Human-centric Computing and Information Sciences
a SpringerOpen Journal

RESEARCH Open Access

MR-Radix: a multi-relational data mining algorithm

Carlos Roberto Valêncio[1*], Fernando Takeshi Oyama[1], Paulo Scarpelini Neto[1], Angelo Cesar Colombini[2], Adriano Mauro Cansian[1], Rogéria Cristiane Gratão de Souza[1] and Pedro Luiz Pizzigatti Corrêa[3]

* Correspondence: valencio@ibilce.unesp.br
[1]São Paulo State University - Unesp, Rua Cristóvão Colombo, 2265 São José do Rio Preto, São Paulo, Brazil
Full list of author information is available at the end of the article

Abstract

Background: Once multi-relational approach has emerged as an alternative for analyzing structured data such as relational databases, since they allow applying data mining in multiple tables directly, thus avoiding expensive joining operations and semantic losses, this work proposes an algorithm with multi-relational approach.

Methods: Aiming to compare traditional approach performance and multi-relational for mining association rules, this paper discusses an empirical study between PatriciaMine - an traditional algorithm - and its corresponding multi-relational proposed, MR-Radix.

Results: This work showed advantages of the multi-relational approach in performance over several tables, which avoids the high cost for joining operations from multiple tables and semantic losses. The performance provided by the algorithm MR-Radix shows faster than PatriciaMine, despite handling complex multi-relational patterns. The utilized memory indicates a more conservative growth curve for MR-Radix than PatriciaMine, which shows the increase in demand of frequent items in MR-Radix does not result in a significant growth of utilized memory like in PatriciaMine.

Conclusion: The comparative study between PatriciaMine and MR-Radix confirmed efficacy of the multi-relational approach in data mining process both in terms of execution time and in relation to memory usage. Besides that, the multi-relational proposed algorithm, unlike other algorithms of this approach, is efficient for use in large relational databases.

Keywords: MR-Radix, Multi-relational data mining, Association rules, Mining frequent itemsets, Relational databases

Introduction

Data mining has emerged as a field of study aimed at developing tools and techniques for the exploration of large data repositories, in order to obtain new, valuable, non-trivial and implicitly existing information [1]. Data mining processing traditional algorithms are carried out taking into account that the data are arranged in a single structure, usually a file or a table. This limitation hinders the use of such algorithms, for example, in a relational database which consists of several tables semantically related [2].

One possibility to bypass this limitation is to join all the tables in a universal table and subsequently apply the data mining techniques. However, this approach could result in a universal table of such a size that it would make the application of traditional techniques unviable.

Other possibility is applying said algorithms to a central table which is made up of attributes that summarise or aggregate the information found in other tables. Nevertheless, this technique has the disadvantage of generating tables having many attributes and data repetitions [3].

Multi-relational data mining algorithms emerged to enable the extraction of multiple relation patterns, with efficiency and efficacy, without the necessity of joining the data into a single table [4,5]. Said algorithms made possible the extraction of knowledge in situations where, due to their semantic relevance, it is important to maintain the structure or relationship of the multiple tables and has been applied in several areas [6-8].

However, for multi-relational data mining in large databases, such algorithms were not efficient due the difficulty of allocating enough memory for all data structure used for this algorithm to represent the database. To bypass the scalability problem with this strategy, the MR-Radix was proposed.

The MR-Radix is a multi-relational data mining algorithm, which has a data structure called Radix-tree that compresses the database in the memory. To demonstrate the efficiency of this algorithm, this work presents a comparative study of MR-Radix and its corresponding multi-relational, the traditional algorithm PatriciaMine. Therefore, two original contributions are obtained: the MR-Radix multi-relational algorithm proposal and the empiric presentation of advantages of the multi-relational approach for mining various tables.

This work is organised in the following manner: section 2 lists the principal concepts about mining association rules-traditional and multi-relational; section 3 presents a comparison between PatriciaMine and other algorithms; section 4 presents the proposed MR-Radix algorithm; section 5 presents a comparison between PatriciaMine and MR-Radix algorithms; lastly, section 6 presents conclusions about the work.

Theory substantiation

Data Mining can be defined as being the use of computational techniques to extract knowledge from a set of data which, generally, is large enough to make human analysis impracticable [9]. Association rules, which have the objective of discovering associative patterns from databases, can be found among these techniques. The classic example that illustrates mining association rules is called "market basket analysis," which identifies associations between items so that, frequently, the presence of some items implies the presence of others [10].

An association rule can be seen as an implication $X \rightarrow Y$, with X and Y being sets of items. Such sets are called itemsets, also commonly referred to as k-itemsets, in which k is the number of items that said set has. That pattern indicates an association between the antecedent set (X) and the consequent set (Y), so that the occurrence of X implies the occurrence of Y. Two interesting measures, called support and confidence [10], are used to obtain and quantify association rules.

The support represents the association rule frequency, that is, indicates the percentage of concomitant occurrence of the X and Y sets in the database. The confidence measurement indicates the frequency in which the occurrence of X set items implies the occurrence of Y set items. Said measurement expresses the validity of the association rule X→Y [10].

Mining association rules algorithms

In this section, some of the mining association rules algorithms found in literature are presented. The best known are: Apriori [1], which uses a "candidate generation-and-test" (CGT); and the FP-growth [2], which adopts the pattern-growth (PG) paradigm.

The FP-growth algorithm uses an FP-tree data structure, which creates a sub-tree for each recursive call and favours the mining of frequent itemsets from dense databases [2]. Differently to FP-growth, the TD-FP-growth algorithm does a top-down processing, it being more efficient in the use of time and memory [11].

Some algorithms, such as H-Mine [12] and OpportuneProject [13], make use of two types of structures to mine dense and sparse databases. Because the structure to do the mining depends on the type of base, said algorithms use heuristics to assess the density of a database and choose the best strategy. In the case of dense databases, tries are used as this type of base generates few branches and intensifies mining efficiency. For sparse databases, structures based on arrays are used as these bases generate many branches in the tree, significantly increasing the size of these structures and, consequently, the time needed for processing [12,13].

The PatriciaMine [14] algorithm is based on the PG approach, but has a superior performance in comparison to the FP-growth. This is due to the use of a more optimised structure called Patricia-trie (Practical Algorithm To Retrieve Information Coded In Alphanumeric) or Radix-tree. Said structure is efficient in its mining algorithm performance and in its use of memory. Moreover, the Patricia-trie structure has a high compression capacity which reduces needed space for its allocation in memory, when compared to the FP-tree standard, by up to 75%, which is the standard trie structure used in several PG approach algorithms [14]. Besides that, the Radix-tree presented a lower computational cost than GFP-tree, a structure of GFP-Growth algorithm, since the Radix-tree enables a good performance in dense and sparse databases as the nodes in them are compressed, thus needing a smaller memory space in which to store the tree.

The Radix-tree structure

The structure of Radix-tree performs data compression by means of cluster nodes that share the same branch. An example of Radix-tree structure is presented in Figure 1.

It can be verified that the items A, D and F are located in the same node, and that the support count is 4. This means that the data set shown in Figure 1 has four entries for the items A, D and F. In a standard trie such items would be represented by four distinct nodes for each item, but in Radix-tree only one node is necessary, reducing the memory space used to store such structure.

Comparison between the PatriciaMine and other algorithms

The PatriciaMine is one of the most efficient mining association rules algorithms existing in literature, having also a better performance than the OpportuneProject

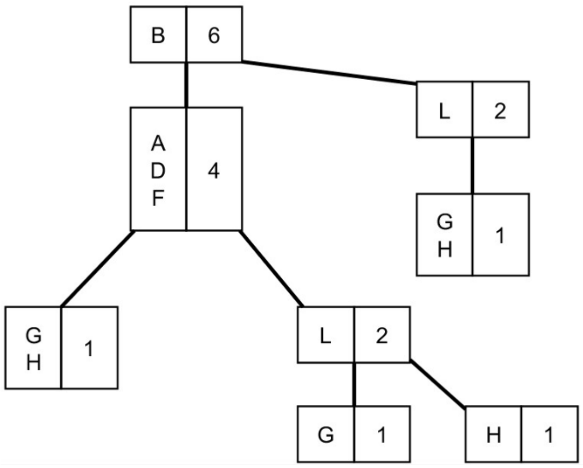


Figure 1 Example of Radix-tree structure.

algorithm, and is considered as being one of the most advanced pattern-growth approach algorithms [14-16].

A graphic in Figure 2 is exhibited, showing the execution times of the PatriciaMine and OpportuneProject algorithms for a database having approximately 50 thousand registers, called Pumsb. For a better visualisation of the graphic, a logarithmic scale was used for the axis Y, corresponding to the time of execution.

Graphic in Figure 3 shows the execution time of the PatriciaMine compared to recent algorithms having a similar performance. The comparative results were obtained by consulting the IEEE Workshop on Frequent Itemset Mining Implementations website [17]. The graphic in Figure 4 shows a comparison, between PatriciaMine and other algorithms, of the amount of occupied principal memory space needed to do the mining.

Although the algorithms show proximate execution times, the PatriciaMine excels with its smaller amount of principal memory space needed to do the mining, as illustrated in Figure 4. That positive characteristic has a direct relationship to the Patricia-trie data structure which, more efficiently, represents the set of frequent items in the memory. On the other hand, the Patricia-trie is capable of partially overcoming the problem of database sparsity as it compresses a representation of the database.

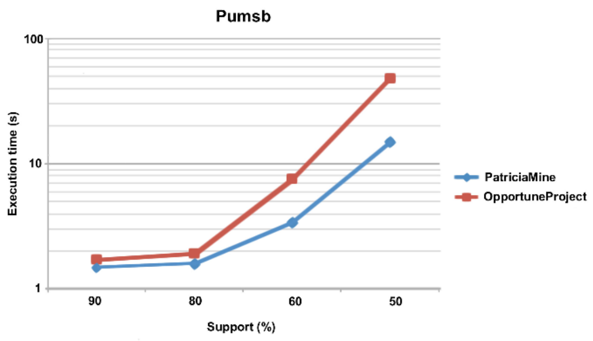


Figure 2 Graphic of execution times of the PatriciaMine and OpportuneProject algorithms [13].

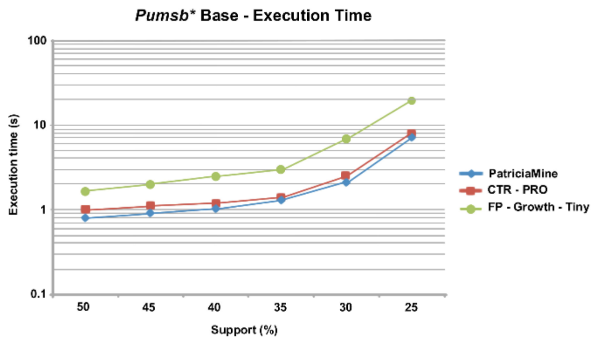


Figure 3 Comparative graphic of the execution time of the PatriciaMine, CT-PRO and FP-Growth-Tiny algorithms.

PatriciaMine performance was also compared to other traditional algorithms of mining association rules, namely Apriori, FP-Growth and FP-Growth* [18]. The last one is an optimized implementation of the algorithm FP-Growth. The Connect database was used, containing more than 67 million records, and a graphic comparing the runtime of such algorithms is shown in Figure 5.

Given the analyses of different association rule algorithms, the PatriciaMine was chosen to be the basis for MR-Radix, since it presents a more efficient data structure, provides performance gains, and optimises the use of space in memory.

Traditional versus multi-relational mining association rules

Traditional data mining algorithms, such as Apriori and FP-growth, seek patterns from data that are ordered in a single structure like a table or file. To apply these algorithms in relational databases, the data from a set of relations must go through a pre-processing stage, in which they must be reunited in a single table by means of a joining or aggregating operation.

Although this proposal is possible and sufficient for some applications, the use of traditional mining algorithms in multiple tables can produce unsatisfactory results, for example, jeopardised algorithm performance, since a joining process could generate a table with many registers when the extraction is done from large databases, as well as the appearance of inconsistencies or loss of information during the data pre-processing [4,19]. To illustrate this problem, Figure 6 shows an example of joining tables in a

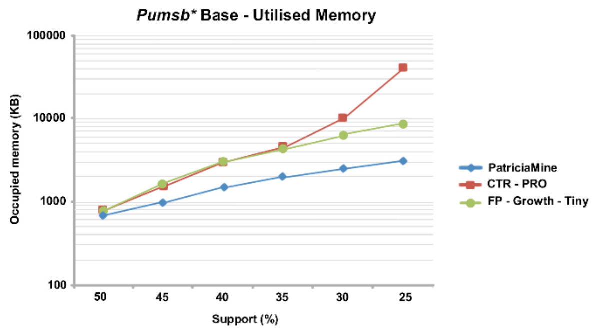


Figure 4 Comparative graphic of occupied memory space of the PatriciaMine, CT-PRO and FP-Growth-Tiny algorithms.

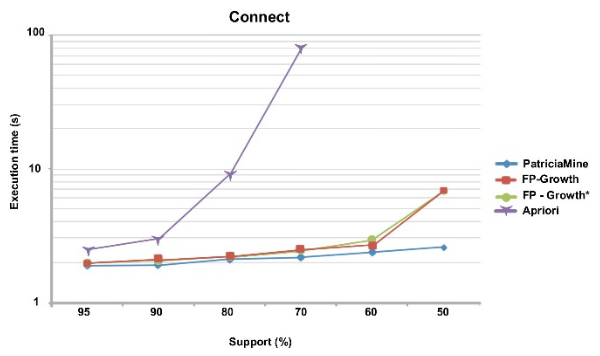
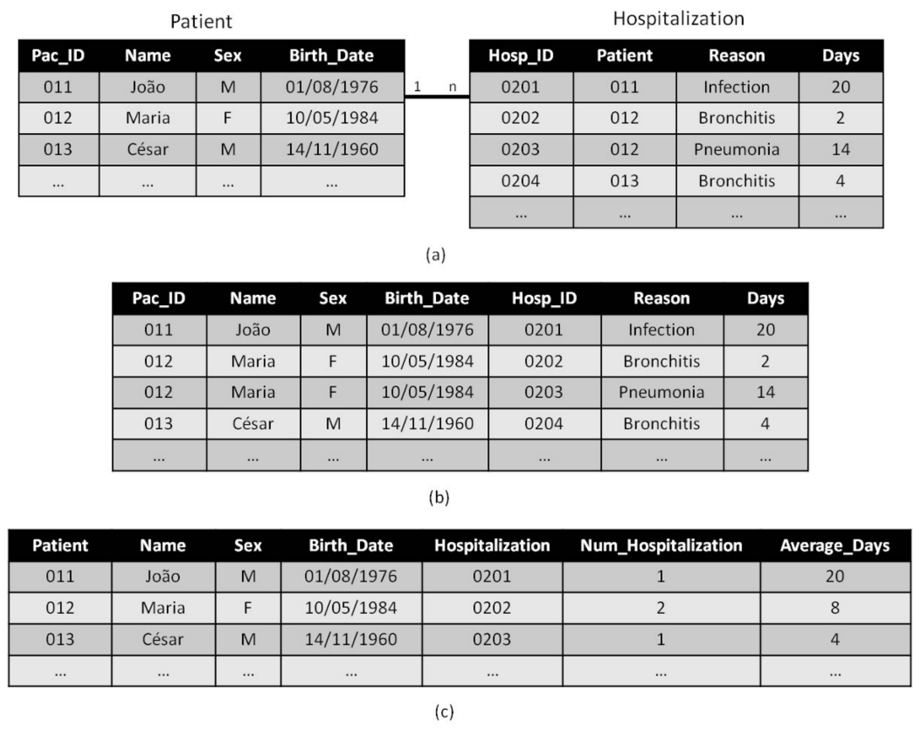


Figure 5 Comparative graphic of PatriciaMine, FP-Growth, FP-Growth* and Apriori [18]

database for a simplified hospital in which data relating to patients and their hospitalizations are stored.

Figure 6(a) shows a table that is associated to a patient and hospitalization table through a relationship of type '1 to n' - one to many. Figure 6(b) displays the result of joining patient and hospitalization data. The junction creates a table containing the data from Patient and Hospitalization tables through the existing foreign key relationship. The resulting table shows data redundancy for the patient Maria, leading to inaccuracies in the results.

Figure 6(c) shows the use of aggregate functions for data generalization, in order to eliminate redundancies from the table in Figure 6(b). In the example, functions sum



Patient

Pac_ID	Name	Sex	Birth_Date
011	João	M	01/08/1976
012	Maria	F	10/05/1984
013	César	M	14/11/1960
...	...	...	...

1 n

Hospitalization

Hosp_ID	Patient	Reason	Days
0201	011	Infection	20
0202	012	Bronchitis	2
0203	012	Pneumonia	14
0204	013	Bronchitis	4
...	...	...	...

(a)

Pac_ID	Name	Sex	Birth_Date	Hosp_ID	Reason	Days
011	João	M	01/08/1976	0201	Infection	20
012	Maria	F	10/05/1984	0202	Bronchitis	2
012	Maria	F	10/05/1984	0203	Pneumonia	14
013	César	M	14/11/1960	0204	Bronchitis	4
...	...	...	...	...	...	...

(b)

Patient	Name	Sex	Birth_Date	Hospitalization	Num_Hospitalization	Average_Days
011	João	M	01/08/1976	0201	1	20
012	Maria	F	10/05/1984	0202	2	8
013	César	M	14/11/1960	0203	1	4
...	...	...	...	...	...	...

(c)

Figure 6 a) Patient and hospitalization relations, b) Result of a natural joining of Patient and Hospitalization, c) Use of functions generating aggregate attributes Num_ Hospitalization and Average_ Days.

and average was used. The first function was used to count the number of admissions for each patient-Num_Hospitalization attribute-the second, in turn, was applied to calculate the average length of stay-Average_Days attribute.

The use of aggregate functions can also cause loss of information, such as occurs when using the mean value (Average_Days) to represent the number of days of hospitalization. The average patient days for Maria is 8 days, however, on checking the table hospitalization, it appears that there are two admissions of the patient of 2 and 14 days respectively. This difference may indicate a pattern associated with the reason for admission, so that the analysis of the Days attribute would then be more interesting than Average_Days. The aggregate functions can also cause the elimination of non-aggregatable attributes such as Reason attribute, since this attribute cannot be summarised, therefore it is removed from the resulting table.

Multi-relational mining association rules

Multi-relational mining is the most recent approach which aims to: overcome the difficulties that are found when applying traditional algorithm; and enable direct pattern extraction from multiple relations, without the necessity of transferring data to a single relation [4,5]. This would avoid costly joining operations and semantic losses caused by the representation limit of a single table.

The first viable multi-relational mining proposals go back to Inductive Logic Programming (ILP), having techniques based on pattern representation from first order logic [3], with its main representative being the WARMR algorithm [20]. Other existing ILP algorithms are the FARMER [21] and the RADAR [22], both proposing performance improvements of the WARMR.

The extraction of multi-relational association rules can also be done with graphs. To do this, mathematical theories are used to identify the set of sub-graphs that occur a determined number of times [23]. The AGN [24] algorithm, also based on Apriori, was one of the first solutions to use a graph mathematical theory for the extraction of frequent patterns. As from then, other techniques surfaced-such as FSG [25]-that aim to improve process performance. On the other hand, algorithms like the GBI [26] propose heuristic techniques that produce approximate solutions though with a better efficiency in the task of finding frequent patterns.

Another strategy for the extraction of multi-relational association rules is based on traditional algorithms, such as Apriori and FP-growth. Those algorithms are adapted to multi-relational data mining as their strategy is more complex for pattern processing. The presentation of some samples of multi-relational algorithms follows.

One example of the multi-relational algorithm is the MRFP-Growth [27] based on FP-growth. In a first stage it finds local frequent patterns from each of the tables and, in the next stage, these patterns are verified to obtain multi-relational patterns. Another example is the Apriori-Group [28] which is based on the Apriori. By means of the clustering obtained with this algorithm, data redundancy errors, deriving from the junction operations of multiple tables, are corrected.

The Connection algorithm [29] has a behaviour which is analogous to the MRFP-growth, is also based on the FP-growth and was initially idealised for use in data warehouses. The ConnectionBlock [30] is a modification of the Connection algorithm to consider the block concept, which consists of a set of table registers that share the same value of a certain identifier. Another algorithm, called ConnectionBlockQ [30], is capable of doing mining association rules in tables having quantitative attributes, by means of transforming these data into classes or range of values.

Another algorithm is the AprioriMR [31], an extension of the Apriori, for the mining of multi-relational association rules, adjusting data structures and processing stages to be used in relational databases. The GFP-Growth [32] algorithm does the extraction of multi-relational patterns based on the functionalities of the FP-growth and on the clustering concept.

Multi-relational algorithms do not present a satisfactory performance when used in large databases because the generated data structures may be too large to be allocable in memory, which would make the use of these algorithms impractical. In this scenario, the main differential between the MR-Radix algorithm and other multi-relational algorithms for mining association rules is the ability to perform efficient mining of multiple relationships in large databases in a satisfactory execution time.

The developed work-MR-Radix algorithm

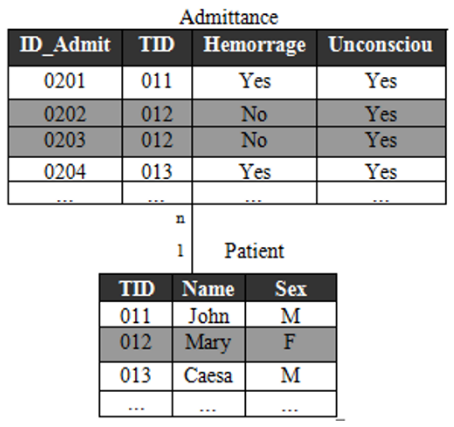
The MR-Radix algorithm proposed in this work is based on the PatriciaMine algorithm which can extract multi-relational association rules from large relational databases. The algorithm uses the Patricia-trie data structure which has a better algorithm performance and reduces the amount of used memory space by up to 75%, when compared to a FP-tree [14].

The proposed algorithm extracts frequent itemsets by means of the top-down strategy, making the processing be done in a global structure, which improves the performance of the algorithm as it is not necessary to construct intermediate or temporary structures. Moreover, the MR-Radix, same as the PatriciaMine, goes through the tree in an iterative manner, which also improves the performance compared to recursive algorithms.

Differential MR-Radix

To enable multi-relational data mining, the MR-Radix presents some modifications to the PatriciaMine. The first change is in relation to generating itemsets, in that the performance of traditional data mining algorithms, that is trivial, becomes complex in a multi-relational approach, since it must contemplate a larger amount of information due to the multiplicity of sources data. With the objective of bypassing this problem, the MR-Radix proposes a new model of representation of itemsets, called “relational itemset”. Another modification of the algorithm was to adapt the search for patterns to a multi-relational context. This work therefore proposed a new strategy called Fusion of Secondary Items (FSI), capable of relating multi-relational patterns so as to avoid the necessity of doing onerous junction operations.

The FSI technique permits relating items from different tables. The operation consists of gathering all the frequent items from the tables involved in the mining process that have the same Transaction Identifier (TID) [25]. Figure 7 shows a database, as an example, where the primary table is the Patient table and the secondary one is the Admittance. Highlighted lines have the same TID so, therefore, correspond to the same patient. During the construction of the tree, the items of the admittance are



Admittance

ID_Admit	TID	Hemorrage	Unconsciou
0201	011	Yes	Yes
0202	012	No	Yes
0203	012	No	Yes
0204	013	Yes	Yes
...	...	...	...

Patient

TID	Name	Sex
011	John	M
012	Mary	F
013	Caesa	M
...	...	...

Figure 7 **Sample of fusion of secondary items**.

aggregated or fused to the frequent items of their respective patient, and are used to construct the Radix-tree.

With the purpose of enabling multiple relationships mining in large databases, an auxiliary structure called ItemMap was proposed which allows the manipulation of data from multiple relational databases through the mapping of relational items in identifiers, which occupy less memory space, optimize the storage and manipulation of nodes in Radix-tree. In addition, all operations are performed for patterns in the same Radix-tree structure, which contributes to the good performance of MR-Radix.

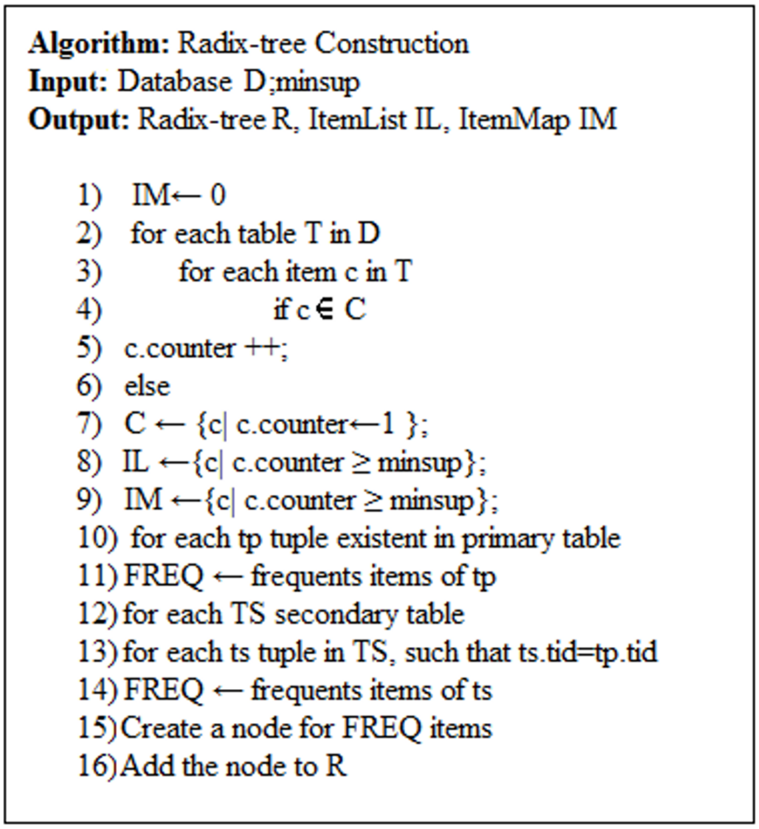
To ensure the processing of large databases, the proposed algorithm uses the strategy of dividing the database into partitions, so that these can be allocated in memory and processed, generating local patterns of these units.

MR-Radix algorithm

In order to provide a better understanding of the MR-Radix, Figure 8 presents the pseudo-code for the Radix-tree, Figure 9 shows the pseudo-code of the stage of mining association rules, and in Figure 10 the pseudo-code of the partitioning step of the MR-Radix is presented.

Comparative study between PatriciaMine and MR-Radix

The PatriciaMine mining association rules algorithm was used for comparative purposes as a representative of a traditional approach and the MR-Radix algorithm as a representative of the multi-relational approach. Tests were done on two basis of real data. The first is called *Sistema de Informação e Vigilância de Acidentes do Trabalho*-SIVAT (Work Accidents Vigilance System) which catalogues all work accident occurrences in a city in São José do Rio Preto, interior of the São Paulo State and was collected and registered by the organ which is responsible for the follow-up of that type of accident. The second database is denominated HC-*Hospital do Cancer* (Cancer Hospital), responsible for storing information relating to a tumour bank of a hospital in the São Paulo State. The HC data repository contemplates information related to the patients and their cancer samples as well as more detailed clinical information. Table 1 shows descriptive information-tables or relations, number of registers and attributes-from the databases used in the tests.



Algorithm: Radix-tree Construction
Input: Database D;minsup
Output: Radix-tree R, ItemList IL, ItemMap IM

```
1)  IM← 0
2)  for each table T in D
3)      for each item c in T
4)          if c ∈ C
5)  c.counter ++;
6)  else
7)  C ← {c| c.counter←1 };
8)  IL ←{c| c.counter ≥ minsup};
9)  IM ←{c| c.counter ≥ minsup};
10) for each tp tuple existent in primary table
11) FREQ ← frequents items of tp
12) for each TS secondary table
13) for each ts tuple in TS, such that ts.tid=tp.tid
14) FREQ ← frequents items of ts
15) Create a node for FREQ items
16) Add the node to R
```

Figure 8 Pseudo-code for the construction stage of the Radix-tree.

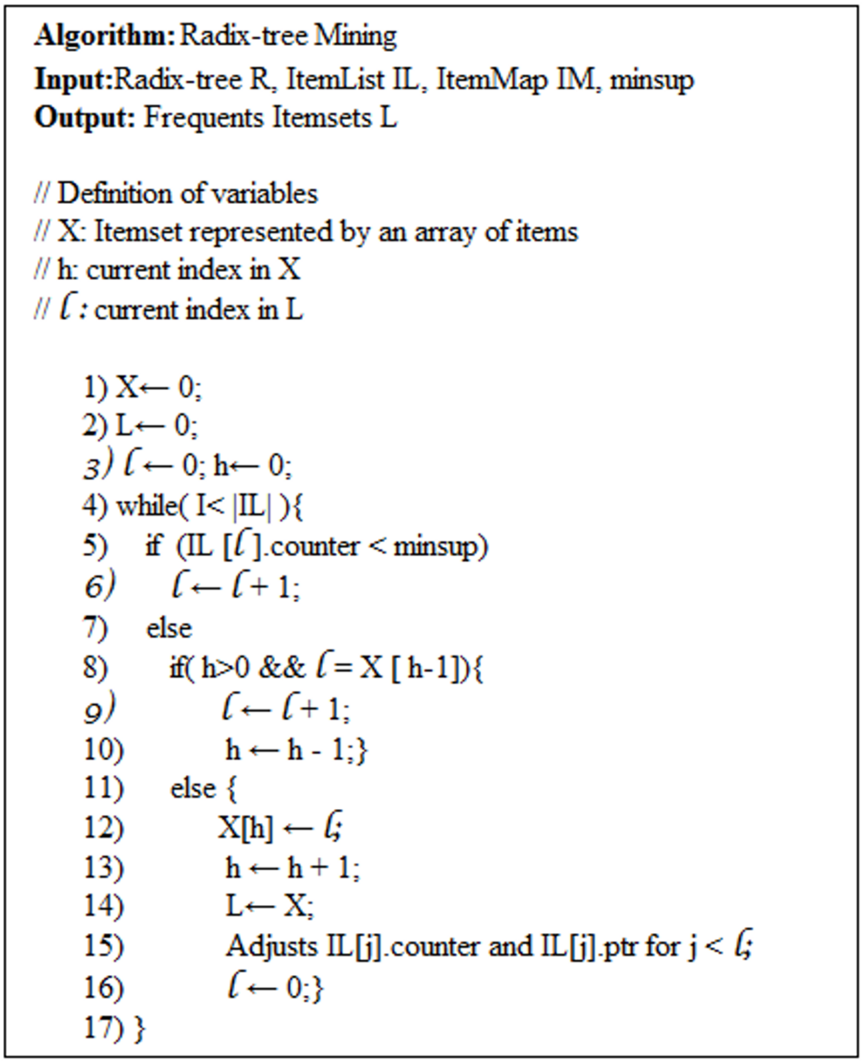
The equipment used to host the algorithms and do the tests was a microcomputer having an Intel Core 2 Quad Q8200 (2.33 Ghz) processor, a DDR 2.4 gigabyte principal memory on a 250 gigabyte hard disc, SATA standard, transfer rate of 3.0 Gbit/s and 7,200 rpm (rotations per minute). The equipment had Microsoft Windows XP Professional SP3 system software, besides a Java J2SE 1.6 Developing Kit from the Netbeans 6.7.1 Developing Environment and the Database Management System MySQL 5.1.

It is worth noting that, different to MR-Radix that can be directly applied to multiple tables, the PatriciaMine traditional algorithm needs a pre-processing stage to join these tables. The comparative study then took into account the computational cost of the joining operation when using the PatriciaMine traditional algorithm. Figure 11 schematically shows the criteria adopted for this comparative test.

To do the joining operation of multiple tables, the PatriciaMine algorithm uses a type of junction called 'full outer join' to list the data from all the involved tables. The resulting list of that junction contains all the registers of the tables participating in the operation, with all not-coinciding values being referenced with a 'null'.

Test parameters

The step of generating association rules is itself independent of the algorithm used so, therefore, it is not appropriate to compare the performance of PatriciaMine and



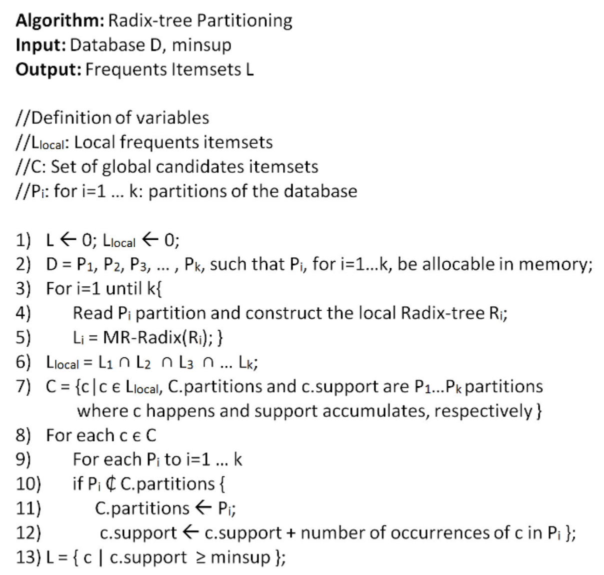
```
Algorithm: Radix-tree Mining
Input: Radix-tree R, ItemList IL, ItemMap IM, minsup
Output: Frequents Itemsets L

// Definition of variables
// X: Itemset represented by an array of items
// h: current index in X
// ℓ: current index in L

    1) $X \leftarrow 0$;
    2) $L \leftarrow 0$;
    3) $\ell \leftarrow 0$; $h \leftarrow 0$;
    4) while( I< |IL| ){
    5)    if (IL [ℓ].counter < minsup)
    6)       $\ell \leftarrow \ell + 1$;
    7)    else
    8)       if( h>0 && ℓ = X [ h-1]){
    9)             $\ell \leftarrow \ell + 1$;
    10)            $h \leftarrow h - 1$;}
    11)      else {
    12)            $X[h] \leftarrow \ell$;
    13)            $h \leftarrow h + 1$;
    14)            $L \leftarrow X$;
    15)            Adjusts IL[j].counter and IL[j].ptr for $j < \ell$;
    16)            $\ell \leftarrow 0$;}
    17) }
```

Figure 9 Pseudo-code for the mining phase of Radix-tree.

MR-Radix. Thus, the algorithms were evaluated for the task of obtaining all frequent itemsets, carrying out successive executions for different values of minimum support, in order to verify the behaviour of the proposals against the variation of this measure of interest.

The structure of Radix-tree compresses the itemsets with the same counter support in a single node. Thus, to compare the performance of PatriciaMine and MR-Radix, the number of nodes generated in the structures Radix-tree of each algorithm is more interesting than the number of itemsets, since the same node can contemplate many itemsets. For this reason, the number of nodes generated in Radix-tree, instead of the number of itemsets, is presented in the tests.

It is noteworthy that each test set had its metrics collected three times, and for the construction of graphs and tables were averaged between the results of three applications for each test case, which reduces the uncertainties generated by possible



```
Algorithm: Radix-tree Partitioning
Input: Database D, minsup
Output: Frequents Itemsets L

//Definition of variables
//$L_{local}$: Local frequents itemsets
//C: Set of global candidates itemsets
//$P_i$: for i=1 … k: partitions of the database

1)  L ← 0; $L_{local}$ ← 0;
2)  D = $P_1, P_2, P_3, … , P_k$, such that $P_i$, for i=1…k, be allocable in memory;
3)  For i=1 until k{
4)      Read $P_i$ partition and construct the local Radix-tree $R_i$;
5)      $L_i$ = MR-Radix($R_i$); }
6)  $L_{local} = L_1 \cap L_2 \cap L_3 \cap … L_k$;
7)  C = {c|c ∈ $L_{local}$, C.partitions and c.support are $P_1…P_k$ partitions
        where c happens and support accumulates, respectively }
8)  For each c ∈ C
9)      For each $P_i$ to i=1 … k
10)     if $P_i$ ⊄ C.partitions {
11)         C.partitions ← $P_i$;
12)          c.support ← c.support + number of occurrences of c in $P_i$ };
13) L = { c | c.support ≥ minsup };
```

Figure 10 Pseudo-code for the partitioning strategy.

variations during the collection. Therefore, it is possible to ensure greater confidence of the results presented in graphs and other resources.

Execution time comparison

The first analysis is about the execution time of the MR-Radix and PatriciaMine algorithms to judge the performance of mining association rules tasks in relational databases. Respective times of execution were measured during the application of the algorithms. Figure 12 shows execution time values of the algorithms application on SIVAT database and Figure 13 shows the same application on HC database.

The SIVAT database has a reasonable volume of data. The data structure used by the algorithm is robust, needing more time to conclude the processing. Even so, results at first show that the MR-Radix algorithm, despite handling complex multi-relational patterns, had a greater efficiency than its corresponding traditional, PatriciaMine, when applied to the SIVAT repository. This indicates that multi-relational mining was shown to be the most advantageous way to prospect for data as the proposed algorithm was two times faster than the PatriciaMine, as can be seen in Figure 12.

The graphic in Figure 13 shows that the performance provided by the algorithm MR-Radix was approximately six times faster than PatriciaMine. Interestingly, the MR-

Table 1 Description of the SIVAT and HC database

Database	Reports	Quantity of registers	Quantity of attributes
SIVAT	Form	32,336	38
	Form_cid_10	64,873	3
	Form_part of body	35,624	2
	Type of machine	859	3
	Occupation	1,237	3
HC	Sample	13.100	31
	Urology_Kidney	60	60
	Patient	3.647	18

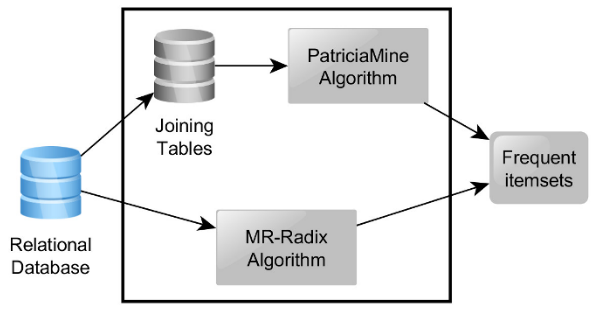


Figure 11 **Scheme of a comparative study between MR-Radix and PatriciaMine.**

Radix presents no substantial variations in the execution time, even when using smaller support values which tends to generate a larger number of items to be processed. In contrast, the PatriciaMine increases the execution time as it reduces the amount of support, indicating that the algorithm is more sensitive to the increase in the number of items to be analyzed in mining. This feature is justified by the fact that the traditional approach is applicable to only one table.

Moreover, the MR-Radix algorithm performed better than the PatriciaMine in relation to execution time, even though the time of joining tables was disregarded. An example is observed for the support value equal to 0.5% in the PatriciaMine held mining step, disregarding the time of the junction of 2.30 seconds, while the MR-Radix performed this task in only 1 second.

Utilised memory comparison

The objective of the study of utilised memory was to investigate whether a manipulation of multi-relational patterns caused a need for a bigger quantity of this resource. Figure 14 shows a graphic comparison in relation to utilised memory in the application of the algorithms on the SIVAT base and Figure 15 shows that application on BT database.

On analyzing the graph of Figure 14, note that utilized memory was similar with both algorithms for most of the support values. Nevertheless, for the lower support values, is this case 1%, the Patricia Mine needed a larger quantity of memory. This behaviour is related to the number of nodes generated by the respective algorithms in the different support configurations, as can be seen in Table 2.

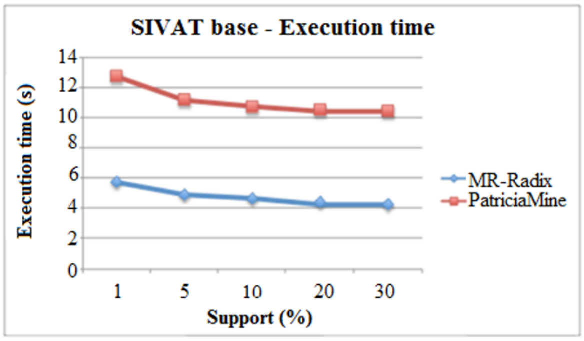


Figure 12 **Execution time graphic-MR-Radix and PatriciaMine in SIVAT base.**

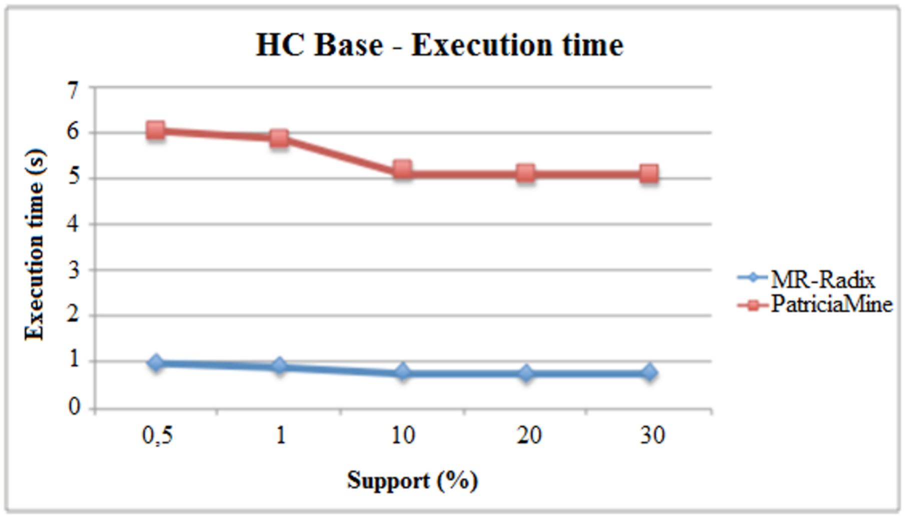


Figure 13 Execution time graphic-MR-Radix and PatriciaMine in HC base.

According to Table 2, for a support value equal to 1%, the PatriciaMine generated 18,435 more nodes than MR-Radix. This difference is explained by the necessity of having to join tables which causes the generation of a higher number of frequent items and, consequently, a higher number of nodes in the Patricia-trie. On the other hand, for higher support values the PatriciaMine algorithm shows less memory use than MR-Radix. This results from the fact that the traditional algorithm theoretically handles more simple patterns which come from a single table.

This characteristic should be related to the results relative to the execution time of the algorithms. Even though the MR-Radix uses a slightly higher use of memory with certain support values, due to the manipulation of multi-relational patterns, its execution time remains substantially lower when compared to PatriciaMine.

Still analysing the utilised memory graphic in Figure 14, it can be seen that the algorithm PatriciaMine has a more vertical growth curve, indicating that the need of memory space intensifies faster as support values reduce. On the other hand, the MR-Radix algorithm has a more conservative curve, which shows an interesting characteristic,

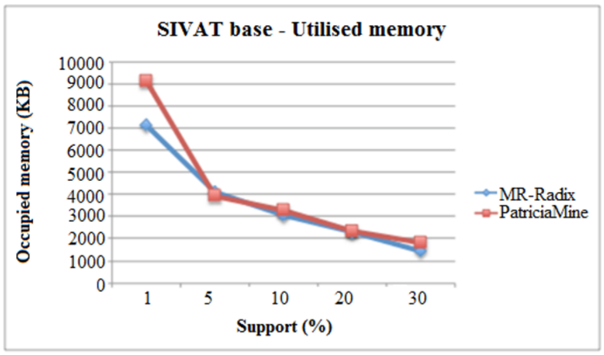


Figure 14 Utilised memory graphic-MR-Radix and PatriciaMine in SIVAT base.

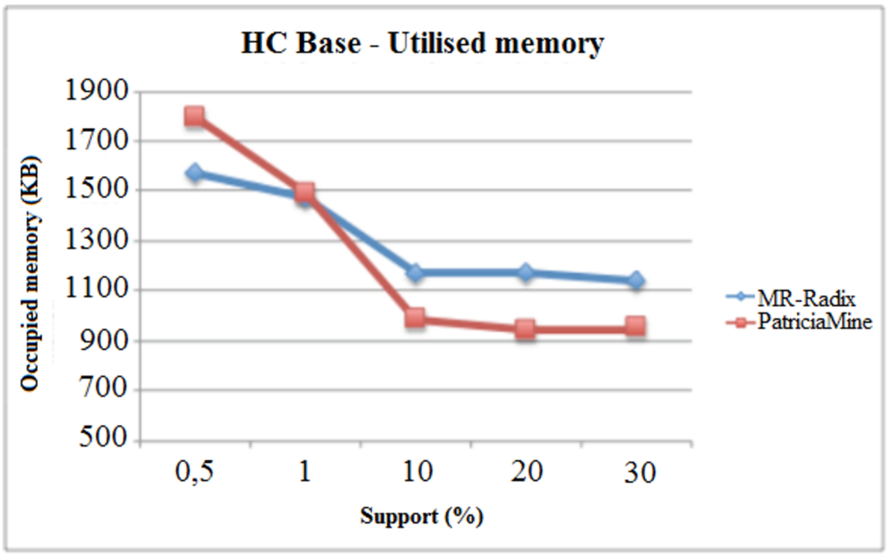


Figure 15 Utilised memory graphic-MR-Radix and PatriciaMine in HC base.

because the increase in demand of frequent items does not result in a significant growth of utilized memory.

In Figure 15 it appears that the MR-Radix algorithm requires more memory than the PatriciaMine to support higher values. However, for lower values of support-in this case, less than 1%-MR-Radix algorithm begins to show a lower use of main memory. This is related to the number of nodes generated in the data structure of the MR-Radix during the mining process. As can be seen in Table 3, the number of nodes generated by the MR-Radix algorithm is greater than the number generated by PatriciaMine to support higher values up to 10%, while for smaller values the traditional approach generates a larger number of nodes, since joining multiple tables generated a larger data set.

With the tests it was possible to confirm the effectiveness and efficiency of a multi-relational mining algorithm in relational databases, since the execution time and memory consumed by the MR-Radix, in most cases, showed better rates compared to the traditional algorithm that has the limitation of needing to execute junctions of data from multiple tables.

Conclusions

The comparative study confirmed the efficacy and efficiency of the multi-relational MR-Radix algorithm for use in relational databases, in terms of execution time and utilized memory. This algorithm was analysed with comparisons to the traditional PatriciaMine algorithm which showed that multi-relational mining has, in fact, a better

Table 2 Number of nodes in the SIVAT base tree

Algorithms	Support				
	1%	**5%**	**10%**	**20%**	**30%**
MR-Radix	40426	24272	16266	10473	5580
PatriciaMine	58861	21670	14545	9287	5865

Table 3 Number of nodes in the HC base tree

Algorithms	Support				
	0.5%	1%	10%	20%	30%
MR-Radix	2999	2586	500	281	216
PatriciaMine	5132	3754	425	258	137

efficiency as it avoids costly multiple table joining operations. Moreover, the multi-relational approach does not show semantic losses, not like the traditional algorithms that can induce errors or inaccuracies during the joining of tables.

Thereby, this work presents, as an original contribution, the MR-Radix algorithm which has a performance of confirmed efficacy of the multi-relational approach in data mining process.

Acknowledgements
This project was financed by CAPES. We thank David R. M. Mercer for English language review and translation.

Author details
[1]São Paulo State University - Unesp, Rua Cristóvão Colombo, 2265 São José do Rio Preto, São Paulo, Brazil [2]Federal University of São Carlos - UFSCar, Rodovia Washington Luís, Km 235, Caixa Postal 676 São Carlos, SP, Brazil [3]University of São Paulo - USP, Avenida Prof. Luciano Gualberto, 380, travessa 3, São Paulo, São Paulo, Brazil

Authors' contributions
CRV and FTO developed the algorithm and drafted the paper. PSN, ACC, AMC, RCGS and PLPC provided the needed support to the work and to the results and reviewed the paper. All authors read and approved the final manuscript.

Competing interests
The authors declare that they have no competing interests.

Received: 24 November 2011 Accepted: 7 March 2012 Published: 7 March 2012

References

1. Kantardzic M (2003) Data Mining: Concepts, Models, Methods and Algorithms. New Jersey: Wiley
2. Knobbe AJ (2004) Multi-Relational Data Mining. Thesis (Ph.D.). The Netherlands p 130
3. Knobbe AJ, Blockeel H, Siebes A (1999) Van Der Wallen DMG (1999) Multi-relational Data Mining. Benelearn: Proc
4. Dzeroski S, Raedt LD, Wrobel S (2003) Multi-Relational Data Mining: Workshop Report. ACM SIGKDD 2003. Explorations Newsletter 5(2):200–202. doi:10.1145/980972.981007.
5. Domingos P (2003) Prospects and challenges for multi-relational data mining. ACM SIGKDD Explorations Newsletter 5(1)
6. Page D, Craven M (2003) Biological applications of multi-relational data mining. ACM SIGKDD Exploration Newsletter 5(1):69–79. doi:10.1145/959242.959250.
7. Habrard A, Bernard M, Jacquenet F (2003) Multi-relational Data Mining in medical databases. Lecture notes in computer science. Springer 2780:365–374
8. Blockeel H, Dzeroski S (2005) Multi-Relational Data Mining. Workshop Report. ACM SIGKDD Explorations Newsletter 7(2):126–128. doi:10.1145/1117454.1117471.
9. Fayyad U, Piatetsky-Shapiro G, Padhraic S., *et al* (1996) From Data Mining to Knowledge Discovery: An Overview. In: FAYYAD UM, et al (ed) Advances in Knowledge Discovery and Data Mining, 0th edn. Menlo Park: AAAI Press, The MIT Press pp 1–34
10. Agrawal R, Imielinski T, Swami A (1993) Mining Association Rules between Sets of Items in Large Databases. Proc. 1993 ACM SIGMOD International Conference on Management of Data 207–216
11. Wang K, Tang L, Han L, Liu J (2002) Top Down FP-Growth for Association Rule Mining. Lecture notes in computer science. Springer 2002 2336:334–340
12. Pei J, Han J, Lu H, Nishio S, Tang S, Yang D (2001) H-mine: hyper-structure mining of frequent patterns in large databases. Proc. IEEE International Conference on Data Mining. IEEE Computer Society 441–448
13. Liu J, Pan Y, Wang K, Han J (2002) Mining frequent item sets by opportunistic projection. New York, USA: ACM Press p 229. Proc. Eighth ACM SIGKDD international conference on Knowledge discovery and data mining-KDD?'?02
14. Pietracaprina A, Zandolin D (2003) Mining frequent itemsets using patricia tries. Proc. IEEE ICDM Workshop Frequent Itemset Mining Implementations, vol. 90. Melbourne: CEUR-WS.org
15. Gopalan R (2004) Sucahyo Y (2004) High performance frequent patterns extraction using compressed FP-tree. SIAM International Workshop on High Performance and Distributed Mining, Orlando, USA: Proc
16. Shang X (2005) SQL Based Frequent Pattern Mining. Thesis (Ph.D.) 146
17. Frequent Itemset Mining Implementations Repository. Proc. IEEE Workshop on Frequent Itemset Mining Implementations (FIMI?'?04). Available at ?<? fimi.cs.helsinki.fi/>. Accessed on 8th May, 2011
18. Grahne G, Zhu J (2003) Efficiently using prefix-trees in mining frequent itemsets. Proc. IEEE ICDM Workshop Frequent Itemset Mining Implementations, vol. 90. Melbourne: CEUR-WS.org

19. Tsechansky MS, Pliskin N, Rabinowitz G, Porath A (1999) "Mining Relational Patterns from Multiple Relational Tables". Decision Support Systems 27(1-2):177–195. doi:10.1016/S0167-9236(99)00043-3.
20. Dehaspe L, De Raedt L (1997) Mining association rules in multiple relations. Proc. 7th Intl. Workshop on Inductive Logic Programming. Prague, Czech Republic pp 125–132
21. Nijssen S, Kok J (2001) Faster association rules for multiple relations. International Joint Conference on Artificial Intelligence 17:891–896. Citeseer
22. Clare A, Williams HE, Lester N (2004) Scalable multi-relational association mining. Proc. 4th IEEE International Conference on Data Mining (ICDM'04) 355–358
23. Ketkar NS, Holder LB, Cook DJ (2005) Comparison of graph-based and logic-based multi-relational Data Mining. ACM SIGKDD Explorations Newsletter 7(2):64–71. doi:10.1145/1117454.1117463.
24. Inokuchi A, Washio T, Motoda H (2000) An apriori-based algorithm for mining frequent substructures from graph data. Lecture notes in computer science. Springer 1910:13–23
25. Kuramochi M, Karypis G (2004) An efficient algorithm for discovering frequent subgraphs. IEEE Transactions On Knowledge and Data Engineering 16(9):1038–1051. doi:10.1109/TKDE.2004.33.
26. Matsuda T, Horiuchi T, Motoda H, Washio T (2000) Extension of graph-based induction for general graph structured data. Lecture notes in computer science. Springer 1805:420–431
27. Teredesai A, Ahmad M, Kanodia J, Gaborski R (2005) CoMMA: a framework for integrated multimedia mining using multi-relational associations. Knowl Inf Syst 10(2):135–162
28. Ribeiro MX, Vieira MTP, Traina AJM (2005) Mining Association Rules Using Clustering. I workshop on algorithms for data mining. Uberlândia, Brazil pp 9–16. [in Portuguese]
29. Pizzi L, Ribeiro M, Vieira M (2005) Analysis of Hepatitis Dataset using Multirelational Association Rules. Proc. ECML/PKDD Discovery Challenge
30. Garcia E (2008) Mining association rules from multi-relational quantities. Thesis (Masters). Methodist University of Piracicaba p 84. [in Portuguese]
31. Oyama FT (2006) Extraction of knowledge in databases using multi-relational clustering tuples. Monograph (Undergraduate) São Paulo State University p 51 [in Portuguese]
32. Pizzi LC (2006) Data mining in multiple tables: GFP-Growth algorithm. Thesis (Masters). Federal University of São Carlos p 106. [in Portuguese]

doi:10.1186/2192-1962-2-4
Cite this article as: Valêncio *et al.*: MR-Radix: a multi-relational data mining algorithm. *Human-centric Computing and Information Sciences* 2012 **2**:4.