A General Architecture for Finding Structural Regularities on the Web

P.A Laur[2] - F. Masseglia[1,2] - P. Poncelet[2] - M. Teisseire[2]

[1]Laboratoire PRiSM, Université de Versailles
45 Avenue des Etats-Unis
78035 Versailles Cedex, France

[2]LIRMM UMR CNRS 5506
161, Rue Ada
34392 Montpellier Cedex 5, France

E-mail: {laur, massegli, poncelet, teisseir}@lirmm.fr

Abstract

With the growing popularity of the World Wide Web, the number of semistructured documents produced in all types of organizations increases at a rapid rate. However the provided information cannot be queried or manipulated in the general way since, although there is some structure in the information, it is too irregular to be modeled using a relational or an object-oriented approach. Nevertheless, some semistructured objects, for the same type of information, have a very similar structure. Analysis of such regularities can provide very useful information for restructuring a Web site, for providing a guideline for building indexes and views and in a general way for applying conventional database technologies which are mainly schema dependent.
In this paper we address the problem of finding such regularities and we propose a general architecture based on a data mining technique.

1 Introduction

With the growing popularity of the World Wide Web (Web), the number of semistructured documents produced increases at a rapid rate. While in classical database applications we first describe the structure of data, i.e. type or schema, and then create instances of that type, in semistructured data, data has no absolute schema and each object contains its own structure [Wor97]. For example, let us consider a person. In a semistructured world, some person-objects might have subobjects with first name and last name. While other person-objects might have a single subobject with a single name as value, or a single name subobject that itself has subobjects first- and last- name. Yet another person-object might have a middle-name subobject, while others have no name at all or have two name subobjects, one of which is a nickname or alias [NUWC97]. On the contrary, some semistructured objects, for the same type of information, have a very similar structure. Analysis of such regularities in such semistructured objects can provide significant and useful information for restructuring a Web site for increased effectiveness [WL98], for improving any meaningfull query for Web documents [KS95], for providing a guideline for building indexes and views [Abi97], etc.
Discovering relationships and global patterns that exist in large files or databases, but are hidden among the vast amounts of data is usually called *data mining*. Motivated by decision support problems, data mining has been extensively addressed in the few past years (e.g. [AIS93, AS94, BMUT97, FPSSU96, SON95, Toi96]). Nevertheless, the proposed approaches mainly concern flat representation of the data and to the best of our knowledge, not much effort has been spent on mining interesting structures from such a data [WL98, WL99]. In fact, this problem is much more complicated than the classical association rule or sequential patterns, since complex structures

in the form of a labeled hirearchical objects partially ordered has to be taken into account.

The groundwork of the approach presented in this paper is the problem of mining structural regularities in a large set of semistructured objects extracted from the Web. More precisely, we want to discover, using data mining techniques, graph structures appearing in some minimum number of objects. We have to pinpoint that this approach is very different from those on extracting the structure of a single individual object since we consider in the following that we are provided with a large collection of graph structures [Wor97].

The rest of this paper is organized as follows. In section 2, the problem is stated and illustrated. The approach is described in section 3. Related work is briefly presented in section 4. Finally section 5 concludes the paper with future directions.

2 Problem statement

In this section we give the formal definition of the structural association mining problem. First we formulate the semistructured data which widely resumes the formal description of the Object Exchange Model (OEM) defined for representing structured data [AQM$^+$97, BDHS96, ABS00]. Second we look at the structural association mining problem in detail. A concrete example is also provided.

2.1 Definitions

The data model that we use is based on the OEM model designed specifically for representing semistructured data. We assume that every object o is a tuple consisting of an *identifier*, a *type* and a *value*. The *identifier* uniquely identifies the object. The type is either *complex* or some identifier denoting an atomic type (like integer, string, gif-image, etc.). When type is complex then the object is called a *complex object* and value is a set (or list) of *identifiers*. Otherwise the object is an *atomic object*, and its value is an atomic value of that type. As we consider set semantics as well as list semantics, we use a circle node to represent an identifier of a set value and a squared node to represented an identifier of a list value.
We can thus view OEM data as a graph where the nodes are the objects and the labels are on the edges. In this paper we assume that there is no cycle in the OEM graph.

We also require that:

(i) *identifier(o)* and *value(o)* denotes the identifier and value of the object o;

(ii) *object(id)* denotes the unique object with an identifier *id*;

(iii) Each atomic object has no outgoing edges.

(iv) If two edges connect the same pair of nodes in the same direction then they must have different label. We thus assume that we are provided with a labeling function $F_E : E \rightarrow L_E$ where L_E is the domain of edge labels.

Figure 1 shows a segment of information about restaurants in the Bay Area [Abi97]. Each circle along with the text inside it represents an object and its identifier. The arrows and their labels represent object references. For instance *value(&44)* = *"El Camino Real"* and *value(&19)="setof"*, standing that *category*, *name* and *address* may be unordered.

A single path in the graph is an alternating sequence of objects and labels $< o_1 l_1 o_2 ... o_{k-1} l_{k-1} o_k >$ beginning and ending with objects, in which each label is incident with the two nodes immediately preceding and following it. Such a path p_k is called a *path expression.* The number of labels from

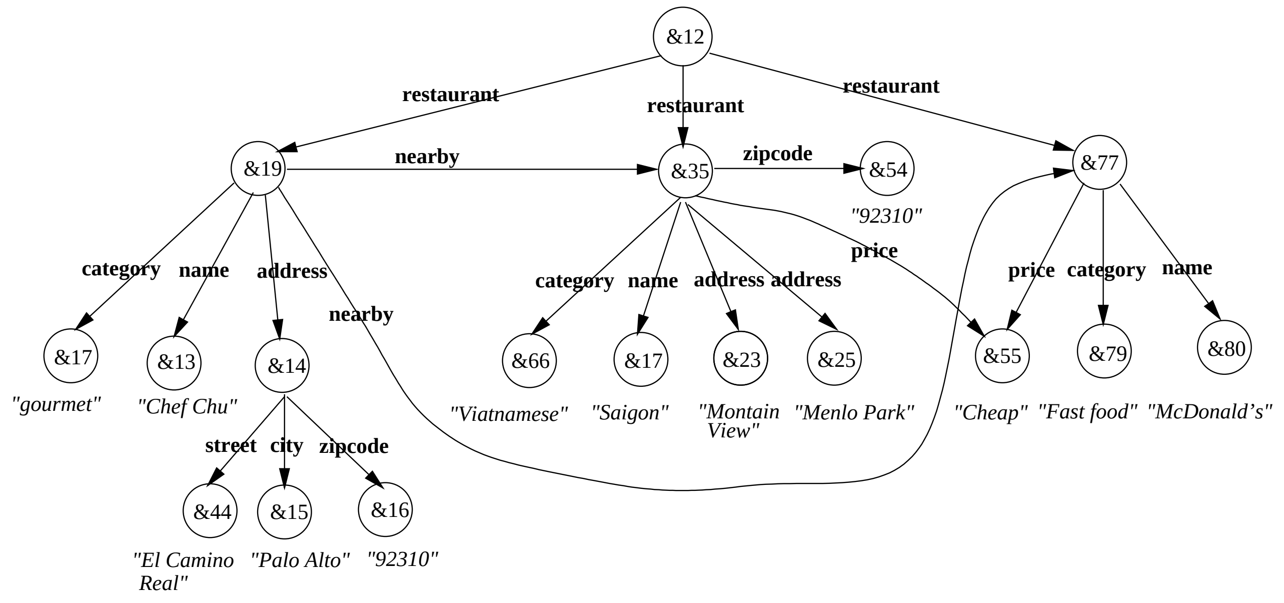

Figure 1: An OEM graph

the source object to the target node in a path, k, is the length of the path. As we consider nested structures, we can consider that the length is similar to the nested level of the structure. Let P_k the set of all paths p where the length of p is k. We now consider multiple path defined as follows: a *multiple path expression* (or *path* for short) is a set of single paths such as the source object is the same in all the single paths[1]. The length of the multiple path is the maximal length of all single paths.
As we are only interested in structural regularity, in the following we do not consider atomic values anymore and we use symbol $\perp$ in order to denote an atomic value in the graph.

Example 1 Let us consider figure 1. By considering the object &19 we are provided with $p^1 = \{category\&17\}$ which is a single path starting from the object &19 and ending with &17. In the same way we can found the two following single paths: $p^2 = \{name\&13\}$ and $p^3 = \{address\&14\}$. As they have the same object source (&19), they can be merged in a multiple path expression $p = \{p^1, p^2, p^3\}$. In a more classical way from the object &19, we have: $p = \{category$: &17, *name*: &13, *address*: &14$\}$. Without considering atomic values, we are thus provided with the following multiple path expression $p = \{category\colon \perp, name\colon \perp, address\colon \perp\}$. If we now consider target object of the restaurant label in the OEM graph, we also obtain the two following multiple path expressions: $p' = \{category\colon \perp, name\colon \perp, address\colon \perp, address\colon \perp, zipcode\colon \perp, price\colon \perp\}$ and $p'' = \{price\colon \perp, category\colon \perp, name\colon \perp\}$. The length of all these paths is 1. Let us now have a closer look to the object source &19. From this object we also find the following multiple path expression $\{category\colon \perp, name\colon \perp, address\colon \{street\colon \perp, city\colon \perp, zipcode\colon \perp\}, nearby\colon \{price\colon \perp, category\colon \perp, name\colon \perp\}, nearby\colon \{category\colon \perp, name\colon \perp, address\colon \perp, address\colon \perp, price\colon \perp, zipcode\colon \perp\}\}$. The length of this new path is 2 corresponding to the length of the maximal single paths.

A multiple path expression p_m is a *sub-path expression* of another multiple path expression p_n if every object of p_m is included in p_n where the inclusion is defined as follows. If the object is a set value, such as $\{x_1, ...x_l\} \subseteq p_m$ and $\{x'_1, ...x'_k\} \subseteq p_n$ then the object of p_m is included in the object of p_n if and only if every x_i is a subset of some x'_j. If the object is a list value, such as $< x_1, ...x_l > \subseteq p_m$ and

[1]In fact, we may consider that a multiple path expression is an OEM graph where the root of the graph is the source object of single paths embedded in the multiple path expression.

Trans_id	**path expressions**
t_1	$person : \{identity\text{: } \{name\text{: } \perp,\ address;\ \perp\}\}$
t_2	$person : \{identity\text{: } \{name\text{: } \perp,\ address\text{: } < street\text{: } \perp, zipcode\text{: } \perp >,$ $company\text{: } \perp, director\text{: } < name\text{: } \perp, firstname\text{: } \perp >\}\}$
t_3	$person : \{identity\text{: } \{id\text{: } \perp, address\text{: } < street : \perp,\ zipcode : \perp >\}\}$
t_4	$person : \{identity\text{: } \{name\text{: } \perp,\ address\text{: } \perp,\ company\text{: } \perp\}\}$
t_5	$person : \{identity\text{: } \{name\text{: } \perp,\ address;\ \perp\}\}$
t_6	$person : \{identity\text{: } \{name\text{: } \perp,\ address\text{: } < street\text{: } \perp, zipcode\text{: } \perp >,$ $director\text{: } < name\text{: } \perp, firstname\text{: } \perp >\}\}$

Figure 2: A transaction database

$< x'_1, ...x'_k > \subseteq p_n$ then the object of p_m is included in the object of p_n if and only if there exist integers $i_1 < i_2 < ... < i_n$ such that $x_1 \subseteq x'_{i_1}$, $x_2 \subseteq x'_{i_2}$ ... $x_2 \subseteq x'_{i_n}$. Furthermore we assume that an atomic value is included in itself.

Example 2 For example, the path expression $\{category\text{: } \perp, name\text{: } \perp, address\text{: } \{street\text{: } \perp, city\text{: } \perp, zipcode\text{: } \perp\}\}$ is a sub-path expression of $\{category\text{: } \perp, name\text{: } \perp, address\text{: } \{street\text{: } \perp, city\text{: } \perp, zipcode\text{: } \perp\}, nearby\text{: } \{price\text{: } \perp, category\text{: } \perp, name\text{: } \perp\}, nearby\text{: } \{category\text{: } \perp, name\text{: } \perp, address\text{: } \perp, address\text{: } \perp, price\text{: } \perp, zipcode\text{: } \perp\}\}$. However the path $\{category\text{: } \perp, name\text{: } \perp, address\text{: } \{street\text{: } \perp, city\text{: } \perp, zipcode\text{: } \perp\}, price\text{: } \perp\}$ is not a sub-path of $\{category\text{: } \perp, name\text{: } \perp, address\text{: } \{street\text{: } \perp, city\text{: } \perp, zipcode\text{: } \perp\}, nearby\text{: } \{price\text{: } \perp, category\text{: } \perp, name\text{: } \perp\}, nearby\text{: } \{category\text{: } \perp, name\text{: } \perp, address\text{: } \perp, address\text{: } \perp, price\text{: } \perp, zipcode\text{: } \perp\}\}$ since the label *price* is not at the same level in the graph.

Let DB be a set of transactions where each transaction T consists of transaction-id and a multiple path expression embedded in the OEM graph and involved in the transaction. All transactions are sorted in increasing order and are called a *data sequence*. Figure 2 gives an exemple of transactions embedded in DB. A support value (*supp(s)*) for a multiple path expression in the OEM graph gives its number of actual occurrences in DB. In other words, the support of a path is defined as the fraction of total data sequences that contain *p*. A data sequence contains a path p if p is a sub-path expression of the data sequence. In order to decide whether a path is frequent or not, a minimum support value (*minSupp*) is specified by user, and the multiple path expression is said *frequent* if the condition $supp(s) \geq minSupp$ holds.

2.2 Problem statement

Given a database DB of customer transactions the problem of regularity mining, called *Schema mining*, is to find all maximal paths occurring in DB whose support is greater than a specified threshold (minimum support). Each of which represents a *frequent path*.

From the problem statement presented so far, discovering structural association sequential patterns resembles closely to mining association rules [AIS93, AS94, BMUT97, FPSSU96, SON95, Toi96] or sequential patterns [AS95, SA96]. However, elements of handled association transactions have structures in the form of a labeled hirearchical objects, and a main difference is introduced with partially ordered references. In other word we have to take into account complex structures while in the association rules or sequential patterns elements are atomics, i.e. flat sets or lists of items.

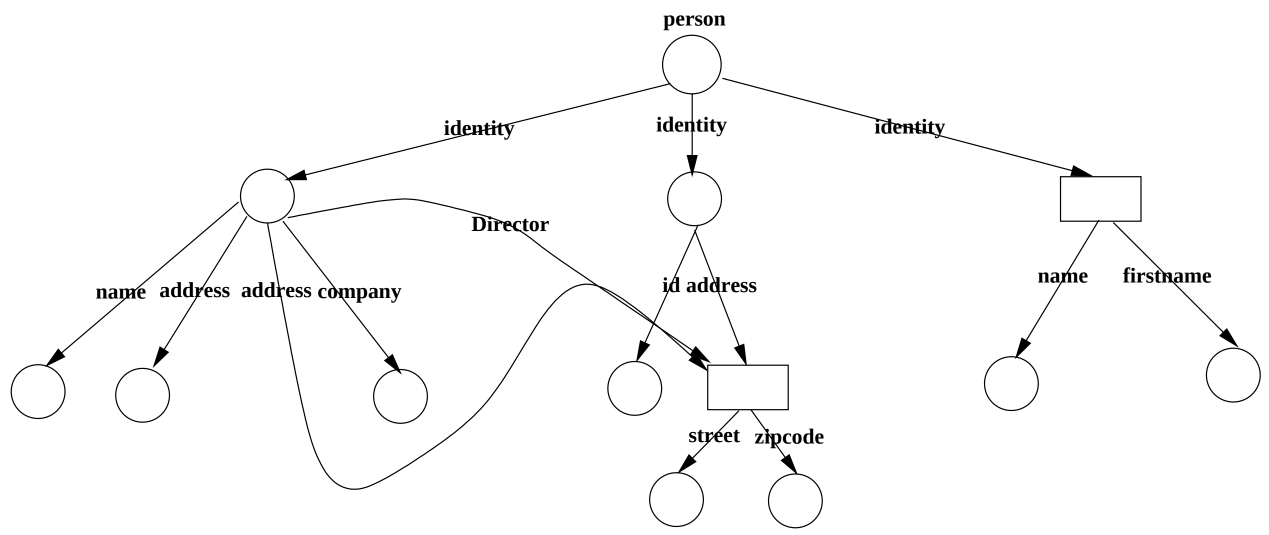

Figure 3: An OEM graph

2.3 An Example

In order to illustrate the problem, let us consider the base D given in figure 2, reporting transactions about a population merely reduced to six. Let us assume that the minimum support value is 50%, thus to be considered as frequent a path must be observed for at least 3 transactions. Let us now consider the associated OEM graph given in figure 3. The only frequent paths, embedded in the DB are the following: *identity*: $\{$*name*: $\perp$, *address*: $\perp\}$ and *identity*: $\{$*address*: $<$ *street*: $\perp$, *zipcode*: $\perp >\}$. The fist one is discovered because it matches with the first transaction t_1 while being detected for t_4 and t_5. In the same way *identity*: $\{$*address*: $<$ *street*: $\perp$, *zipcode*: $\perp >\}$ is supported by transactions t_2, t_3 and t_6. For instance the multiple path expression *identity*: $\{$*name*: $\perp$, *address*: $<$ *street*: $\perp, zipcode$: $\perp >, director$: $<$ *name*: $\perp, firstname$: $\perp >\}$ is supported by transaction t_2 and t_6 but it is not frequent since the number of data sequences supporting this path does not verify the mininum support constraint.

3 Principles

For presenting our approach, we adopt the chronological viewpoint of data processing: from collected raw data to exhibited knowledge. We consider that the mechanism for discovering regularities on semistructured data in a large database is a 2-phase process. The starting point of the former phase is semistructured objects extracted from sources on the Web and collected in a large file.
From such a file, the mapping phase performs a transformation of original data. It results in a new populated database containing the meaningful remaining data. From such a database, data mining technique is applied in order to extract useful regularities on graph structures. The architecture of our approach is depicted in figure 4.

3.1 Data extraction

The structure of each object embedded in the source must be extracting during this phase. First of all, a data filtering step is performed in order to filter out irrelevant semistructured data such as image, video, sound, etc. Furthermore, according to the end user view point not interesting substructures are also prunned out.
In order to perform such an extraction, we assume that the end user is provided with a parser or a

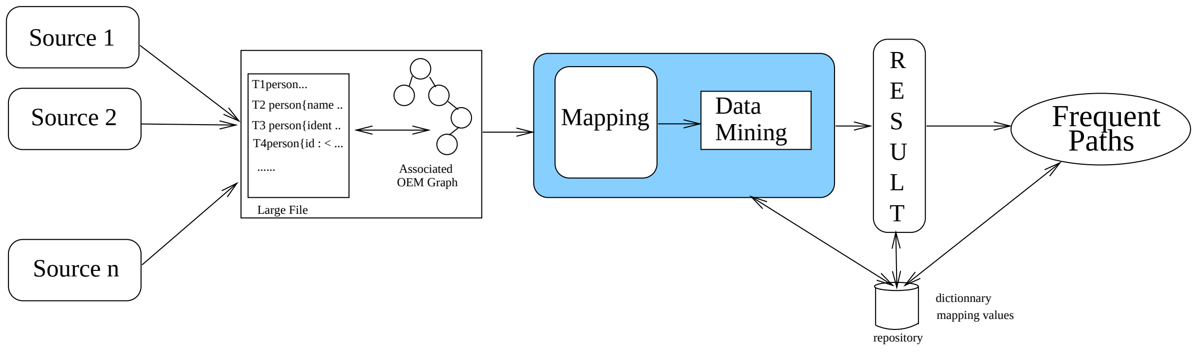

Figure 4: General Architecture

wrappers. Of course, this parser requires the implementation of efficient algorithms for extracting graph structure. During our experiments, we assumed that such extraction has been done. For instance, in [HGMC$^+$97], a very efficient tool for extracting semistructured data from a set of HTML pages and for converting the extracted information into an OEM graph is addressed. This extraction is done on differents sources in order to provide a large collection of graph structure where we want to discover substructures appearing in some number of graph structures.

3.2 Knowledge Discovery

From the data yielded by the extraction phase, data mining technique is applied for fully meeting the analyst needs.
We split the problem of mining structural association of semistructured data in a large database into the following sub-phases: mapping and data mining.

3.2.1 Mapping phase

The transaction database is sorted with transaction id as a major key and values embedded in a set-of are sorted according to the lexicographic order. In order to efficiently find structural regularity among path expressions, each path expression is mapped in the following way. If the object value is a part of a *set-of* value, then we merge an 'S' to the label (resp. a 'L' for a *list-of* value). Furthermore, in order to take into account the level of the label into the transaction, we append to the label an integer standing for the level of the label in the nested structure. When two labels occur at the same level and if the first one directly follows the second one, they are grouped together in the same set otherwise they form a new set of labels. The ordering of the *list-of* value is taken into account by creating a new set of labels. The "composite" transaction which results from the union of such sets obtained from original transaction describe a *sequence of label* and the ordering of such a sequence may be seen as the way of navigating through the path expression.
This step converts the original original database into a database D of data-sequences where each data-sequence describes the ordering of labels into the transaction according to the path expression.

Example 3 In order to illustrate the mapping phase, let us consider the two following transactions $[t_1, \{a : \{c : \{b : \bot, f : \bot\}\}\}]$ and $[t_2, \{a :< e : \{f : \bot, b : \bot\}, d :< g : \bot, h : \bot >>, c : \bot\}]$. Considering the fisrt one transaction, each label occurs in a set-of value then we merge an 'S' to all the labels occurring into the transaction. Now we can append an integer standing for the level of the label into the transactions, original labels a, c, b, f are thus transformed as follows: Sa_1, Sc_2, Sb_3, Sf_3. Labels occurring in the same level are grouped together and the transformation of the original transaction becomes: $[t_1, (Sa_1)\ (Sc_2)(Sb_3Sf_3)]$. According to the previous discussion, the transaction t_2 is mapped into the following: $[t_2, (Sa_1)\ (Le_2)\ (Sf_3Sb_3)\ (Lg_3)(Lh_3)(Ld_2)\ (Sc_1)]$. Let us have a closer look at the

Trans_id	path expressions mapped
t_1	$(Sidentity_1)$ $(Sname_2\ Saddress_2)$
t_2	$(Sidentity_1)$ $(Sname_2\ Saddress_2)$ $(Lstreet_3)(Lzipcode_3)$
	$(Scompany_2\ Sdirector_2)$ $(Lname_3)$ $(Lfirstname_3)$
t_3	$(Sidentity_1)$ $(Sid_2\ Saddress_2)$ $(Lstreet_3)$ $(Lzipcode_3)$
t_4	$(Sidentity_1)$ $(Sname_2\ Saddress_2\ Scompany2)$
t_5	$(Sidentity_1)$ $(Sname_2\ Saddress_2)$
t_6	$(Sidentity_1)$ $(Sname_2\ Saddress_2)$ $(Lstreet_3)$ $(Lzipcode_3)$
	$(Sdirector_2)$ $(Lname_3)$ $(Lfirstname_3)$

Figure 5: A transaction database mapped

list-of part $< g : \perp, h : \perp >$ of the transaction. As labels g and h are ordered according to the list-of value, we consider that even if they occur in the same level and if h directly follows g, they are not grouped into the same set.

Figure 5 describes the database of our previous example after the mapping phase.

3.2.2 Mining Phase

From the database obtained from the mapping phase, the problem resembles closely to mining association rules (also known as *market-basket* proble) which was initially introduced in [AIS93] where association could be seen as relationships between facts, embedded in the database. Nevertheless our problem is quite different since we have to take into account hierarchical structure. In fact, our approach is very similar to the problem of sequential pattern which is introduced to capture typical behaviour over time, i.e. behaviours sufficiently repeated by individuals to be relevant for the decision maker [AS95]. In this context, we assume that we are given a database D of customers' transactions, each of which having the following characteristics: sequence-id or customer-id, transaction-time and the items involved in the transaction. Such a database is called a base of data sequences (C.f. Fig. 2).

Definition 1 Let $I = \{i_1, i_2, ..., i_m\}$ be a set of literals called *items*. An *itemset* is a non-empty set of items. A sequence s is a set of itemsets ordered according to their time-stamp. It is denoted by $< s_1 s_2 ... s_n >$ where s_j is an itemset. A *k-sequence* is a sequence of k-items (or of length k).
A sequence $< s_1 s_2 ... s_n >$ is a sub-sequence of another sequence $< s'_1 s'_2 ... s'_m >$ if there exist integers $i_1 < i_2 < ... < i_n$ such that $s_1 \subseteq s'_{i_1}, s_2 \subseteq s'_{i_2}, ... s_n \subseteq s'_{i_n}$.

Example 1 Let us consider that a given customer purchased items A, B, C, D, E, according to the following sequence: $s =<$ (A) (B, C) (D) (E)$>$. This means that apart from B and C which were purchased together, i.e. during a common transaction, items in the sequence were bought separately. The sequence $s' = <$ (B) (E) $>$ is a sub-sequence of s because (B) $\subseteq$ (B, C) and (E) $\subseteq$ (E). However $<$ (B) (C) $>$ is not a sub-sequence of s since items were not bought during the same transaction.

For aiding efficiently decision making, the aim is discarding non typical behaviours according to user's viewpoint. Performing such a task requires providing data sub-sequence s in the DB with a support value ($supp(s)$) giving its number of actual occurrences in the DB. In order to decide whether a sequence is frequent or not, a minimum support value (σ) is specified by user, and the sequence s is said frequent if the condition $supp(s) \geq \sigma$ holds.
The interested reader could refer to [AS95, SA96, MCP98] in which approaches for exhibiting sequences are presented and compared.

Our approach for mining structural regularities fully resumes the fundamental principles sequential pattern problem. In order to improve the efficiciency of retrievals we use an algorithm, called PSP, that we defined for mining sequential patterns. It resumes the principles of GSP [SA96] but it makes use of a different intermediary data structure which is proved to be more efficient than in GSP. Due to lack of space we do not detail the algorithm but interested reader may refer to [MCP98, MPC00].

As illustration, when running our algorithm on the database of the figure 5 with a support value of 50%, we obtain the following frequent paths: $< (Sidentity_1)\ (Sname_2\ Saddress2) >$ and $< (Sidentity_1)\ (Sname_2)\ (Lstreet_3)\ (Lzipcode_3) >$. These results may thus be transformed according to information obtained from the mapping phase and we are thus provided with the following semistructured graphs: *identity*: $\{$*name*: $\perp$, *address*: $\perp\}$ and *identity*: $\{$*address*: $<$ *street*: $\perp$, *zipcode*: $\perp >\}$.

The data mining algorithm is implemented using Gnu C++ and preliminary results show that the approach is efficient. Experiments have been performed on synthetic datasets which mimic real world semistrcutured graphs. Due to lack of space we do not report experiments but interested reader may refer to [Lau00] and [MPC00] where the efficiency of a quite similar structure, even if defined in an other context, is proved.

4 Related Work

To the best of our knowledge there is few work on mining such a structural regularity in a large database. Nevertheless, our work is very related to the problem of mining structural association of semistructured data proposed in [WL98, WL99] where a very efficient approach for mining such regularities is provided. The author propose a very efficient approach and solutions based on a new representation of the search space. Furthermore they give some pruning strategies in order to improve the candidate generation. Nevertheless our work has some important differences. Unlike their approach we are insterested in all structures embedded in the database while they are interested in mining tree expression which are defined as a path from the root of the OEM graph to the atomic values. According to this definition of the tree expression they cannot find regularities such as *identity*: $\{$*address*: $<$ *street*: $\perp$, *zipcode*: $\perp >\}$ (Cf. section 2). In fact, when parsing the database in order to find frequent tree, they are only provided with maximal tree and when only a part of the tree is frequent it is not discovered.
In [HF95, SA95] approaches for mining multi-level association rules are addressed. Authors assume that they are provided with a database of customer transactions and a taxonomy (*is-a* hierachy) on the items. They are interested in finding itemsets from different levels of the taxonomy. Nevertheless, these approaches are quite differents from our work. They consider the problem of data enrichment by an is-a hierarchy while we are interested in transactions where the data enrichment is done through set-of and list hierarchy.
Discovering structural information from semistructured data has been studied in some interesting works. In this context, they are some propositions for extracting the typing of semistructured data [NUWC97, BDHS97, NAM98]. For instance in [NAM98] they extract the structure implicit in a semistructured schema. This approach is quite different from our since we address the repetition of a structure in a schema. Nevertheless we assume that our transaction database has been preprocessed using such a technique in order to provide an OEM graph where the raw data has been casted in terms of this structure. Related to the problem but in the context of text searching, in [BYG96], the authors present algorithms for searching of regular expressions on preprocessed text. They propose a very efficient algorithm according to the complexity.

5 Conclusion

In this paper we present an approach for mining regularities of semistructured objects in a large database. This approach is based on a data mining technique and preliminary results show that such a technique may be useful in order to discover schema regularities on the Web. We have defined the problem and proposed a very efficient approach to solve it.

Future work will be done in various directions. First, we aim to apply our approach on semistructured documents on the Web. For the moment, the efficiency and effectiveness of our approach were only evaluated on synthetic datasets and we have to define a new parser in order to valid our approach on Web sources. Second, assuming that document format is HTML, we think that discovered regularities may be useful for restructuring a Web site for instance with a semi-automatic generation of DTD in the XML format. Finally, we are also investigating how to improve the representation of the OEM graph in order to take into account cyclic structures.

References

[Abi97] S. Abiteboul. Querying Semi-Structured Data. In *Proceedings of International Conference on Database Theory (ICDT'97)*, pages 1–18, Delphi, Greece, January 1997.

[ABS00] S. Abiteboul, P. Buneman, and D. Suciu. *Data on the Web.* Morgan Kaufmann, 2000.

[AIS93] R. Agrawal, T. Imielinski, and A. Swami. Mining Association Rules between Sets of Items in Large Databases. In *Proceedings of the 1993 ACM SIGMOD Conference*, pages 207–216, Washington DC, USA, May 1993.

[AQM$^+$97] S. Abiteboul, D. Quass, J. McHugh, J. Widom, and J.L Wiener. The Lorel Query Language for Semi-Structured Data. *International Journal on Digital Libraries*, 1(1):68–88, April 1997.

[AS94] R. Agrawal and R. Srikant. Fast Algorithms for Mining Generalized Association Rules. In *Proceedings of the 20th International Conference on Very Large Databases (VLDB'94)*, Santiago, Chile, September 1994.

[AS95] R. Agrawal and R. Srikant. Mining Sequential Patterns. In *Proceedings of the 11th International Conference on Data Engineering (ICDE'95)*, Tapei, Taiwan, March 1995.

[BDHS96] P. Buneman, S. Davidson, G. Hillebrand, and D. Suciu. A Query Language and Optimization Techniques for Unstructured Data. In *Proceedings of the International Conference on Management of Data (SIGMOD'96)*, pages 505–516, Montreal, Canada, May 1996.

[BDHS97] P. Buneman, S. Davidson, G. Hillebrand, and D. Suciu. Adding Structure to Unstructured Data. In *Proceedings of International Conference on Database Theory (ICDT'97)*, pages 336–350, Delphi, Greece, January 1997.

[BMUT97] S. Brin, R. Motwani, J.D. Ullman, and S. Tsur. Dynamic Itemset Counting and Implication Rules for Market Basket Data. In *Proceedings of the International Conference on Management of Data (SIGMOD'97)*, pages 255–264, Tucson, Arizona, May 1997.

[BYG96] R.A. Baeza-Yates and G.H. Gonnet. Fast Text Searching for Regular Expressions or Automaton Searching on Tries. *Journal of the ACM*, 43(6):915–936, November 1996.

[FPSSU96] U.M. Fayad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors. *Advances in Knowledge Discovery and Data Mining.* AAAI Press, Menlo Park, CA, 1996.

[HF95] J. Han and Y. Fu. Discovery of Multiple-Level Association Rules from Large Databases. In *Proceedings of the 21 st International Conference on Very Large Databases (VLDB'95)*, pages 420–431, Zurich, Switzerland, September 1995.

[HGMC$^+$97] J. Hammer, H. Garcia-Molina, J. Cho, R. Aranha, and A. Crespo. Extracting Semistructured Information from the Web. In *Proceedings of the Workshop on Management of Semistructured Data. See [Wor97]*, Tucson, Arizona, May 1997.

[KS95] D. Konopnicki and O. Shmueli. W3QS: A Query System for the World-Wide Web. In *Proceedings of the 21 st International Conference on Very Large Databases (VLDB'95)*, pages 54–65, Zurich, Switzerland, September 1995.

[Lau00] P.A. Laur. Schema Mining: vers une approche efficace. Technical Report (in preparation), LIRMM, France, June 2000.

[MCP98] F. Masseglia, F. Cathala, and P. Poncelet. The PSP Approach for Mining Sequential Patterns. In *Proceedings of the 2nd European Symposium on Principles of Data Mining and Knowledge Discovery (PKDD'98), LNAI, Vol. 1510*, pages 176–184, Nantes, France, September 1998.

[MPC00] F. Masseglia, P. Poncelet, and R. Cicchetti. An Efficient Algorithm for Web Usage Mining. *Networking and Information Systems Journal*, (to appear), April 2000.

[NAM98] S. Nestorov, S. Abiteboul, and R. Motwani. Extracting Schema from Semistructured Data. *Proceedings of the International Conference on Management of Data (SIGMOD'98) - SIGMOD record*, 27(2), 1998.

[NUWC97] S. Nestorov, J. Ullman, J. Wiener, and S. Chawathe. Representative Objects: Concise Representations of Semistructured, Hierarchical Data. In *Proceedings of the 13th International Conference on Data Engineering (ICDE'97)*, pages 79–90, Birmingham, U.K., April 1997.

[SA95] R. Srikant and R. Agrawal. Mining Generalized Association Rules. In *Proceedings of the 21 st International Conference on Very Large Databases (VLDB'95)*, pages 407–419, Zurich, Switzerland, September 1995.

[SA96] R. Srikant and R. Agrawal. Mining Sequential Patterns: Generalizations and Performance Improvements. In *Proceedings of the 5th International Conference on Extending Database Technology (EDBT'96)*, pages 3–17, Avignon, France, September 1996.

[SON95] A. Savasere, E. Omiecinski, and S. Navathe. An Efficient Algorithm for Mining Association Rules in Large Databases. In *Proceedings of the 21 st International Conference on Very Large Databases (VLDB'95)*, pages 432–444, Zurich, Switzerland, September 1995.

[Toi96] H. Toivonen. Sampling Large Databases for Association Rules. In *Proceedings of the 22nd International Conference on Very Large Databases (VLDB'96)*, September 1996.

[WL98] K. Wang and H.Q. Liu. Discovering Typical Structures of Documents: A Road Map Approach. In *Proceedings of ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 146–154, Melbourne, Austrialia, August 1998.

[WL99] K. Wang and H. Liu. Discovering Structural Association of Semistructured Data. *IEEE Transactions on Knowledge and Data Engineering*, 1999.

[Wor97] Work97. The Workshop on Management of Semistructured Data. In *www.research.att.com/s̃uciu/workshop-papers.html*, Tucson, Arizona, May 1997.