International Journal of Computer Science and Applications,
Vol. 5, No. 3, pp 36 - 45

TOWARDS EFFECTIVE SOLUTIONS FOR PATTERN MANAGEMENT

BARBARA CATANIA

Dipartimento di Informatica e Scienze dell'Informazione, Via Dodecaneso 35, 16146 Genova, Italy
catania@disi.unige.it
http://www.disi.unige.it/person/CataniaB

Patterns can be defined as knowledge artifacts, providing a compact and semantically rich representation of a huge quantity of heterogeneous raw data. Due to the specific characteristics of patterns, ad hoc systems are required for pattern management, in order to model, store, retrieve, and manipulate patterns in an efficient and effective way. Pattern management historically concerned the definition of specific pattern extraction methods. Nowdays, the diffusion of knowledge intensive applications has required the ability to model and manage, possibly heterogeneous, patterns in an integrated and more specialized way. In this paper, after introducing the pattern management problem, new issues concerning pattern management are pointed out, discussing possible solutions. The identified challenges concern modeling, querying, manipulation, and architectural issues that, from the author's point of view, should be taken into account in order to design next generation pattern-management systems.

Keywords: pattern, pattern management, data mining.

1. Introduction

In many different modern contexts, a huge quantity of raw data is often collected, possibly from different devices. An usual approach to analyze such vast volume of data, which do not constitute information per se and do not allow any kind of easy management, is to generate some compact knowledge artifacts (i.e., clusters, association rules, frequent itemsets, etc.) through data mining methods. We refer to those knowledge artifacts with the term *pattern*. Patterns can therefore be defined as knowledge units that effectively describe entire subsets of data (in this sense, they are *compact*). The quality of the representation achieved by a pattern can be evaluated by using some statistical measures describing relevant data properties (in this sense, they are *rich in semantics*). With the term *pattern management* we mean the overall set of functionalities required to generate, store, retrieve, analyze, and modify patterns.

Pattern management is nowdays an important issue in many different contexts and domains. The most important context in which pattern management is required is certainly data mining. Additional domains of interests include: information retrieval, where keyword frequencies and frequent sets of words extracted from documents are examples of patterns; signal processing, for example for the analysis of audio data; image processing, where recurrent figures in shapes can be interpreted as specific types of patterns; machine learning, where predictions and forecasting activities are based on classifiers, which can be interpreted as specific types of patterns.

Of course, patterns can be interpreted as a kind of metadata. Metadata represent data over data and, since patterns represent knowledge over data, there is a strong relationship between metadata management and pattern management. However, metadata are usually provided for maintaining process information, as in data warehousing, or for representing knowledge in order to guarantee interoperability, as in the Semantic Web and intelligent agent systems. On the other hand, patterns are usually used for representing the semantics of huge quantity of data for the sake of decision making. Additionally, specific pattern characteristics, such as quantification of importance through quality measures, are not taken into account in representing and managing generic metadata.

Historically, much more attention has been posed on extraction and representation of homogeneous (i.e., single type) patterns than on pattern management as a whole. As a result, several techniques have been provided for efficiently generate patterns of specific types [Hand and Kamber, (2001)] and several standards have been proposed for pattern representation (e.g., PMML [PMML, (2007)] and JDM [JDM, (2008)]).

With the diffusion of knowledge intensive applications and new distributed architectures, single type patterns cannot be considered as a significant assumption any more. Indeed, patterns can be quite different and require, at the same time, an integrated management. As an example, in the context of the market-basket analysis association rules over sale transactions, are often generated. Moreover, in order to perform a market segmentation, the user may also be interested in identifying clusters of customers, based on their buying preferences, or clusters of products, based on customer buying habits. As another example, in a Web context, in order to well understand e-commerce buying habits of Web users, different patterns can be combined, for example: (i) navigational patterns (identified by clickstream analysis) describing user surfing and browsing behavior; (ii) demographic and geographical clusters, obtained with market segmentation analysis based on personal data and geographical features; (iii) frequencies of the searching keywords specified by the user when using a searching engine.

Besides pattern heterogeneity, sophisticated technologies are required to support all required pattern management functionalities. Indeed, patterns share some characteristics that make traditional DBMSs unable to represent and manage them. For example, patterns can be generated from raw data by using some data mining tools (a-posteriori patterns) but also known by the users and used for example to check how well some data source is represented by them (a-priori patterns). Since source data change with high frequency, another important issue consists in determining whether existing patterns, after a certain time, still represent the data source from which they have been generated, possibly being able to change pattern information when the quality of the representation changes. Finally, all kinds of patterns should be manipulated (e.g. extracted, synchronized, deleted) and queried through dedicated languages.

In [Rizzi *et al.*, (2003)], the term *Pattern Based Management System (PBMS)* has been introduced as a system for handling (storing/processing/retrieving) patterns defined over raw data in order to efficiently support pattern matching and to exploit pattern-related operations generating intensional information. A PBMS, therefore, is not a simple knowledge repository; it is an engine supporting pattern storage (according to a

chosen logical model) and processing (involving also complex activities requiring computational efforts). Thus, the design of a PBMS relies on solutions developed in several disciplines, such as: data mining and knowledge discovery, for a-posteriori pattern extraction; database management systems, for pattern storage and retrieval; data warehousing, for providing raw data sets; artificial intelligence and machine learning, for pattern extraction and reasoning; metadata management.

Many efforts have been devoted towards the development of PBMSs and in general of pattern management applications. Scientific community efforts are mainly devoted to the formalization of the overall principles under which a PBMS can be developed, providing the background for the design of back-end technologies to be used by pattern-based applications. The 3W Model [Johnson, *et al.*, (2000)] and the PSYCHO system [Catania, *et al.*, (2005)] are examples of such approaches, in which raw data are stored and managed in a traditional way by using, for example, a DBMS whereas patterns are stored and managed by a dedicated PBMS. On the other hand, under the inductive databases approach, mainly investigated in the context of the CINQ project [CINQ, (2001)], raw data and patterns are stored by using the same data model and managed in the same way by the same system. Industrial proposals mainly deal with standard representation purposes for patterns resulting from data mining, in order to support their exchange between different platforms. Examples of such approaches are the Predictive Model Markup Language [PMML, (2007)], an XML-based language for common data mining representation, and the Java Data Mining API [JDM, (2003)], a Java API for pattern management. Finally, in the commercial world, the most important DBMSs address the pattern management problem by offering features for representing and managing typical data mining patterns [DB2, (2008)], [ODM, (2008)], [MS SQL, (2008)]. Usually, they do not provide a comprehensive framework for pattern management; rather they support Business Intelligence applications by providing an application layer offering data mining features, in order to extract knowledge from data, and by integrating mining results with OLAP instruments in order to support advanced pattern analysis. For this reason, in general, they do not provide a dedicated logical model for pattern representation and querying, since these aspects are demanded to the applications using the mined results.

Even if several approaches have been provided in order to model and query (heterogeneous) patterns in an homogeneous way, we are still far from the design of PBMSs, addressing all pattern management issues relevant for new applications in an effective and efficient way. But what are such issues? Which are the applications that may get benefits from such a technology? How far are we from consolidated PBMS technologies? In this paper, we try to answer these questions by presenting research challenges and potential solutions for pattern management, grouping them in three distinct categories: modeling, querying and manipulation, and architectural issues. Some final remarks concerning the development of next generation PBMSs concludes the paper.

2. Modeling Issues in Pattern Management

The logical model adopted for representing patterns is a formalism by which patterns are described and manipulated inside a PBMS or inside an application layer. We claim that, in order to be able to design next generation PBMSs, the following challenges concerning pattern modeling should be taken into account.

Challenge 1. Design of flexible pattern models. With flexible pattern modeling we mean the ability to model both heterogeneous and user-defined patterns, in order to make the system completely extensible. Both characteristics are very relevant in modeling advanced and data intensive applications. The model should also be able to represent hierarchically organized patterns (as a simple example, clusters of association rules), in order to increase expressivity, reusability, and modularity.

At the state of the art, existing standards or commercial PBMSs are far from allowing the representation of user-defined or heterogeneous patterns. The main problem here is how a pattern can be modeled in a general way. Even if some theoretical proposals already exist [Johnson, *et al.*, (2000)], [Catania, *et al.*, (2004)], standards have not taken them into account yet. A different situation arises for pattern hierarchies, that, besides theoretical proposals, can also be modeled, to some extent, in PMML [PMML, (2007)].

Challenge 2. Representation of relationships between raw data and patterns. By definition, each pattern represents a possibly huge set of source data. There exists, therefore, a relationship between each pattern and the data source it represents. In order to enhance pattern semantics and provide additional significant information for pattern retrieval and query processing, it may be useful to store some information concerning such relation. In case of patterns generated by applying a mining algorithm, the data source corresponds to the data set from which patterns have been mined. Besides the source data set, it may also be useful to know the subset of the source data set represented by the pattern.

With the exception of some theoretical proposals [Johnson, *et al.*, (2000)], [Catania, *et al.*, (2005)], usually relationships between data and patterns are not represented by standard representation models and commercial PBMSs. On the other hand, most existing approaches allow the description of the source data set, which is used for pattern extraction. We claim that the representation of the specific subset of data represented by a pattern could make query processing much more flexible. Indeed, such data set can be represented either in a precise way, by listing its components, or in an approximated way, by providing a formula satisfied by the items of the source data set from which the pattern has been generated. In this second case, by exploiting such intensional representation, the PBMS could be able to answer queries concerning both patterns and data (the so called *cross-over* queries) without accessing raw data, with obvious improvements in terms of performance. Of course, the issue here concerns the definition of a representation format guaranteeing good expressive power and tractable complexity. The usage of linear constraint polynomial could be a choice to be exploited.

Challenge 3. Quality measures support. It is important to be able to quantify how well a pattern represents a raw data set, by associating each pattern with some quantitative measures, depending on the specific application purposes. For instance, considering association rules, confidence and support are the most widely used measures, but there exist other quality measures such as coverage, leverage, and lift [Han and Kamber, (2001)], which could be used in specific applications.

The capability of choosing quality measures to be associated with patterns of a certain type in the context of a given application or application domain is not present in most existing PBMSs, which usually offer a fixed set of measures for each pattern type. If different measures are provided, ontologies should be probably used to relate their semantics and to be able to consistently use them inside queries (see Challenge 6).

Challenge 4. Pattern validity support. Since data change with a frequency rate higher than the pattern change rate, it is important to establish whether existing patterns, after a certain time, still represent the data source from which they have been generated or, more in general, which they represent. To this end, some notion of pattern validity should be exploited. Different notions of pattern validity could be probably defined, either based on application constraints or on raw data semantics.

As far as we know, some preliminary formalization of the validity issue has only been provided in the context of some theoretical proposals [Catania, *et al.* (2004)].

3. Querying and Manipulation Issues in Pattern Management

Similarly to a DBMS, a PBMS must support a language for querying patterns and a language for manipulating them. Such languages should be designed by taking pattern peculiarities into account. We identify the following challenges in designing pattern languages.

Challenge 5. Definition of powerful query languages. Pattern queries are usually defined over pattern collections. Since patterns represent knowledge, besides simple operations such as selection, advanced operations over patterns must be supported. As an example, we claim that pattern combination should be defined as an advanced form of reasoning over heterogeneous patterns. Combination can be seen as a sort of “join” between patterns. For example, transitivity between association rules can be seen as a kind of pattern join. Additionally, by combining patterns of different types we may get deeper insights of the underlying raw data sets. For example, we may be interested in determining all clusters of products with similar composition which are often sold together, in the context of the same transaction (thus, which appear in the same frequent itemset). A system managing heterogeneous patterns should also provide operations not only for querying patterns in isolation, but also for querying patterns and data together, by applying a cross-over processing between them.

Pattern query languages supporting all or some of the previous features have been defined only in the context of theoretical proposals. However, also in those cases, the provided operators are quite simple. For example, based on the limitations in pattern modeling, often the provided operators deal with specific types of patterns and cross-over queries are usually implemented by considering the overall data source from which patterns have been generated. This requires the system to access both a pattern set and a, usually quite huge, data set for answering cross-over queries. An intensional representation, at the model level, of the relationship existing between pattern and data, as pointed out in Challenge 1, may reduce access costs, since in this case cross-over queries could be executed, even if in an approximated way, completely by the PBMS. In general, the definition of a formal semantics for heterogeneous pattern combination and for approximated cross-over queries is a very hot issue for developing powerful pattern query languages, suitable for advanced applications.

We remark that standards providing pattern access operations, like ISO SQL/MM [SQL/MM, (2001)] and Java Data Mining [JDM, (2008)], assume that raw data and patterns are stored together, under the same system and using the same logical model; therefore, manipulation and querying are possible by using typical languages used for accessing data.

Challenge 6. Ability to reason about patterns. An overall approach for reasoning about (heterogeneous) patterns needs sophisticated techniques, relying on a description of the semantics of the various pattern characteristics. As an example, consider measures. In general, various approaches exist for measure computation (general probabilities, Dempster-Schafer, Bayesian Networks - see for example [Silberschatz and Tuzhillin, (1996)]). It is not clear how patterns, possibly having the same type but characterized by different measures, can be compared and managed together. Probably, measure ontologies can be used to support such kind of quantitative pattern reasoning. Additionally, similarity functions to compare patterns, possibly of different types, are needed, to provide advanced forms of reasoning, especially in emerging distributed architectures, such as the Semantic Web, P2P architectures, or the Semantic Grid, where no complete information may be available about patterns to be retrieved. Techniques for similarity-based pattern retrieval may allow users to analyze the differences existing among some data sets just by analyzing patterns associated with them.

Pattern reasoning is supported only in few theoretical proposals, in the form of pattern combination (see, for instance, [Johnson, *et al.*, (2000)]). An overall proposal for pattern reasoning is still missing. Similarly, only few general approaches for pattern similarity have been provided that can be homogeneously applied to different types of patterns (see for example [Bartolini, *et al.* (2004)] and [Ganti, *et al.*, (1999)]). In general, this constitutes a very hot topic for researchers and there is no general agreement on a general enough pattern similarity function. Of course, the definition of a formal and flexible model for pattern representation is a prerequisite for the definition of such similarity functions.

Challenge 7. Synchronization over source data. Another important issue to be taken into account concerns the ability to deal with unexpected changes in raw data, which may require changing the patterns representing them. Indeed, since source data change with high frequency, it becomes fundamental to determine whether existing patterns, after a certain time, still represent the data source from which they have been generated (i.e., whether they are still semantically valid), possibly being able to change pattern information, in particular measures, when the quality of the representation changes (i.e., synchronizing raw data and patterns). Synchronization can be performed against the source data set the pattern is associated with or with respect to a different data set, totally recomputing pattern measures.

This issue constitutes a key feature for any pattern management system, and it is an interesting topic for researches in the areas of machine learning and data mining. It is often referred in literature as the *virtual concept drift* problem [Tsymbal, (2004)]. In the data mining context, incremental mining algorithms have been proposed to update data mining patterns as soon as the raw data sets change. The integration of such algorithms in specific pattern manipulation languages, however, has not been extensively considered.

4. Architectural Issues in Pattern Management

Similarly to DBMSs, in order to make PBMSs a practical technology, several additional aspects have to be taken into account, in order to be able to manipulate patterns in an efficient and reliable way. In the following, we summarize the main challenges related to this topic.

Challenge 8. Definition of specific design techniques for pattern storage. Since patterns are assumed to be stored in a repository, specific techniques for physical storage must be developed.

Unfortunately, it is not clear up to now what should be a reasonable physical layer for patterns. Most commercial DBMSs store patterns as BLOB that are then manipulated using specific methods. However in order to provide a more efficient access, specific physical representations, clustering, partitioning, caching, and indexing techniques should be developed. Such techniques may consider different pattern properties, such as their type, in the simplest case, or the characteristics of the raw data set they represent, or even specific quality measures.

Challenge 9. Pattern query optimization. Query optimization for pattern queries has been only marginally investigated. Some preliminary work, concerning query rewriting, has been proposed in the context of the 3W framework [Johnson, *et al.*, (2000)]. An overall query optimization approach, taking into account choices concerning physical layer characteristics, has not been defined yet. Assuming to store patterns separately with respect to raw data, the main issue is how to deal with data and patterns computations in an efficient way. An important topic here is how it is possible to use patterns to reduce

data access in data and cross-over queries and how data and pattern query processors can be combined together.

Some work has been done in the context of inductive databases, where approaches to optimize pattern extraction, based on constraints on pattern properties, [Ng, *et al.*, (1998)], or to refine the set of generated patterns [Baralis and Psaila, (1999)], have been proposed. Techniques for reducing the size of the generated pattern sets, by representing them using condensed representations, have also been proposed for itemsets and association rules [CINQ, (2001)]. However, no overall pattern query optimization approach, taking into account PBMS statistics and logical properties of the considered query language, has been defined yet.

Challenge 10. Access control models for patterns. Patterns represent high-sensitive information. Their access has therefore to be adequately controlled. The problem is quite similar to access control in presence of inference [Farkas and Jajodia, (2002)]. In general, assuming a user has access to some non-sensitive data, the inference problem arises when, through inference, sensitive data can be discovered from non-sensitive one. In terms of patterns, this means that users may have the right to access some patterns, for example some association rules, and starting from them they may infer additional knowledge over data, over which they may not have the access right.

Techniques already proposed in the inference context should be adapted and extended to cope with the more general pattern management framework. Some of these approaches rely on pre-processing techniques, checking through mining techniques whether it is possible to infer sensitive data; some others can be applied at run-time, i.e. during the knowledge discovery phase,, releasing patterns only when they do not represent sensitive information; finally modifications over original data, such as perturbation and sample size restrictions, do not disturbing data mining results, can be also applied in order to encript the original data and to prevent unauthorized user data access.

Challenge 11. Integration of pattern bases. Recently, pattern management issues are becoming much more important not only in centralized architecture but also in distributed ones. Indeed, in recent years, due to the World Wide Web pervasiveness and to the diffuse demand for large-scale, intensive data applications, the need of being able to retrieve information and knowledge which may be stored on different, autonomous, and heterogeneous data sources has become much more pressing and has sped up the requirement for knowledge discovery and management systems able to handle and analyze multi-site and multi-owner knowledge repositories, which must be integrated.

Classical solutions to data and knowledge integration rely on semantic integration approaches, which define a mapping between a global schema and each data source schema, before any service can be provided. Often, in real cases, the data integration approach is not usable, since writing (and maintaining updated) the mapping between each local source schema and the global one may be costly or even unfeasible. As an alternative approach, the notion of *dataspace* has been recently proposed as a new abstraction for modeling data co-existence, providing functionalities over all data sources

regardless of their integration level [Franklin, *et al.*, (2005)], [Halevy, *et al.*, (2006)]. Queries over a dataspace range from structure aware queries (whenever they are supported by dataspace participants) to keywords queries (in the most general case, when participants offer highly heterogeneous query capabilities).

We claim that dataspace technology could be relevant also for pattern management when: (i) different subjects acting in the same area use different platforms to generate and analyze different patterns of knowledge in order to reach their specific business objectives; (ii) the ability to globally access and relate such local knowledge may provide deeper insight into the considered business area.

Up to now, since dataspace theory is still in its infancy, no solutions have been provided for developing knowledge spaces. From our point of view, the most important issue concerns the identification of query and analysis operations, suitable for the pattern context, to be specified and implemented independently from the specific PBMS or, more generally, knowledge source functionalities and the design of ad hoc pattern models to be used for the efficient implementation of such operations.

5. Concluding Remarks

Pattern management is a complex activity, concerning any functionality required to generate, store, retrieve, analyze, and modify compact and rich in semantics knowledge artifacts. Even if several proposals exist and have already been consolidated, further issues need to be investigated in order to make pattern management a practical technology. In particular, from the author's point of view, several aspects concerning the representation and the integrated management of heterogeneous patterns have not been deeply investigated yet. Limitations of current proposals span from a lack of modeling capabilities (such as no support for user-defined patterns, pattern hierarchies, or validity notions) to a lack of manipulation, analysis, and querying operators (no manipulation of heterogeneous patterns, no support for similarity, pattern combination, restricted support of synchronization in standard/commercial proposals).

More generally, it seems that an overall framework, in terms of the current standards, for pattern management is still missing; indeed, most pattern-related operations are often directly implemented in the application layer, with obvious interoperability and efficiency problems, compared to the efficiency which can be obtained by executing the same operations by a PBMS designed for that purpose.

On the other hand, the diffusion of new application domains that may get benefits from pattern technology is increasing. At the same time, the advent of new distributed architectures and infrastructures calls for solutions enabling an integrated management of heterogeneous patterns. A combined effort of the scientific community with industries is at this time required in order to provide the right extensions to existing solutions and standards, moving towards next-generation pattern management systems.

References

Baralis, E. and Psaila, G. (1999) Incremental Refinement of Mining Queries. In LNCS 1676: Proc. of DaWaK'99, pp. 173–182.

Bartolini, I., *et al.* (2004) A Unified and Flexible Framework for Comparing Simple and Complex Patterns. In LNAI 3202: *Proc. of the 15th ECML/PKDD*, pp. 496–499.

Catania, B., *et al.* (2004). A Framework for Data Mining Pattern Management. In LNAI 3202: *Proc. of ECML-PKDD'04*, pp. 87-98.

Catania, B., *et al.* (2005). PSYCHO: A Prototype System for Pattern Management. In *Proc. of VLDB'05*, pp. 1346-1349.

CINQ (2001). The CINQ project. http://www.cinq-project.org

DB2. (2008) DB2 Intelligent Miner. http://www-306.ibm.com/software/data/iminer/

De Raedt, L. (2002). A Perspective on Inductive Databases. *ACM SIGKDD Explorations Newsletter*, **4**(2), pp. 69–77.

Farkas C. and Jajodia, S. (2002) The Inference Problem: a Survey. *SIGKDD Explor. Newsl.*, **4**(2), pp. 6-11. ACM Press.

Franklin, M., *et al.* (2005). From Databases to Dataspaces: a New Abstraction for Information Management. SIGMOD Rec., **34**(4), pp. 27–33.

Ganti, V., *et al.* (1999). A Framework for Measuring Changes in Data Characteristics. In *Proc.of PODS'99*, pp. 126-137.

Halevy, A., *et al.* (2006) Principles of Dataspace Systems. In *Proc. of PODS '06*, pp. 1–9.

Han, J. and Kamber, M. (2001). *Data mining: Concepts and techniques.* Academic Press.

JDM (2008). Java Data Mining API. http://www.jcp.org/jsr/detail/73.prt

Johnson, S., *et al.* (2000). The 3W Model and Algebra for Unified Data Mining. In *Proc. of VLDB'00* , pp. 21-32.

MS SQL. (2005). Microsoft SQL Server Analysis Server. http://www.microsoft.com/sql/technologies/analysis/default.mspx

Ng, R., *et al.* (1998) Exploratory Mining and Pruning Optimizations of Constrained Associations Rules. In *Proc. of SIGMOD'98*, pp. 13–24.

ODM. (2008). Oracle data mining tools. http://www.oracle.com/technology/products/bi/odm

PMML (2007). Predictive Model Markup Language. http://www.dmg.org/pmml-v3-0.html

Rizzi, S. et al. (2003). Towards a Logical Model for Patterns. In *Proc. of ER'03*, pp. 77-90.

Silberschatz, A. and Tuzhillin (1996). What Makes Patterns Interesting in Knowledge Discovery Systems, IEEE Transactions on Knowledge and Data Engineering, **8**(6), pp. 970–974.

SQL/MM, (2001). ISO/IEC FCD 13249-6, Information technology – Database languages - SQL Multimedia and Application Packages - Part 6: Data Mining

Tsymbal, A. (2004). The Problem of Concept Drift: Definitions and Related Work (Tech. Rep. No. TCD-CS-2004-15). Trinity College Dublin, Department of Computer Science, Ireland. https://www.cs.tcd. ie/publications/tech-reports/reports.04/TCD-CS-2004-15.pdf