Semantic-Driven Design and Management of KDD Processes

Emanuele Storti
Dipartimento di Ingegneria Informatica, Gestionale e dell'Automazione
Universita' Politecnica delle Marche
via Brecce Bianche, 60131 Ancona, Italy
storti@diiga.univpm.it
Advisor: Claudia Diamantini

DOCTORAL DISSERTATION EXTENDED ABSTRACT

KEYWORDS: Collaborative Knowledge Discovery in Databases, Semantic Technologies for KDD, Knowledge Discovery Support System, e-Science

1. INTRODUCTION

The rapid growth of databases in last years asks organizations to deal with issues related to the management of large amounts of data, which represent a valuable resource for decision making processes. Although technologies for data management/storage are widely available, much effort is still needed to provide users with systems for effectively analyzing and understanding data. We use the term Knowledge Discovery in Databases (KDD) to refer to the non-trivial process of extracting interesting, valid and useful patterns from data. As a process, it often involves several steps, which may include: selection of a subset of data from the whole dataset, data cleaning and transformation, feature extraction, choice of the appropriate Data Mining technique for extracting patterns, its configuration and execution, evaluation and interpretation of results and deployment of new knowledge to users. Especially for non-experts, definition and management of a KDD process are themselves demanding activities, because they require user to know how to choose the proper tools among the plethora of available ones, how to setup them, how to interpret their output. In order to manage a KDD process, it is usually needed a team of different experts, each of which is able to configure only a part of the whole process. Such a team either can belong to the same organization or can be a geographically distributed virtual team of experts. Hence, integration of distributed users and tools, along with heterogeneity of these latter are issues to take into account in order to define effective solutions for the problem at hand. Real scenarios for KDD may include not only network organizations, for which distributed cooperative KDD projects may represent a significant added value, but also the support to e-Science processes (e.g. particle physics, earth sciences, and bioinformatics). In such a highly distributed environment, in fact, scientists need technologies for a collaborative analysis of data produced by scientific experimentations.

So far, to the best of our knowledge, no solution can be found in commercial software or in the Literature to properly face all these challenging issues. About the former, commercial KDD platforms typically provide tools for building a KDD process out of a set of algorithms, but users have to manually compose the workflow. Even if some software are open-source, tools can be executed only locally with little or no actual chance for extension/integration with complex support functionalities like those previously mentioned. Several works, available in the Literature, faced these topics, especially in last years, although most of them are mostly concerned with large-scale and high-performance issues, focusing mainly on Data Mining phase without considering the KDD process as a whole. Few authors specifically dealt with semiautomatic composition of KDD processes. Among them, [1] and [2] defined procedures to compose tools for forming KDD processes by exploring the space of possible solutions. In order to effectively support composition, experts' knowledge about algorithms must be coded in a computer-readable format: storing semantic information about algorithms is a widely accepted solution, even though at present a shared ontology for KDD is still missing; some of the approaches propose top-level Data Mining ontologies, which are too

978-1-4244-6622-1/10/$26.00 ©2010 IEEE

abstract to support any specific application, while others, (e.g.[1],[2]) define a KDD ontology focused on concrete KDD tools, although limiting the possibility for finding more general relationships among algorithms by means of inferential reasoning.

2. METHOD USED AND PROPOSED SOLUTION

To provide an active support in such complex activities, our research focuses on solutions for the management of a KDD process in a cooperative distributed environment. Such a scenario can be built on an open and modular architecture, based on an explicit representation of knowledge and information, and on a set of support services, which are devoted both to facilitate the exploitation of KDD tools and to reduce the knowledge gap among the different actors in a KDD project. Therefore, we are currently developing a service-oriented architecture where KDD tools are available as services, remotely accessible through standard protocols, which ensure communication among the participants apart from the interfaces they use. In order to get the highest transparency and reusability, we describe KDD tool's information in three logical layers, with different abstraction degrees: KDD algorithm level is the most abstract one, whereas different implementations of the same algorithm are described at the KDD tool level. Finally, different instances of the same tool can be made available on the net as KDD services by several providers. As different tools are likely to have different interfaces/behaviors, our system is based on descriptive languages for representing syntax and semantics of tools in a consistent manner. KDD tools and services are described through XML-based descriptors, which contain not only syntactic details about interfaces, but also tool's semantics: in such a way, we can explicitly map each service to a tool, and this last to an algorithm. Definition of algorithms are represented into a formal KDD ontology [3], together with data structures, methods and tasks, arranged into taxonomies and linked through relations. Such an open and loosely coupled architecture allows to design user-centric client applications, exploiting only those services that are needed. In our architecture, a set of basic services implement tools for all the phases in a KDD process (preprocessing, modeling, postprocessing). In addition, more complex services are needed for supporting users in specific activities, e.g. publishing new KDD services into a repository and browsing the repository for retrieving services which satisfy specific syntactic/semantic requirements. However, providing support for KDD process management does not only mean accessing to each tool separately, but also defining higher-level functionalities, especially useful for non-expert users. Among these, in particular we are focusing on technologies for semi-automatic goal-oriented composition of KDD processes, and on their execution as distributed workflows. Such a procedure generates abstract KDD processes, i.e. workflows of algorithms that allow achieving the goal requested by the user (classification, clustering, etc...). Therefore, in order to execute a process, each algorithm must be replaced with a concrete service implementing it.

The systematic use of semantic information, a loosely coupled and layered architecture are the main aspects that distinguish our approach from the others. While most solutions focus only on Data Mining step or on local KDD support systems, our proposal is more general and suited to an open environment, in which tools and users are both distributed. The separation of information in 3 layers allows us to reuse components and to provide support at different abstraction levels, for users with different expertise or different needs. Moreover, the usage of semantically enriched descriptions of tools and algorithms, where any element is linked to a formal defined concept into the ontology, leads to multi-fold benefits: since it embodies experts' knowledge into the system, it allows to support complex functionalities by automating several steps; moreover, by exploiting inferential reasoning on the ontology we can discover new valid and interesting processes; finally, it helps to improve ease-of-use by providing information about the meaning of any element. Besides the aforementioned considerations, our approach in process composition, working at different abstraction layers, allows to produce abstract KDD processes, which are semantically valid, general, and reusable, since each instance of algorithm can be replaced with a service implementing it. Furthermore, such abstract processes can be themselves considered useful and unknown knowledge, because they provide valid insight about which chains of abstract steps ought to be carried out to achieve the specified goal.

3. CURRENT STATUS OF THE RESEARCH

So far, many elements of the platform have been developed and are available at the project website [4]. Together with basic KDD services, which perform any kind of KDD task, some support services are available as well. Among them: a *service registry*, a *semantic broker* (for managing publication of new services into a registry, discovery of new services by semantic queries) and an *ontology browser.* About process composition, in [5] we first introduced a goal-oriented procedure aimed at

automatically composing abstract KDD processes. Then, we focused on the formal definition of exact/approximate matches, on the specification of the composition procedure and on metrics for evaluating the cost of a match [6]. In more details, the design of an abstract process can be ultimately reduced to the sub-problem of algorithm matching, which is a basic step for automatically composing KDD processes. Given two algorithms, the matching is based on the comparison between the output of the first and the input of the second, in order to determine whether they can be executed in sequence, i.e. whether their interfaces are compatible or not. Such a compatibility is evaluated w.r.t. the relations that may exist among the data within the KDD ontology. In such a way, it is possible to perform reasoning on the ontology for deducing hidden and non-explicit relations among algorithms. In particular, by browsing semantic relations it is possible not only to find suitable algorithms on the basis of an *exact match* between input and output, but also to define *approximate matches*. Such matches are based on subsumption relations and, unlike many previous works, also parthood relations among a compound data structure and its subcomponents. A score can be assigned to each kind of match according to a semantic distance function. Composition proceeds backwards from the goal to the datasets, applying iteratively the matching functions, and stops as soon as some constraints (about time, cost, process dimension) are violated. The outcomes of this procedure are workflows of algorithms allowing to achieve the goal requested by the user. The generated processes are ranked according to both user-defined and built-in criteria, allowing users to choose the most suitable ones, with respect to their requests. A working prototype has been realized for implementing the composition procedure and for evaluation and assessment of results.

4. PLANS TO COMPLETE THE RESEARCH

At present we are involved in further extensions of process composition, aimed both at improving even more the reliability of the procedure's output and at translating abstract processes into concrete workflows. While the former activity requires to take into account further information during the composition (e.g. dataset's statistical distribution, for suggesting the algorithms that are likely to produce the best models), the latter requires to replace each abstract algorithm in a process with a corresponding KDD service, that can be found by exploiting our layered architecture through invoking some support services (e.g. repository, broker). For achieving such a goal, we need to deal with syntactic differences between the interfaces of abstract algorithms and actual services. A possible way is to represent both the abstract process and the interface/behavior of a concrete KDD service through a formal language, which can be used to formally check whether the service can replace a specific algorithm in the process. Furthermore, since we refer to a collaborative environment, KDD processes are likely to be executed in a distributed fashion. Each service can be actually available at a different site, as well as each user that takes part in the KDD project. For such a reason, even though our open platform helps in managing integration/heterogeneity of different services, we still need to face the coordination issues that are intrinsically related to such a distributed context. Through the coordination language Reo [7], we are currently planning to design workflows for distributed execution of complex KDD processes, in which both iteration/interaction and automation are taken into account. Moreover, such workflows could be formally verified in order to check safety/liveness properties, like lack of deadlocks or compliance with project specifications.

REFERENCES

[1] A. Bernstein, F. Provost and S. Hill, "Toward Intelligent Assistance for a Data Mining Process: An Ontology Based Approach for Cost-Sensitive Classification," *IEEE Transactions on Knowledge and Data Engineering*, Vol. 17, No. 4, 2005, pp. 503–518

[2] M. Zakova, P. Kremen, F. Zelezny and N. Lavrac, "Using Ontological Reasoning and Planning for Data Mining Workflow Composition", *ECML/PKDD 2008 workshop on Third Generation Data Mining: Towards Service-oriented Knowledge Discovery*, Antwerp, Belgium, 2008.

[3] C. Diamantini, D. Potena and E. Storti, "KDDONTO: an Ontology for Discovery and Composition of KDD Algorithms", *SOKD: ECML/PKDD 2009 Workshop on Third Generation Data Mining: Towards Service-oriented Knowledge Discovery*, Bled, Slovenia, Sep 7-11 2009, pp. 13-24.

[4] KDDVM project website: http://boole.diiga.univpm.it

[5] C. Diamantini, D. Potena and E. Storti, "Ontology-driven KDD Process Composition", *Advances in Data Analysis VIII*, LNCS, Springer, Vol. 5772, 2009, pp. 285-296.

[6] C. Diamantini, D. Potena, and E. Storti, "Automatic Definition of KDD Prototype Processes by Composition". A. D'Atri et al., eds., MANAGEMENT OF INTERCONNECTED WORLD, Springer, 2010

[7] F. Arbab, "Reo: a channel-based coordination model for component composition", *Mathematical Structures in Computer Science*, Cambridge University Press, Vol. 14, No. 3, 2004, pp. 329-366.