. Discovery of multiple-level association rules from large databases. In *VLDB*, volume 95, pages 420–431, 1995.

[12] S. Fortin and L. Liu. An object-oriented approach to multi-level association rule mining. In *Proceedings of the fifth int. conf. on Information and knowledge management*, pages 65–72. ACM, 1996.

[13] X. Zhou and J. Geller. Raising, to enhance rule mining in web marketing with the use of an ontology. *Data Mining with Ontologies: Implementations, Findings and Frameworks*, pages 18–36, 2007.

[14] J. Phillips and B. G. Buchanan. Ontology-guided knowledge discovery in databases. In *Proceedings of the 1st int. conf. on Knowledge capture*, pages 123–130. ACM, 2001.

[15] H. Cešpivová, J. Rauch, V. Svatek, M. Kejkula, and M. Tomeckova. Roles of medical ontology in association mining crisp-dm cycle. In *ECML/PKDD04 Workshop on Knowledge Discovery and Ontologies (KDO04), Pisa*. Citeseer, 2004.

[16] M. Adda, P. Valtchev, R. Missaoui, and C. Djeraba. Toward recommendation based on ontology-powered web-usage mining. *IEEE Internet Computing*, 11(4):45–52, 2007.

[17] L. De Raedt. Logical and relational learning. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 5249 LNAI, page 1, 2008.

[18] D. Nadeau and S. Sekine. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26, 2007.

[19] Y. Li, K. Bontcheva, and H. Cunningham. Using uneven margins svm and perceptron for information extraction. In *Proceedings of the Ninth Conference on Computational Natural Language Learning*, CONLL '05, pages 72–79, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics.

[20] N. Bach and S. Badaskar. A survey on relation extraction. *Language Technologies Institute, Carnegie Mellon University, www. ark. cs. cmu. edu/LS2/images/9/97/BachBadaskar*, 2007.

[21] J. W. Leigh, F. J. Lapointe, P. Lopez, and E. Bapteste. Evaluating phylogenetic congruence in the post-genomic era. *Genome Biology and Evolution*, 3:571–587, 2011.

[22] E. Lord, M. Leclercq, A. Boc, A. B. Diallo, and V. Makarenkov. Armadillo 1.1: An Original Workflow Platform for Designing and Conducting Phylogenetic Analysis and Simulations. *PloS one*, 7(1):e29903, January 2012.