

Workflow Analysis using Graph Kernels

Natalja Friesen and Stefan Rüping

Fraunhofer IAIS

Schloss Birlinghoven, St. Augustin, Germany

{natalja.friesen, stefan.rueping}@iais.fraunhofer.de

Abstract

Workflow enacting systems are a popular technology in business and e-science alike to flexibly define and enact complex data processing tasks. Since the construction of a workflow for a specific task can become quite complex, efforts are currently underway to increase the re-use of workflows through the implementation of specialized workflow repositories. While existing methods to exploit the knowledge in these repositories usually consider workflows as an atomic entity, our work is based on the fact that workflows can naturally be viewed as graphs. Hence, in this paper we investigate the use of graph kernels for the problems of workflow discovery, workflow recommendation, and workflow pattern extraction, paying special attention on the typical situation of few labeled and many unlabeled workflows. To empirically demonstrate the feasibility of our approach we investigate a dataset of bioinformatics workflows retrieved from the website myexperiment.org.¹

1 Introduction

Workflow enacting systems are a popular technology in business and e-science alike to flexibly define and enact complex data processing tasks. A workflow is basically a description of the order in which a set of services have to be called with which input in order to solve a given task. Since the construction of a workflow for a specific task can become quite complex, efforts are currently underway to increase the re-use of workflows through the implementation of specialized workflow repositories.

Driven by specific applications, a large collection of workflow systems have been prototyped such as Taverna [Oinn *et al.*, 2004] or Triana [Taylor *et al.*, 2006].

As the high numbers of workflows can be generated and stored relatively easily it becomes increasingly hard to keep an overview about the available workflows. Workflow repositories and websites such as myexperiment.org tackle this problem by offering the research community the possibility to publish and exchange complete workflows. An even higher amount of integration has been described in the idea of developing a Virtual Research Environment (VRE, [Fraser, 2005]).

Due to the complexity of managing a large repository of workflows, data mining approaches are needed to sup-

port the user in making good use of the knowledge that is encoded in these workflows. In order to improve the flexibility of a workflow system, a number of data mining tasks can be defined:

Workflow recommendation: Compute a ranking of the available workflow with respect to their interestingness to the user for a given task. As it is hard to formally model the user's task and his interest in a workflow, one can also define the task of finding a measure of similarities on workflows. Given a (partial) workflow for the task the user is interested in, the most similar workflows are then recommended to the user.

Metadata extraction: Given a workflow (and possibly partial metadata), infer the metadata that describes the workflow best. As most approaches for searching and organizing workflow are based on descriptive metadata, this task can be seen as the automatization of the extraction of workflow semantics.

Pattern extraction: Given a set of workflows, extract a set of sub-patterns that are characteristic for these workflow. A practical purpose of these patterns is to serve as building block for new workflows. In particular, given several sets of workflows, one can also define the task of extracting the most discriminative patterns, i.e. patterns that are characteristic for one group but not the others.

Several approaches to these problems exists in the data mining literature, in particular on machine learning for recommender systems [Goderis, 2008]. However, existing methods usually consider workflows as an atomic entity, using workflow meta data such as its usage history, textual descriptions (in particular tags), or user-generated quality labels as descriptive attributes. While these approaches can deliver high quality results, they are limited by the fact that all these attributes require either a high user effort to describe the workflow (to use text mining techniques), or a frequent use of each workflow by many different users (to mine for correlations).

Figure 1 shows the evolution of information about a workflow over its lifetime. In the construction phase, mainly technical information about the workflow itself is generated. Possibly starting from a workflow pattern, several versions of a workflow are generated and tested. In the submission phase, the workflow is made public for possible re-use. Here, the owner gives a more or less detailed description about the purpose of the workflow and about the task that it is supposed to achieve. In the final phase, the re-use of the workflow by different users begins, generating information about workflow usage, such as correlations

¹This paper also appears at the Workshop on Service-Oriented Knowledge Discovery (SoKD 2010)

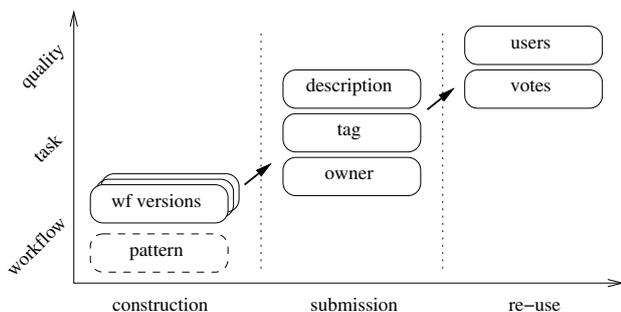


Figure 1: Workflow data: over the lifetime of a workflow (horizontal axis), information on different levels of generality is generated (vertical axis)

with usage of other workflows. User possibly vote on the quality of the workflow.

In this paper we are interested in supporting the user in constructing the workflow and reducing the manual effort of workflow tagging. The reason for the focus on the early phases of workflow construction is that in practice it can be observed that often users are reluctant to put too much effort into describing a workflow; they are usually only interested in using the workflow system as a means to get their work done. A second aspect to be considered is that without proper means to discover existing workflows for re-use, it will be hard to receive enough usage information on a new workflow to start up a correlation-based recommendation in the first place.

To address these problems, we have opted to investigate solutions to the previously described data mining tasks that can be applied in the common situation of many unlabeled workflows, using only the workflow description itself and no meta data. Our work is based on the fact that workflows can be viewed as graphs. We will demonstrate that by the use of graph kernels it is possible to effectively extract workflow semantics and use this knowledge for the problems of workflow recommendation and metadata extraction. The purpose of this paper is to answer the following questions:

- Q1:** How good are graph kernels at performing the tasks of workflow recommendation without explicit user input? We will present an approach that is based on exploiting workflow similarity.
- Q2:** Can appropriate meta data about a workflow be extracted from the workflow itself? What can we infer about the semantics of a workflow and its key characteristics? In particular, we will investigate the task of tagging a workflow with a set of user-defined keywords.
- Q3:** How good does graph mining perform at a descriptive approach of workflow analysis, namely the extraction of meaningful graph patterns?

The remainder of the paper is structured as follows: Next, we will discuss related work in the area of workflow systems. In Section 3, we give a detailed discussion of representation of workflows and the associated metadata. Section 4 will present the approach of using graph kernels for workflow analysis. The approach will be evaluated on 4 distinct learning tasks on a dataset of bioinformatics workflows retrieved from the website <http://myexperiment.org> in Section 5. Section 6 concludes.

2 Related Work

Since workflow systems are getting more complicated, the development of effective discovery techniques particularly for this field has been addressed by many researcher during the last years. Public repositories that enables sharing of workflows are widely used both in business and scientific communities. While first steps toward supporting the user have been made, there is still a need to improve effectiveness of discovery methods and support the user in navigating in the space of available workflows. A detailed overview of different approaches for workflow discovery is given by Goderis [Goderis, 2008].

Most approaches are based on simple search functionalities and consider workflow as an atomic entity. Searching over workflow annotation like titles, textual description or discovery on the basis of user profiles belongs to basic capabilities of such academic repositories as myExperiment [Roure, 2009], BioWep², Kepler³ or commercial systems like Infosense and Pipeline Pilot.

In [Goderis *et al.*, 2009] a detailed study about current practices in workflow sharing, re-using and retrieval is presented. To summarize, the need to take into account structural properties of workflows in the retrieval process was underlined by several users. The authors demonstrate that existing techniques are not sufficient and there is still a need for effective discovery tools. In [Goderis *et al.*, 2006] retrieval techniques and methods for ranking discovered workflows based on graph-subisomorphism matching are presented. Coralles [Corrales *et al.*, 2006] proposes a method for calculating the structural similarity of two BPEL (Business Process Execution Language) workflows represented by graphs. It is based on error correcting graph subisomorphism detection.

Apart from workflow sharing and retrieval, the design of new workflows is an immense challenge to users of workflow systems. It is both time-consuming and error-prone, as there is a great diversity of choices regarding services, parameters, and their interconnections. It requires the researcher to have specific knowledge in both his research area and in the use of the workflow system. Consequently, it is preferable for a researcher to not start from scratch, but to receive assistance in the creation of a new workflow.

A good way to implement this assistance is to reuse or re-purpose existing workflows or workflow patterns (i.e. more generic fragments of workflows). An example of workflow re-use is given in [Goderis, 2008], where a workflow to identify genes involved in tolerance to Trypanosomiasis in East African cattle was reused successfully by another scientist to identify the biological pathways implicated in the ability of mice to expel the *Trichuris Muris* parasite.

In [Goderis *et al.*, 2005] it is argued that designing new workflows by reusing and re-purposing previous workflows or workflows patterns has the following advantages:

- Reduction of workflow authoring time
- Improved quality through shared workflow development
- Improved experimental provenance through reuse of established and validated workflows
- Avoidance of workflow redundancy

²<http://bioinformatics.istge.it/biowep/>

³<https://kepler-project.org/>

While there has been some research comparing workflow patterns in a number of commercially available workflow management systems [Van Der Aalst *et al.*, 2003] or identifying patterns that describe the behavior of business processes [White, March 2004], to the best of our knowledge there exists no work to automatically extract patterns. A pattern mining method for business workflows based on calculation of support values is presented in [Thom *et al.*, 2007]. However, the set of patterns that was used was derived manually based on an extensive literature study.

3 Workflows

A workflow is a way to formalize and structure complex data analysis experiments. Scientific workflows can be described as a sequence of computation steps together with predefined input and output that arise in scientific problem-solving. Such definition of workflow enables sharing analysis knowledge within scientific communities in convenient way. Since we want to provide the user with suggestions about possibly interesting workflows, we need to be able to compare workflows available in existing VREs with the one that the user has. That implies that workflows need a consistent representation.

3.1 Example: myExperiment.org

We consider the discovery of similar workflow in the context of a specific VRE called myExperiment [Roure *et al.*, 2008], MyExperiment has been developed to support sharing of scientific objects associated with an experiment. It is a collaborative environment where scientists can safely publish their workflows and experiment plans, share them with groups and find those of others. MyExperiment is designed to make it easy for scientists to contribute to a pool of scientific workflows, build communities and form relationships. Its goal is to enable scientists to share and reuse workflows, reduce the time-to-experiment, and avoid reinvention.

Figure 2 shows a workflow view in myExperiment. Each stored workflow is created by a specific user, is associated with a workflow graph, and contains metadata and certain statistics such as the number of downloads or rating. We split all available information about a workflow into four different groups: the workflow graph, textual data, user information, and workflow statistics. Next we will characterize each group in more detail. Graph properties of workflow are detailed described in 4

3.2 Textual Data

Each workflow in myExperiment has a title and a description text and contains information about the creator and date of creation. Furthermore, the associated tags annotate workflow by several keywords that facilitate searching for workflows and provide more precise results. Additionally, for workflows in Taverna format myExperiment displays visual previews, detailed information about single workflow components, such as type of inputs and outputs, number and type of certain processors.

3.3 User Information

MyExperiment was created not only as an environment for sharing workflows, but also as a social infrastructure for the researchers. The social component is realized by registration of users and allowing them to create profiles with different kind of personal information, details about their work and professional life. The members of myExperiment

can form complex relationships with other members, such as creating or joining user groups or giving credit to others. All this information can be used in order to find the groups of users having similar research interests or working in related projects. In the end, this type of information can be used to generate the well known correlation-based recommendations of the type “users who liked this workflow also liked the following workflows...”.

3.4 Workflow Statistics

As statistic data we consider information that is changing with the time, such as the number of views or downloads or the average rating. Statistic data can be very useful for providing a user with a workflow he is likely to be interested in. As we do not have direct information about user preferences, some of the statistics data, e.g. number of downloads or rating, can be considered as a kind of quality measure.

4 A Graph Mining Approach to Workflow Analysis

The characterization of a workflow by metadata alone is challenging because neither of these features give an insight into the underlying sub-structures of the workflow. It is clear that users do not always create a new workflow from scratch, but most likely re-use old components and sub-workflows. Hence, knowledge of sub-structures is important information to characterize a workflow completely.

The common approach to represent objects for a learning problem is to describe them as vectors in a feature space. However, when we handle objects that have important sub-structures, such as workflows, the design of a suitable feature space is not trivial. For this reason, we opt to follow a graph mining approach.

4.1 Frequent Subgraphs

Frequent subgraph discovery has received intense and still growing attention, since it has a wide range of applications areas. Frequently occurring subgraphs in a large set of graphs could represent important motifs in the data. Given a set of graphs \mathcal{G} , the support of a graph G ($S(G)$) is defined as the fraction of graphs in \mathcal{G} in which G occurs. The problem of finding frequent patterns is defined as follows:

Given a set of graphs \mathcal{G} and minimum support S_{min} . We want to find all connected subgraphs G that occur frequently enough (i.e. $S(G) \geq S_{min}$) over the entire set of graphs. The output of the discovery process may contain a large number of such patterns.

4.2 Graph Kernels

Graph kernels, as originally proposed by [Gaertner *et al.*, 2003; Kashima and Koyanagi, 2002] provide a general framework for handling graph data structures by kernel methods. Different approaches for defining graph kernels exist.

A popular representation of graphs that is used for examples in protein modeling and drug screening are kernels based on cyclic patterns [Horváth *et al.*, 2004]. However, these are not applicable to workflow data, as workflows are by definition acyclic (because an edge between services A and B represents the relation “A must finish before B can start”).

To adequately represent the decomposition of workflows into functional substructures, we apply the following approach: the set of graphs is searched for substructures (subtrees) that occur in at least a given percentage (support)

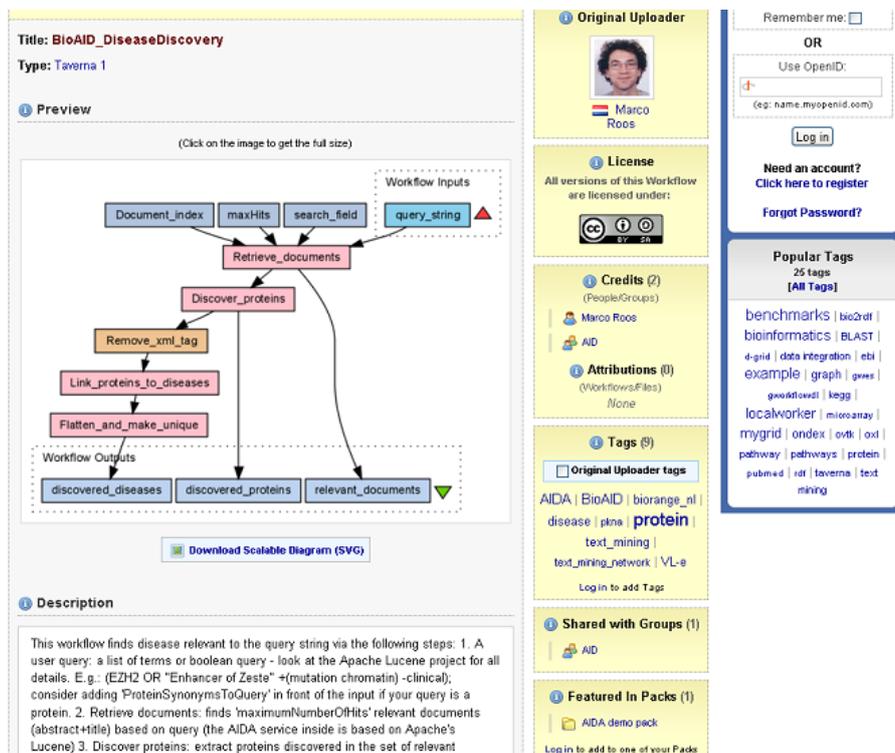


Figure 2: MyExperiment: workflow with associated metadata.

of all graphs. Then, the feature vector is composed of the weighted counts of the substructures. The substructures are sequences of labeled vertices that were produced by graph traversal. The length of a substructure is equal to the number of vertices in it. This family of kernels is called Label Sequence Kernels. The main difference among the kernels lies in how graphs are traversed and how weights are involved in computing a kernel. According to the extracted substructures, these are kernels based on walks, trees or cycles. In our work we used the kernels based on random walks with exponential weights proposed by Gärtner et al. [Gaertner *et al.*, 2003]. Since workflows are directed acyclic graphs, in our special case the hardness results of [3] (such random walks can not be enumerated in general case) no longer hold and we actually can enumerate all walks. This allows us to explicitly generate the feature space representation of the kernels by defining the attribute values for every substructure (walk). For each substructure s in the set of graphs, let k be the length of the substructure. Then, the attribute λ_s is defined as:

$$\lambda_s = \frac{\beta^k}{k!} \quad (1)$$

(and thus generate the explicit finite set of features directly).

if the graph contains the substructure s and $\lambda_s = 0$ else. Here β is a parameter that we have optimized by cross-validation.

The graph kernel between graphs G_1 and G_2 is now defined as

$$k(G_1, G_2) = \sum_{s \in \text{sub}(G_1) \cap \text{sub}(G_2)} \lambda_s$$

where the sum runs over all substructures s that are present in both G_1 and G_2 . Note that if for a finite set of d substructures s_1, \dots, s_d we define

$$\lambda(G) = (\sqrt{\lambda_{s_1}} \mathbf{1}_{\{s_1 \in G\}}, \dots, \sqrt{\lambda_{s_d}} \mathbf{1}_{\{s_d \in G\}})$$

this is equivalent to

$$k(G_1, G_2) = \lambda(G_1)^T \lambda(G_2).$$

A very important advantage of graph kernels approach for the discovery task is that distinct substructures can provide an insight into the specific behavior of the workflow.

4.3 Graph Representation of Workflows

A workflow can be formalized as a directed acyclic labeled graph. The workflow graph has two kind of nodes: regular nodes representing the computation operations and nodes defining input/output data structure. A set of edges shows information and control flow between the nodes. More formally, a workflow graph can be defined as a tuple $W = (N, T)$, where:

$$N = \{C, I, O\}$$

C = finite set of computation operations,

I/O = finite set of inputs or outputs

$T \subseteq N \times N$ = finite set of transitions defining control flow.

Labeled graphs contain an additional source of information. There are several alternatives to obtain node labels. On the one hand, users often annotate single workflow components by a combination of words or abbreviations. On the other hand, each component within workflow system has a signature and an identifier associated with it, e.g. in web-service WSDL format. User created labels suffer from subjectivity and diversity, e.g. the same node representing the same computational operation can be labeled in very different way. The first alternative again assumes some type of user input, so we opt to use the second alternative.

Figure 3 shows an example of such transformation obtained for a Taverna workflow [Oinn *et al.*, 2004]. While the left pictures shows a user annotated components the

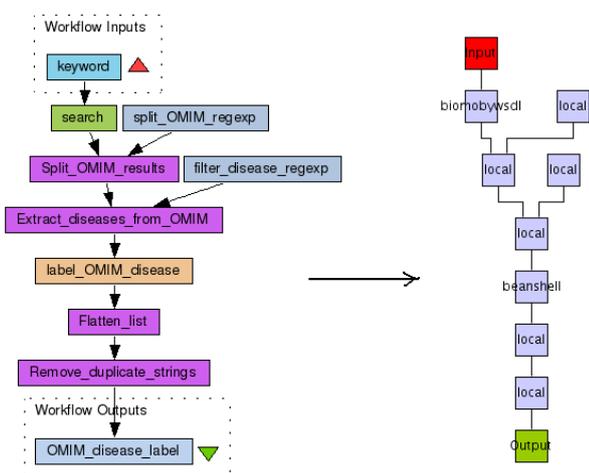


Figure 3: Transformation of Taverna workflow to the workflow graph.

right picture presents workflow graph on the next abstraction level.

5 Evaluation

In this section we illustrate the use of workflow structure and graph kernels in particular for workflow discovery and pattern extraction. We evaluate results on a real-world dataset of Taverna workflows. However, the same approach can be applied to other workflow systems, as long as we can obtain meaningful consistent labels from workflows.

5.1 Dataset

For the purposes of this evaluation we used a corpus of 300 real-world bioinformatics workflows retrieved from myExperiment [Roure *et al.*, 2008]. We chose to restrict ourselves to workflows that were created in Taverna workbench [Oinn *et al.*, 2004] in order to simplify the formatting of workflows. Since the application area of myExperiment is restricted to bioinformatics, we may be sure that there are sets of similar workflows there. The user feedback about similarity of workflow pairs is missing. Hence, we used semantic information to obtain workflows similarity. We made the assumption that workflows targeting the same tasks are similar. Under this assumption we used the cosine similarity of the vector of tags assigned to the workflow as a proxy for the true similarity. An optimization over the number of clusters resulted in 5 groups shown in Table 1. These tags indeed impose a clear structuring with few overlaps on the workflows.

5.2 Workflow Recommendation

In this section, we address Question **Q1**: How good are graph kernels at performing the tasks of workflow recommendation without explicit user input? The goal is to retrieve workflows that are "close enough" to a user's context. Therefore, we need to be able to compare workflows available in existing VREs with the user's one. As similarity measure we use graph kernel, which is equivalent to the cosine distance between the feature vectors consisting of the weighted counts of the frequent subgraphs.

We compare our approach based on graph kernels to several techniques representing the current state of the art

[Goderis *et al.*, 2006]: matching of workflow graphs based on the size of the maximal common subgraph (MCS) and a method that considers a workflow as a bag of services. In addition to these techniques we also consider a standard text mining approach, whose main idea is that workflows are documents in XML format. The similarity of a workflow pair is then calculated as the cosine distance between the respective word vectors.

In our experiment we predict if two workflows belong to the same cluster. Table 2 summarizes the average performances of a leave-one-out evaluation for the four approaches. It can be seen that graph kernels clearly outperform all other approaches in accuracy and recall. For precision, MCS performs best, however, at the cost of a minimal recall. The recall of graph kernels ranks second and is close to the value of MCS.

Therefore, we conclude that graph kernels are suitable for the task of workflow recommendation based only on graph structure without explicit user input.

5.3 Workflow Tagging

We are now interested in Question **Q2** of extraction of appropriate metadata from workflows. As a prototypical piece of metadata, we investigate user-defined tags.

20 tags were selected that occur in at least 3% of all workflows. We use tags as proxies that represent the real-world task that a workflow can perform. Selected tags present 20 different classification problems. For each tag we would like to predict if it describes a given workflow. To do that we utilize graph kernels representation of workflow. We tested two algorithms: SVM and k-Nearest Neighbor. Table 3 shows results of tags prediction evaluated by 2-fold cross validation over 20 keywords. It can be seen that an SVM with graph kernels can predict the selected tags with high AUC and precision, while a Nearest Neighbor approach using graph kernels to define the distance achieves a higher recall.

We can conclude that the graph representation of workflow contains enough information to predict appropriate metadata.

5.4 Pattern extraction

Finally, we investigate question **Q3**, which deals with the more descriptive task of extracting meaningful patterns from sets of workflows that are helpful in the construction of new workflows.

We address the issue of extracting patterns that are particularly important within a group of similar workflows in several steps. First, we use a linear SVM to build a classification model based on the graph kernels. This model identifies all workflows which belong to the same group against workflows from other groups. Then we search for features having high weight value which the model considers as important. We performed such pattern extraction targeting consequently each workflow group. A 10-fold accuracy shows that this classification can be achieved with high accuracy, values ranging between 81.3% and 94.7%, depending on the class. However, we are more interested in the most significant patterns, which we determine based on the weight that was assigned by the SVM (taking different standard deviation into account). Examples of such patterns are described below.

Figure 4 shows an example of workflow patterns and the same pattern inside of one of workflows that it occurs in. It was considered as important for classifying workflow from

Group	Size	Most frequent tags	Description
1	30%	localworker, example, mygrid	Workflows using local scripts.
2	29%	bioinformatics, sequence, protein, BLAST, alignment, similarity, structure, search, retrieval	Sequence similarity search using the BLAST algorithm
3	24%	benchmarks	Benchmarks WFs.
4	6.7%	AIDA , BioAID, text mining, bioassist, demo, biorange	Text mining on biomedical texts using the AIDA toolbox and BioAID web services
5	6.3%	Pathway, microarray, kegg	Molecular pathway analysis using the Kyoto Encyclopedia of Genes and Genomes (KEGG)

Table 1: Characterization of workflow groups derived by clustering.

Method	Accuracy	Precision	Recall
Graph Kernels	81.2 \pm 10.0	71.9 \pm 22.0	38.3 \pm 21.1
MCS	73.9 \pm 9.3	73.5 \pm 24.7	4.8 \pm 27.4
Bags of services	73.5 \pm 10.3	15.5 \pm 20.6	3.4 \pm 30.1
Text Mining	77.8 \pm 8.31	67.2 \pm 21.5	31.2 \pm 25.8

Table 2: Performance of workflow discovery.

group 2, which consists of workflows using the BLAST algorithm to calculate sequences similarity. The presented pattern is a sequence of components that are needed to run a BLAST service.

This example shows that graph kernels can be used to extract useful patterns. These patterns then can be recommended to the user during creation of a new workflow.

6 Conclusions

Workflow enacting systems have become a popular tool for the easy orchestration of complex data processing tasks. However, the design and management of workflows are a complex tasks. Machine learning techniques have the potential to significantly simplify this work for the user.

In this paper, we have discussed the usage of graph kernels for the analysis of workflow data. We argue that graph kernels are a good tool for the analysis of workflow data in the practical important situation where no meta data is available. This is due to the fact that the graph kernel approach allows to take decompositions of the workflow into its important substructures into account while allowing a flexible integration of these information contained into these substructures into several learning algorithms.

We have evaluated the use of graph kernels in the fields of workflow similarity prediction, metadata extraction, and pattern extraction. A comparison of graph-based workflow analysis with metadata-based workflow analysis in the field of workflow quality modeling showed that metadata-based approaches outperform graph-based approaches in this application. However, it is important to recognize that the goal of the graph-based approach is not to replace the metadata-based approaches, but to serve as an extension when no or few metadata is available.

References

[Corrales *et al.*, 2006] Juan Carlos Corrales, Daniela Grigori, and Mokrane Bouzeghoub. Bpel processes matchmaking for service discovery. In *In Proc. CoopIS 2006, Lecture Notes in Computer Science 4275*, pages 237–254. Springer, 2006.

[Fraser, 2005] M. Fraser. *Virtual Research Environments: Overview and Activity*. Ariadne, 2005.

[Gaertner *et al.*, 2003] Thomas Gaertner, Peter Flach, and Stefan Wrobel. On graph kernels: Hardness results and efficient alternatives. In *Proceedings of the 16th Annual Conference on Computational Learning Theory and 7th Kernel Workshop*, pages 129–143. Springer-Verlag, August 2003.

[Goderis *et al.*, 2005] Antoon Goderis, Ulrike Sattler, Phillip Lord, and Carole Goble. Seven bottlenecks to workflow reuse and repurposing. *The Semantic Web ISWC 2005*, pages 323–337, 2005.

[Goderis *et al.*, 2006] Antoon Goderis, Peter Li, and Carole Goble. Workflow discovery: the problem, a case study from e-science and a graph-based solution. In *ICWS '06: Proceedings of the IEEE International Conference on Web Services*, pages 312–319. IEEE Computer Society, 2006.

[Goderis *et al.*, 2009] Antoon Goderis, Paul Fisher, Andrew Gibson, Franck Tanoh, Katy Wolstencroft, David De Roure, and Carole Goble. Benchmarking workflow discovery: a case study from bioinformatics. *Concurr. Comput. : Pract. Exper.*, (16):2052–2069, 2009.

[Goderis, 2008] Antoon Goderis. *Workflow re-use and discovery in bioinformatics*. PhD thesis, School of Computer Science, The University of Manchester, 2008.

[Horváth *et al.*, 2004] Tamás Horváth, Thomas Gärtner, and Stefan Wrobel. Cyclic pattern kernels for predictive graph mining. In *KDD '04: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 158–167, New York, NY, USA, 2004. ACM.

[Kashima and Koyanagi, 2002] Hisashi Kashima and Teruo Koyanagi. Kernels for semi-structured data. In *ICML '02: Proceedings of the Nineteenth International Conference on Machine Learning*, pages 291–298, San Francisco, CA, USA, 2002. Morgan Kaufmann Publishers Inc.

- [Oinn *et al.*, 2004] T Oinn, M.J. Addis, J. Ferris, D.J. Marvin, M. Senger, T. Carver, M. Greenwood, K Glover, M. R. Pocock, A. Wipat, and P. Li. Taverna: a tool for the composition and enactment of bioinformatics workflows. *Bioinformatics*, 20(17):3045–3054, June 2004.
- [Roure *et al.*, 2008] David De Roure, Carole Goble, Jiten Bhagat, Don Cruickshank, Antoon Goderis, Danius Michaelides, and David Newman. myexperiment: Defining the social virtual research environment. In *4th IEEE International Conference on e-Science*, pages 182–189. IEEE Press, December 2008.
- [Roure, 2009] Robert Stevens David De Roure. The design and realisation of the myexperiment virtual research environment for social sharing of workflows. 2009.
- [Taylor *et al.*, 2006] Ian J. Taylor, Ewa Deelman, Dennis B. Gannon, and Matthew Shields. *Workflows for e-Science: Scientific Workflows for Grids*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.
- [Thom *et al.*, 2007] Lucinea Thom, Cirano Iochpe, and Manfred Reichert. Workflow patterns for business process modeling. In *Proc. of the CAiSE'06 Workshops - 8th Int'l Workshop on Business Process Modeling, Development, and Support (BPMDS'07)*, page Vol. 1. Trondheim, Norway, 2007.
- [Van Der Aalst *et al.*, 2003] W. M. P. Van Der Aalst, A. H. M. Ter Hofstede, B. Kiepuszewski, and A. P. Barros. Workflow patterns. *Distrib. Parallel Databases*, 14(1):5–51, 2003.
- [White, March 2004] Stephen A. White. Business process trends. In *Business Process Trends*, March, 2004.