IARMMD: A Novel System for Incremental Association Rules Mining from Medical Documents

Hany Mahgoub
Faculty of Computers and Information, Menoufia University, Egypt

ABSTRACT

This paper presents a novel system for Incremental Association Rules Mining from Medical Documents (IARMMD). The system concerns with maintenance of the discovered association rules and avoids redoing the mining process on whole documents during the updating process. The design of the system is based on concepts representation. It designed to develop our previous D-EART system. The IARMMD improves the updating process, and will lead to decrease the number of scanning and the execution time. The system consists of three phases that are Text Preprocessing, Incremental Association Rule Mining, and Visualization phase. Hash-based Incremental Association Rule Mining Algorithm (HIARM) is introduced in the mining phase. The algorithm employs the power of data structure called Hash Table. The performance of the algorithm is compared with both Apriori and FUP algorithms for the execution time and the evaluation of the extracted association rules. The results reveal that the number of extracted association rules in the IARMMD system is always less than that in Apriori-based and FUP-based systems. Furthermore, the execution time of HIARM algorithm is much better than Apriori and FUP algorithms in the updating process in all experimental cases.

General Terms

Knowledge Discovery in Text

Keywords

Knowledge Engineering, Text mining, Data mining, Knowledge Mining, Incremental Association Rules Mining

1. INTRODUCTION

Rule generation is an important technique for data mining. There have been many studies on efficient discovering of association rules from large datasets [1]. Association Rules Mining (ARM) process is not at all free but it is costly specially frequently updating is very costly. It is nontrivial to maintain such discovered rules from large datasets because a dataset may allow frequent or occasional updates such as insertion, deletion and modification of transactions in the dataset. Updating process may not only invalidate some existing strong association rules but also turn some weak rules into strong ones. Incremental Association Rules Mining (IARM) concepts concerned with maintenance of the discovered association rules and avoid redoing the mining process on a whole dataset. A method for handling incremental dataset updates for the rules discovered by the generalization-based approach was briefly discussed by Han, Cai, and Cercone in 1993 [2]. The problem of maintaining association rules is first studied by Cheung and et al in 1996 [3]. There are several important characteristics in the updating problem such as:

1. The updating problem can lead to find a new set of large frequent itemsets and after that, the new association rules can be computed from the new large frequent itemsets.
2. The old large frequent itemsets has the potential to become small in the updated dataset.
3. the old small itemset could become large in the new dataset
4. In order to find a new large frequent itemsets, all the transactions in the updated dataset, including those from the original dataset, have to be checked against every candidate set [3].

The former direction to achieve efficient update in ARM process is adding the new dataset to the original dataset and re-scanning the whole dataset by applying Apriori algorithm [4] using the same user thresholds (minimum support *s* and minimum confidence *c*). This approach, though simple, has some obvious disadvantages, such as all the computations done initially at finding out the old large frequent itemsets are wasted and all large frequent itemsets have to be computed again from the scratch [3]. ARM captures all possible rules that explain the presence of some attribute that lead to the presence of another attribute. The traditional approaches for discovering association rules mainly depend on existing of the original dataset, which may not be suitable for fast mining when the dataset becomes very large and frequently updated. It is well known that the critical effected factors of mining process are the execution time and the number of scans on the dataset. The most effected factor in ARM proposed by Agrawal was the number of dataset scanning [4].

Since are many studies on efficient discovering of association rules from large dataset. Also, there are many directions to maintain discovered association rules after some update activities. In the incremental mining process, most of the proposed techniques used the old dataset besides the new dataset concurrently. Without using old dataset, the proposed techniques are not beneficiary and hard to be used. From different point of view, the existing of the original dataset requires more numbers of scanning on the dataset and increases the time used for execution as well as storage space. So the existing of the old dataset may not be suitable for fast mining or interactive mining when dataset becomes very large and frequently updated. The scenario for most proposed techniques is scanning old dataset (*OD*) and adding the new dataset (*ND*) then creates total dataset (TD). Then apply Apriori technique, which scan *TD* to extract the association rules. In general, there are many directions to maintain the discovered association rules after some update activities. The first direction was merging the *OD* and the *ND* then applying the Apriori on the *TD*. The updating process includes adding new transactions or deleting existing transactions from the *OD*. This direction is suitable when the *ND* is small compared to the *OD*. While if the updated dataset (*ND*) is very large compared to the *OD*, then this direction becomes very hard.

Since, after adding the *ND* to the *OD*, the *TD* becomes very large. Also, applying Apriori technique leads to multiple scanning on *OD* and *TD*. The second direction idea is creating an intermediate file between *OD* and *ND*. This intermediate file represents both accepted and rejected itemset. The main problem associated with this direction is changing the user thresholds (*s* and *c*) can create more files, which requires more storage space and time. The second major problem for this direction is some itemsets will not appear in accepted or rejected files resulted from using *OD* but may appear by adding a *ND*. The above IARM concept is introduced from the point of view of data mining field and I found that there is no any research in the text mining dealing with this problem. In other words, the previous text mining approaches do not take into account the updating problem. In addition, the problem is the existing of huge amount of textual available information in textual form in online sources and these sources are dynamically changed. So I think that we always need to the updating process.

This paper presents a new text mining system for Incremental Association Rules Mining from Medical Documents. The system is designed to develop our previous D-EART system which is presented in [5]. It is consists of three phases that are Text Preprocessing, Incremental Association Rule Mining, and Visualization phase. It is similar to our previous D-EART system in its preprocessing phase and the difference appears in the Incremental Association Rules Mining phase. The system concerns with maintenance of the discovered association rules and avoids redoing the mining process on whole documents during the updating process. A hash-based Incremental Association Rules Mining algorithm is presented in the mining process. The algorithm employs the power of data structure called Hash Table. The new algorithm improves the updating process and leads to decrease the number of scanning and the execution time. Moreover, it permits the end user to extract new rules with different user threshold values with no need to re-scan the documents. In the IARMMD system, MEDLINE abstracts are selected for the breast cancer treatments and side effects as the main domain of online collecting documents [6].

The reset of this paper is organized as follows. Section 2 presents the literature review. Section 3 presents the Incremental Association Rules Mining from Medical Documents system architecture. Experimental results are presented in section 4. Section 5 provides conclusions and future work.

2. LITERATURE REVIEW

The prominent algorithm for association rule mining is Apriori algorithm [4]. Apriori computes frequent itemsets in a large database through several iterations based on a prior knowledge. Each iteration, it generates a number of candidate frequent itemsets and then verify them by scanning the database. For a frequent itemset, its support must be higher than a user-specified minimum support threshold. The association rule can be discovered based on frequent itemsets. The disadvantages of this algorithm is that large number of candidate generation and time consuming as it required multiple passes for processing.

DHP [7] algorithm is an extension of Apriori by further reducing the number of candidate itemsets using a hashing technique. DHP is very efficient for the generation of candidate large itemsets, in particular for the large 2-itemsets. In addition, DHP proposed the effective pruning techniques to progressively reduce the number of transaction database and the number of items in each transaction. While both Apriori and DHP efficiently discover association rules from a database, the rules maintenance problem is not addressed. For dynamic databases, several incremental updating techniques have been developed for mining association rules. One of the previous works for incremental association rule mining is FUP algorithm that was presented by Cheung et al [3]. FUP algorithm is the first incremental updating technique for maintaining association rules when new data are inserted into database. Based on the concepts of Apriori algorithm, FUP computes frequent itemsets using large itemsets found at previous iteration. The major idea of FUP is re-using frequent itemsets of previous mining to update with frequent itemsets of an incremental database. At each iteration, the supports of the size-k frequent item sets of an original database are updated by scanning an incremental database. As well as any k-frequent itemset of the incremental database are updated by scanning the original database to find the new frequent itemsets. As a result, FUP algorithm requires scanning passes over an original database several times when new frequent itemsets are found. This can degrade the performance of FUP algorithm. They have extended their work to FUP and FUP2 [8, 9] to *k*-pass algorithms; they scan the database k^{th} time.

Lee et al [10] proposed an algorithm called Different Estimation for Large Itemsets (DELI), which assumed that after some update activities, old transactions are deleted from dataset OD and new transactions are added. The experiments showed that DELI is three times faster than FUP2. Chang et al [11] proposed a new algorithm, called the New Fast UPdate algorithm (NFUP) for efficiently incrementally mining association rules from large transaction database. NFUP does not require the rescanning of the original database and can determine new frequent itemsets at the latest time intervals. The running time of NFUP rises almost in direct proportion with the transaction number of the incremental database. Accordingly, NFUP is suited frequently updated databases. To deal with the rescanning problem, negative border approach is presented by Toivonen [12], Thomas et al [13] and Feldman et al [14]. This approach maintains both frequent itemsets and border itemsets. Theses border-based algorithms need large memory space to keep the border itemsets.

To reduce memory space, Hong el al. [15] and Amornchewin and Kreesuradej [16] propose a new approach. The approach maintains both frequent itemsets and expected frequent itemsets. In order to guarantee that all frequent itemsets can be found when a new database is added to an original database, the approach can only allow very small size of an incremental database to insert into an original database. To deal with the problem that the previous approach can only allow very small size of an increment database to insert into an original database, Probability-based Incremental Association Rule Discovery Algorithm, is introduced by Amornchewin and Kreesuradej [17, 18]. Similar to the previous approach, Amornchewin [19] developed an algorithm that is an improvement of Probability-based algorithm with inclusion of the direct hashing technique. The approach applied the hashing technique for efficient the 2 frequent itemset generation.

FP-tree [20, 21, 22] is one of best approach to discover frequent pattern to overcome the drawback of the Apriori algorithm. FP-tree as better performance than Apriori as reduce database scan. Since even if small insertion is done, restructuring of item is required again to arrange in descending order. FP-growth [22, 23] algorithm is applied on FP-tree to discover frequent pattern. It is based on divide conquer approach to discover frequent pattern of various sizes. Liu et al [24] present an algorithm called FUFP-tree based incremental association rule mining algorithm (Pre-FP). It is based FUFP [25, 26] (Fast Updated Frequent Pattern) concept. The major idea of FUFP is re-use of previously mine frequent items to update with incremental database. It reduces number of candidate set in updating process. Nath et al [27] presented an efficient incremental association mining technique which can discover non-redundant, reduced set of interesting rules from continuous valued dataset, without converting into the market-basket domain. Unlike the other traditional association mining techniques, it can extract the rules from the real-life dataset directly in a single phase.

From all the previous works, we noticed that the problem of IARM is introduced only from the point of view of data mining field. Until now, there is no any incremental mining research taken from the point of view of text mining. Therefore, this work focuses on the incremental mining problem from the point of view of text mining.

3. Incremental Association Rules Mining from Medical Documents (IARMMD) System Architecture

The IARMMD system consists of three main phases after collecting the documents online as shown in Figure 1. The main phases are Text Preprocessing Phase that includes transformation, filtration, stemming, synonymy and indexing of documents, Incremental Association Rule Mining Phase that includes the new algorithm HIARM, and Visualization Phase.

3.1 Online Documents Collection

The IARMMD system works online, so it is considered to be as a web-based text mining system. It accepts the documents that in XML format (structured) and also the unstructured documents. From the interface of the system, the user can online access the PubMed website and writes the search keywords. The selected documents and their tags are automatically loading into the system and the user can select the specific part of documents that will work on it.

3.2 Text Preprocessing Phase

The IARMMD system has the ability to deal with the native XML documents and the unstructured documents. The process of concept extraction is done and the documents are filtered, stemmed and synonyms are used. Finally, the XML documents are automatically indexed by using the fuzzy weighting schema. This phase is the same as in [5].

3.2.1 Transformation

Once the online XML documents download into the system, their tags are automatically extracted in a combobox as shown in Figure 2. The user can determine a specific part of the documents (for example the abstract part, </Abstract Text>) to work on it. Therefore the IARMMD system is flexible to work on specific or all parts of documents. In the case of the unstructured documents, the IARMMD system transforms them to the XML format.

3.2.2 Concept Extraction

The concept is a single word or a group of consecutive words that occurs frequently enough in the entire document collection. It is important to appear the concepts as a unit in the extracted association rules. The process of concept extraction can be done as in [5].

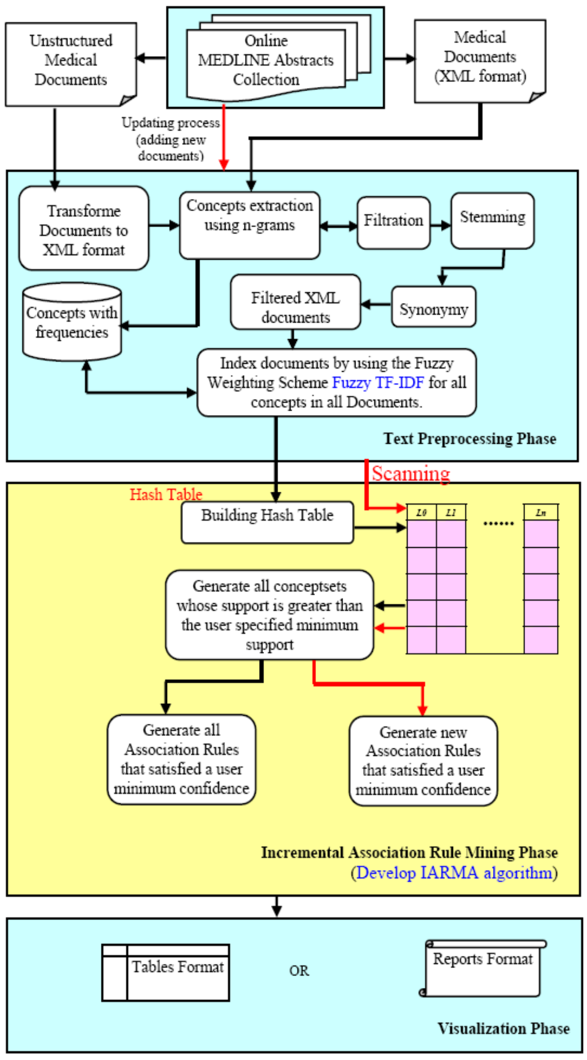
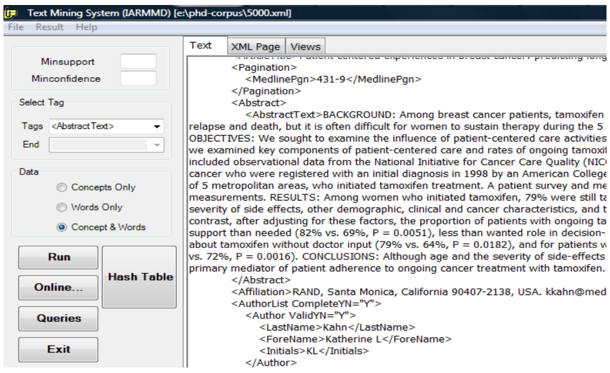

Unstructured Medical Documents
Online MEDLINE Abstracts Collection
Medical Documents (XML format)
Updating process (adding new documents)
Transform Documents to XML format
Concepts extraction using n-grams
Filtration
Stemming
Concepts with frequencies
Filtered XML documents
Synonymy
Index documents by using the Fuzzy Weighting Scheme Fuzzy TF-IDF for all concepts in all Documents.
Text Preprocessing Phase
Scanning
Hash Table
Building Hash Table
L0
L1
......
Ln
Generate all conceptsets whose support is greater than the user specified minimum support
Generate all Association Rules that satisfied a user minimum confidence
Generate new Association Rules that satisfied a user minimum confidence
Incremental Association Rule Mining Phase
(Develop IARMA algorithm)
Tables Format
OR
Reports Format
Visualization Phase

Fig 1: The IARMMD system architecture



Minsupport
Minconfidence
Select Tag
Tags <AbstractText>
End
Data
( ) Concepts Only
( ) Words Only
[✓ Concept & Words]

```
        <Pagination>
          <MedlinePgn>431-9</MedlinePgn>
        </Pagination>
        <Abstract>
          <AbstractText>BACKGROUND: Among breast cancer patients, tamoxifen
relapse and death, but it is often difficult for women to sustain therapy during the 5
OBJECTIVES: We sought to examine the influence of patient-centered care activities
we examined key components of patient-centered care and rates of ongoing tamoxi
included observational data from the National Initiative for Cancer Care Quality (NIC
cancer who were registered with an initial diagnosis in 1998 by an American College
of 5 metropolitan areas, who initiated tamoxifen treatment. A patient survey and me
measurements. RESULTS: Among women who initiated tamoxifen, 79% were still ta
severity of side effects, other demographic, clinical and cancer characteristics, and t
contrast, after adjusting for these factors, the proportion of patients with ongoing ta
support than needed (82% vs. 69%, P = 0.0051), less than wanted role in decision-
about tamoxifen without doctor input (79% vs. 64%, P = 0.0182), and for patients w
vs. 72%, P = 0.0016). CONCLUSIONS: Although age and the severity of side-effects
primary mediator of patient adherence to ongoing cancer treatment with tamoxifen.
        </Abstract>
        <Affiliation>RAND, Santa Monica, California 90407-2138, USA. kkahn@med
        <AuthorList CompleteYN="Y">
          <Author ValidYN="Y">
            <LastName>Kahn</LastName>
            <ForeName>Katherine L</ForeName>
            <Initials>KL</Initials>
          </Author>
```

Fig. 2: Selecting a specific tag of the documents

3.2.3 Filtration

The documents are filtered by removing the unimportant words from the documents. A list of unimportant words called stopwords is built. The system checks the documents content and eliminates these unimportant words (e.g. articles, pronouns, conjunctions, and common adverbs). Moreover, the system replaces special characters, parentheses, commas, etc., with distance between words and concepts in the documents.

3.2.4 Stemming
*A*fter the filtration process had done, the IARMMD system automatically do word stemming based on the inflectional stemming algorithm. The inflectional stemming algorithm consists of both part of rule-based and dictionary-based [5].

3.2.5 Synonymy
In the synonymy process, the IARMMD system matches each concept in the documents with the augment synonymous list. When the system finds a synonymy for the concept, it replaces the concept in all documents with its synonymy. This process is done as in [5].

3.2.6 Indexing
Mathematical formula of weighting schema in IARMMD system is used from [5], and it named fuzzy weighting schema. This formula overcomes the drawbacks of the standard weighting schema. All weighted concepts are store in XML file for using them as input to the mining process. One of the drawbacks of our previous EART system [28, 29] is that the value of the threshold weight is hard. So we developed the system to automatically compute the weight value for each word and select the actually important concepts without entering the threshold weight value M. We developed the mathematical formula weighting schema and named it fuzzy weighting schema since the threshold weight value is replaced with the fuzzy membership value as shown in equation (1)

$$\mu_{i,j} = \left\{ \frac{Nt_j}{|C|} \quad where \quad 0 \le \mu \le 1 \right. \tag{1}$$

Where Nt_j denotes the number of documents in collection *C* in which t_j occurs at least once (document frequency of the term t_j) and $|C|$ denotes the number of the documents in collection *C*. Therefore, the fuzzy weighting schema is defined as follows:

$$Fuzzy\cdot w(i,j) = \mu_{i,j} * \begin{cases} Nd_i, t_j * \log_2 \frac{|C|}{Nt_j} & if\ Nd_i, t_j \ge 1 \\ 0 & if\ Nd_i, t_j = 0 \end{cases} \tag{2}$$

This formula caused developing in the system since the high weighted values were given to the concepts that are more occurrences in the documents. Moreover, new concepts appeared with high fuzzy weighted values although they are disappeared by using the weighing schema. The IARMMD system automatically eliminates 10% of all concepts that have low weighted values. After that the system stores all concepts without redundancy with their frequencies in XML file for using them as input to the mining process.

Consider the 6-collection of documents as shown in Figure 3. In the indexing process, the fuzzy weighted values are calculated for each concept in the 6 documents. The total number of concepts is equal to 21 concepts in all documents. We summarized all concepts with their two weighted values in Table 1. From Table 1, it noticed that a concept C_4 has zero weighted values so that the system automatically eliminates it from all documents. The system resorts the concepts based on their weighted values from the highest to the lowest. Table 2 shows all resorted concepts with their TF-IDF values. By choosing the threshold weight value *M*=50%, all concepts that in the shaded region had discarded. The system stores all accepted concepts without redundancy which are approximately 4 concepts (C_1, C_2, C_3 and C_6) in XML file. Table 3 shows the same resorted concepts but with their Fuzzy TF-IDF values. The concepts that appear in the shaded region had discarded, since the less important concepts with fewer frequencies always exist in the bottom of the table. After that the system stores all concepts without redundancy with their frequencies which are approximately 4 concepts (C_1, C_2, C_3 and C_5) in XML file for using them as input in the mining process.

Collection of Documents	
DID	**Concepts**
D1	C_1 C_2 C_1C_3 C_6 C_4
D2	C_3 C_4 C_5 C_3 C_5 C_5 C_4
D3	C_2 C_3C_4 C_2 C_3 C_3 C_3 C_5
D4	C_1C_5 C_4 C_1 C_5 C_1 C_5 C_5 C_5
D5	C_3 C_4 C_5 C_3 C_4 C_5 C_3
D6	C_2 C_5 C_4 C_5 C_2 C_5 C_2 C_5

Fig 3: The collection of 6 documents

Table 1. TF-IDF and fuzzy TF-IDF values for each concept in 6 documents

D-ID	**Concept**	**Frequencies**	**No. of documents**	**TF-IDF**	**Fuzzy TF-IDF**
D1	C_1	2	2	0.954	0.318
	C_2	1	3	0.301	0.151
	C_3	1	4	0.176	0.117
	C_6	**1**	**1**	**0.778**	**0.129**
	C_4	**1**	**6**	**0.0**	**0.0**
D2	C_3	2	4	0.352	0.235
	C_4	2	6	**0.0**	**0.0**
	C_5	3	5	0.237	0.197
D3	C_2	2	3	0.602	0.301
	C_3	4	4	0.704	0.469
	C_4	1	6	**0.0**	**0.0**
	C_5	1	5	0.079	0.066
D4	C_1	3	2	1.431	0.477
	C_4	**1**	**6**	**0.0**	**0.0**
	C_5	5	5	0.395	0.329
D5	C_3	3	4	0.528	0.352
	C_4	**2**	**6**	**0.0**	**0.0**
	C_5	2	5	0.158	0.132
D6	C_2	3	3	0.903	0.452
	C_4	**1**	**6**	**0.0**	**0.0**
	C_5	4	5	0.316	0.263

It noticed that the descending order of the concepts becomes different from the order in Table 2. The main reasons for the difference are:

1. The first effect of the fuzzy weighting schema, since the high weighted values are given to the concepts that are more occurrences in documents. For example, the concept C_6 is in two different orders as shown in Table 2 and Table 3. The weighing schema considered the concept C_6 an important concept although it occurred only one time in all documents.
2. The second effect of the fuzzy weighting schema is the appearing of new concepts with high fuzzy weighted values in the top of the list. For example, in Table 2 the concept C_5 does not satisfy the threshold weight value

although C_5 occurred 5 times in D4. In contrast in Table 3 the concept C_5 has a high fuzzy weighted value and exists in the top of the table.

Table 2. The concepts with their TF-IDF

Concept	**Documents**	**TF-IDF**
C_1	D4	1.431
C_1	D1	0.954
C_2	D6	0.903
C_6	**D1**	**0.778**
C_3	**D3**	**0.704**
C_2	D3	0.602
C_3	D5	0.528
C_5	**D4**	**0.395**
C_3	D2	0.352
C_5	D6	0.316
C_2	D1	0.301
C_5	D2	0.237
C_3	D1	0.176
C_5	D5	0.158
C_5	D3	0.79

Table 3. The concepts with their FUZZY TF-IDF

Concept	**Documents**	**Fuzzy TF-IDF**
C_1	D4	0.477
C_3	**D3**	**0.469**
C_2	D6	0.452
C_3	D5	0.352
C_5	**D4**	**0.329**
C_1	D1	0.318
C_2	D3	0.301
C_5	D6	0.263
C_3	D2	0.235
C_5	D2	0.197
C_2	D1	0.151
C_5	D5	0.132
C_6	**D1**	**0.129**
C_3	D1	0.117
C_5	D3	0.066

3.3 Incremental Association Rule Mining Phase

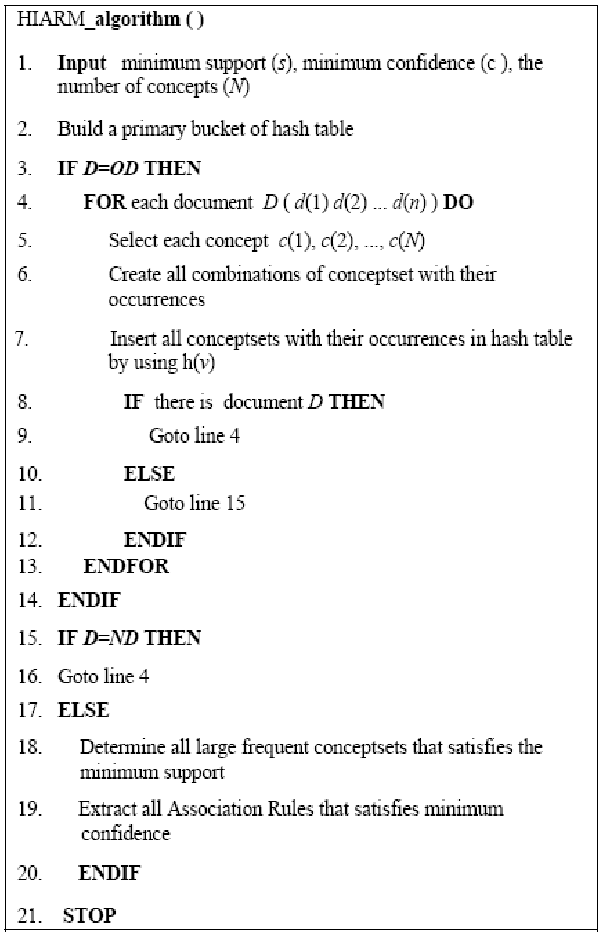
The main idea of the IARMMD system is that it avoids redoing the mining process on whole documents during the updating process. Hash-based Incremental Association Rule Mining Algorithm (HIARM) is introduced in the mining phase. The algorithm employs the power of data structure called Hash Table. The hashing function h(v) and concepts number (N) considered the key factors in hash table building and search performance.

3.3.1 Hash-based Incremental Association Rule Mining Algorithm (HIARM)

The proposed HIARM algorithm as in Figure 4 employs the following two main steps:

1. Based on the number of concepts (N) in the documents, a dictionary table was constructed for $N = 6$ concepts, and
2. There are also two main processes for a dynamic hash table: the building process, and the scanning process.

The mining process for HIARM algorithm includes the two processes (building and scanning process) on the given XML file that contains all concepts. The hash function h(v) = v mod N, where v is a key (location of primary concept) is used to build a primary bucket of the hash table. The algorithm scans only the XML file that contained all important concepts not the original documents. After building a primary bucket, the incremental mining process is performed in two phases: the first phase called the **First Time** that includes all two processes for HIARM algorithm (building process and the scanning process) on the given old documents, and the second phase called the **Update Time** that includes adding new documents to the dynamic hash table that has been built and saved. On the other words, start by applying the second process of HIARM algorithm (the scanning process) on the given new documents.



```
HIARM_algorithm ( )
1.  Input  minimum support (s), minimum confidence (c ), the
    number of concepts (N)
2.  Build a primary bucket of hash table
3.  IF D=OD THEN
4.      FOR each document  D ( d(1) d(2) ... d(n) ) DO
5.          Select each concept  c(1), c(2), ..., c(N)
6.          Create all combinations of conceptset with their
            occurrences
7.          Insert all conceptsets with their occurrences in hash table
            by using h(v)
8.            IF  there is  document D THEN
9.                Goto line 4
10.           ELSE
11.               Goto line 15
12.           ENDIF
13.     ENDFOR
14. ENDIF
15. IF D=ND THEN
16. Goto line 4
17. ELSE
18.     Determine all large frequent conceptsets that satisfies the
        minimum support
19.     Extract all Association Rules that satisfies minimum
        confidence
20.     ENDIF
21.  STOP
```

Fig 4: The HIARM algorithm

The HIARM algorithm has the following advantages:

1. The algorithm uses the old documentsets only one time to build the dynamic hash table.
2. The updating process updates the existing dynamic hash table, not building a new one
3. The algorithm permits the end user to change the threshold support and confidence factors.
4. Small size of dynamic hash table, since with changing the size of conceptsets the size of dynamic hash table will change.
5. Less number of conceptsets, since there is no conceptsets with zero occurrences will occupy a size in a dynamic hash table.

3.3.2 The HIARM Algorithm Case Study

Consider the following case study in Figure 5, where an old documentsets (*OD*) consists of 12 documents and 6 concepts {A, B, D, F, M, T} and new documentsets (*ND*) consists of 5 documents as shown in Figure 9 with the same concepts with given s = 50%, and c = 66%.

At First Time: consider the case study for *OD* which consists of 12 documents and N = 6 concepts with minimum support s = 50% and minimum confidence c = 66% as shown in Figure

5. The mining process includes the building and scanning process as follows:

DID	Old Documentsets *OD*
	Concepts
1	*A D F M T*
2	*B F T*
3	*A F M*
4	*A B F M T*
5	*A B D M T*
6	*B T*
7	*A F M T*
8	*A B F T*
9	*A F T*
10	*A T*
11	*B D T*
12	*A D*

Fig 5: The *OD*=12 documents

1. Based on the number of concepts in *OD*, create the dictionary table for these items.
2. Building a primary bucket for these items by using the hash function **h(*v*) = *v* mod *N*** to determine their locations
3. After that the process of scanning starts by scanning each document in the *OD* only once. We chose the second document in *OD* and tokenize it into number of small tokens then determine their locations in the dynamic hash table by using the hash function h(*v*).
4. This process continues until there is no document in the documentsets as shown in Figure 6. We noticed from the table that; there is no cell with zero occurrences, and the actual occupied cells are equal to 55 cells. Thus, the size of dynamic hash table is 533 byte.
5. The large frequent conceptset, with minimum support *s* = 50% are determined. Thus, the dynamic hash table is as shown in Figure 7.
6. The resultant rules that satisfy *c* = 66% are shown in Figure 8.

Lo (0)		*Lo* (1)		*Lo* (2)		*Lo* (3)		*Lo*(4)		*Lo* (5)	
Conceptset	*s*	Conceptset	*s*	Conceptset	*s*	Conceptset	*s*	Conceptset	*s*	Conceptset	*s*
A	9	*B*	6	*D*	4	*F*	7	*M*	5	*T*	10
DM	2	*DT*	3	*AD*	3	*AF*	6	*AM*	5	*AT*	7
ADM	2	*FM*	4	*FT*	6	*MT*	4	*DFT*	1	*DF*	1
FMT	3	*ADT*	2	*AFT*	5	*AMT*	4	*ADFT*	1	*ADF*	1
AFMT	3	*AFM*	4	*DFMT*	1	*DFM*	1	*BF*	3	*DMT*	2
BT	6	*AB*	3	*BFM*	1	*ADFM*	1	*ABF*	2	*ADMT*	2
ABT	3	*BFMT*	1	*ABFM*	1	*BFT*	3	*BMT*	2	*BM*	2
BEMT	1	*BDM*	1	*BDT*	2	*ABFT*	2	*ABMT*	2	*ABM*	2
ABDMT	1	*ABDM*	1	*ABDT*	1	*BD*	2				
		ABFMT	1	*ADFMT*	1	*ABD*	1				

Fig 6: The Dynamic hash table for *OD*=12 and *N*=6.

Lo (0)		*Lo* (1)		*Lo* (2)		*Lo* (3)		*Lo* (5)	
conceptset	*s*	conceptset	*s*	conceptset	*s*	conceptset	*s*	conceptset	*s*
A	9	*B*	6	*FT*	6	*F*	7	*T*	10
BT	6					*AF*	6	*AT*	7

Fig 7: The large frequent conceptset that satisfy *s* = 50 %.

B => *T*	(100 %)
F => *T*	(85.71 %)
A => *F*	(66.67 %)
F => *A*	(85.71 %)
A => *T*	(77.78 %)
T => *A*	(70%)

Fig 8: The resultant rules that satisfy *c* = 66 %.

At Update Time : After some updating activities on the same *OD* in Figure 5, the new documentsets *ND* consists of 5 documents as shown in Figure 9, is added to the *OD* with the same concepts, given *s* = 40% and *c* = 60%.

DID	New Documentsets *ND*
	concepts
1	*A B D M*
2	*A D M T*
3	*A F M*
4	*D T*
5	*A B M*

Fig 9: The *ND* = 5 documents.

The mining process includes the scanning process only as follows:

1. Open the dynamic hash table of *OD*=12 documents that has been built before, as shown in Figure 6.
2. Repeat the steps of the scanning process on the *ND* by scanning each document in the *ND* only once. Then, update the frequency or occurrence of each documentsets in dynamic hash table as shown in Figure 10.
3. The large frequent conceptset, with minimum support *s* = 40% are determined. Thus, the dynamic hash table is as shown in Figure 11.
4. The resultant rules that satisfy *c* = 60% are shown in Figure 12.

Lo (0)		*Lo* (1)		*Lo* (2)		*Lo* (3)		*Lo*(4)		*Lo* (5)	
itemset	*s*	itemset	*s*	itemset	*s*	itemset	*s*	itemset	*s*	itemset	*s*
A	13	*B*	8	*D*	4	*F*	8	*M*	9	*T*	12
DM	4	*DT*	5	*AD*	3	*AF*	7	*AM*	9	*AT*	8
ADM	4	*FM*	5	*FT*	6	*MT*	5	*DFT*	1	*DF*	1
FMT	3	*ADT*	3	*AFT*	5	*AMT*	5	*ADFT*	1	*ADF*	1
AFMT	3	*AFM*	5	*DFMT*	1	*DFM*	1	*BF*	3	*DMT*	3
BT	6	*AB*	5	*BFM*	1	*ADFM*	1	*ABF*	2	*ADMT*	3
ABT	3	*BFMT*	1	*ABFM*	1	*BFT*	3	*BMT*	2	*BM*	4
BDMT	1	*BDM*	2	*BDT*	2	*ABFT*	2	*ABMT*	2	*ABM*	4
ABDMT	1	*ABDM*	2	*ABDT*	1	*BD*	3				
		ABFMT	1	*ADFMT*	1	*ABD*	2				

Fig 10: The Dynamic Hash Table for *TD*=17, *N*=6.

Lo (0)		*Lo* (1)		*Lo* (3)		*Lo*(4)		*Lo* (5)	
itemset	*s*	itemset	*s*	itemset	*s*	itemset	*s*	itemset	*s*
A	13	*B*	8	*F*	8	*M*	9	*T*	12
				AF	7	*AM*	9	*AT*	8

Fig. 11: The large frequent conceptset that satisfy *s* = 40%.

F => A	(87.5 %)
A => M	(69.2 %)
M => A	(100 %)
A => T	(61.5 %)
T => A	(66.6 %)

Fig 12: The resultant rules that satisfy *c* = 60%.

4. EXPERIMENTAL RESULTS

The experiments are performed to compare the performance of the HIARM algorithm in the IARMMD system with the two systems that are dependent on Apriori and FUP algorithms. The comparison is suited to the number of extracted association rules and the execution time. The corpus of the experiments is selected from the PubMed abstracts with keyword search "*breast cancer treatments and side effects*" [6]. All experiments are applied on *TD* which is divided into *OD* and *ND*. The systems are implemented by using VS .Net 2010 (C#) and the experiments were performed on Intel Core2 Duo, 1.8 GHz system with Windows Vista Home Premium and 2 Giga of RAM.

4.1 The number of Association Rules

A large number of association rules can be extracted by entering different values of minimum support and confidence in the mining process. The IARMMD system always gives the best results by using low support and high confidence values. Moreover, the number of concepts that entered to the mining process is fewer by using the fuzzy weighting schema. It noticed that the number of extracted association rules in IARMMD system is useful and always less than that in Apriori-based and FUP-based systems. The reason returns to the strong effect of using the fuzzy weighting schema in IARMMD system.

4.2 The execution time for Apriori, FUP and HIARM algorithms at first time and update time

At First Time: The methodology works as follows. First, the execution time after applying Apriori and FUP algorithms on the same *OD* = 1000, 2500, 5000 documents with different user thresholds values is determined. Second, the execution time after applying HIARM algorithm (hash table Build; including building time for hash table in HIARM) is determined on the same *OD* and the same user thresholds values. Third, the execution time for Apriori, FUP and HIARM algorithm are compared as shown in Figure 13.

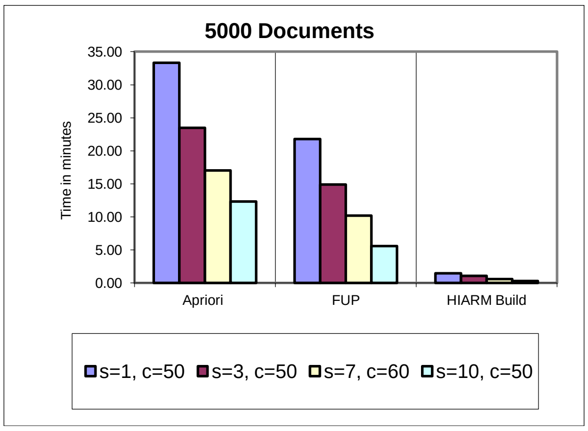


Fig 13: Apriori, FUP and HIARM execution time for *OD*=5000 document at first time.

From Figure 13, it is seen that the execution time of the HIARM algorithm is much better than that of the Apriori and FUP algorithm. At small documentsets with different user thresholds values, the execution time of the Apriori and FUP algorithms is about between two or three fold that of the HIARM algorithm. As the documentsets becomes very large, the HIARM algorithm gives execution time better than that of the Apriori and FUP algorithms. The results reveal that the execution time of the Apriori algorithm and FUP is about tenth and sixth fold of the HIARM algorithm respectively. The reason is the consumed time for the numbers of scanning on the documentsets and the searching process.

At Update Time: The methodology works as follows. First, the execution time after applying the Apriori and FUP algorithms on total documentsets (*TD* = *OD* + *ND*) *TD* = 2000, 5000 and 10000 documents with different user thresholds values is determined. Second, the total execution time after updating process in HIARM algorithm with the same user thresholds values is determined. On the other words, the total execution time (HIARM Total) includes the execution time for building dynamic table at first time with *OD* (HIARM build) and updating time after adding *ND* to hash table (HIARM update). Third, the execution time for Apriori algorithm, FUP algorithm and the total execution time for HIARM algorithm (HIARM Total) are compared as shown in Figure 14.

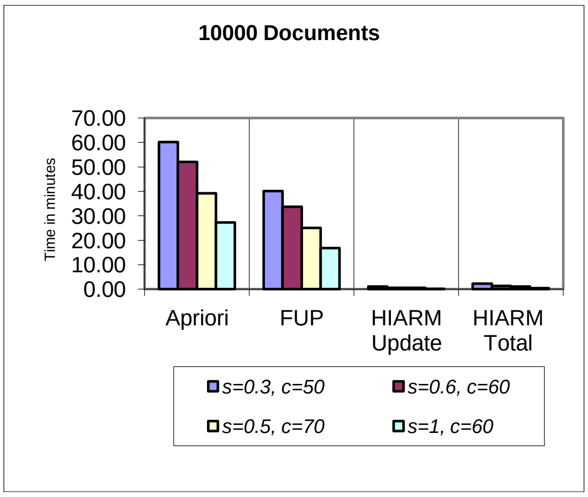


Fig 14: Apriori, FUP and HIARM execution time for *TD* =10000 documents when update.

From Figure 14, it is seen that the total execution time of HIARM algorithm is smaller than that of the Apriori and FUP algorithm in the updating process. The results reveal that the execution time of the Apriori and FUP algorithm is about between tenth and sixth fold of the HIARM algorithm respectively. In the Apriori algorithm, the multiple scanning processes on the total documentsets increased the execution time. While FUP algorithm requires scanning passes over an original documents several times when new frequent conceptsets are found. This can degrade the performance of FUP algorithm. Whereas the HIARM algorithm is applied only on the *ND* and the execution time (HIARM Update) consumes for updating the existing hash table. Thus, the total execution time (HIARM Total) is always better than that of Apriori and FUP algorithm in all cases.

5. CONCLUSIONS

This paper presented an online text mining system for incremental association rules mining from medical documents. The IARMMD system avoided redoing the mining process on whole documents during the updating process. The Hash-based Incremental Association Rules Mining Algorithm used in the mining process to decrease the number of scanning and the execution time. Moreover, it permits the end user to extract new association rules with different user threshold values with no need to re-scan the original documents. The performance of the algorithm is compared with both Apriori and FUP algorithms for the execution time and the evaluation of the extracted association rules. The results reveal that the number of extracted association rules in the IARMMD system is always less than that in Apriori-based and FUP-based systems. Furthermore, the execution time of HIARM algorithm is much better than Apriori and FUP algorithms in the updating process in all experimental cases. In future work we intend to develop the IARMMD system to deal with WebPages. Since the philosophy of the Incremental process is compatible with the dynamical characteristics of the WebPages. Such type of characteristics is useful for real-world applications such as web mining.

6. REFERENCES

[1] Agrawal, R., Imielinski,T. and Swami, A. 1993. Mining association rules between Sets of items in large databases. In Proceedings of the ACMSIGMOD Int. Conf. on Management of Data, Washington, D.C.

[2] Han, J., Cai, Y. and Cercone N. 1993. *Data-driven Discovered of Quantitive Rules in Relational Databases*," In Proc. of IEEE KDE Conference.

[3] Cheung, D., Han, J., Ng V., and Wong, C. Y. 1996 Maintenance of discovered association rules in large databases: An incremental updating technique. In 12th IEEE International Conference on Data Engineering.

[4] Agrawal, R., and Srikant, R. 1994. Fast algorithms for mining association rules," In Jorge B. Bocca, Matthias Jarke, and Carlo Zaniolo, editors, Proc. 20th Int. conf. of very Large Data Bases, VLDB, Santigo, Chile.

[5] Mahgoub, H., Keshk, A., Torkey, F. and Ismail N. 2010. An Efficient Online System of Concept Based Association Rules Mining," in Proc. 7th Int. Conf. on Informatics and Systems (INFOS 2010), Faculty of Computers and Information, Cairo University, Egypt.

[6] (2009) the PubMed website [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/

[7] Park, J. S., Chen, M. S., and Yu, P. S. 1995. An effective hash based algorithm for mining association rules. In Proc.1995 ACM-SIGMOD Int. Conf. on Management of Data, San Jose, CA, pp. 175-186.

[8] Cheung, D., Han, J., Ng, V., and Wong, C. Y. 1996 Maintenance of discovered Knowledge: A Case in Multi-level Association Rules. In Proceedings of the 2nd Int. Conf. on Knowledge Discovery and Data Mining, pp. 307-310.

[9] Cheung, D. W., Lee, S.D. and Kao B. 1997. A General incremental technique for maintaining discovered association rules" , In Proc. of the 5th Intl. Conf. on Database Systems for Advanced Applications (DASFAA'97), Melbourne, Australia.

[10] Lee, S. and Cheung, D. 1997. Maintenance of Discovered association rules. When to update? In Proc. of Research Issues on Data Mining and Knowledge Discovery, pp 51- 58.

[11] Chang, C.C., Li, Y.C. and Lee, J.S. 2005. An efficient algorithm for incremental mining of association rules. In Proc. of the 15th Int. Workshop on research issues in data engineering: stream data mining and applications (RIDE-SDMA'05), IEEE.

[12] Toivonen, H. 1996. Sampling Large Databases for Association Rules. Proceeding of the 22th International conference on Very Large Data Bases.

[13] Thomas, S., Bodagala, S., Alsabti, K., and Ranka, S. 1997. An efficient algorithm for the incremental updation of association rules in large databases. In Proceedings of the 3rd Intl. Conf. on Knowledge Discovery and Data Mining (KDD'97), New Port Beach, California, pp. 263-266.

[14] R. Feldman, Y. Aumann, and O. Lipshtat, "Borders: An efficient algorithm for association generation in dynamic databases", Journal, Intelligent Information System, 1990, pp. 61-73.

[15] T.P. Hong C.Y. Wang and Y.H. Tao, "A new incremental data mining algorithm using pre-large itemsets", Journal, Intelligent Data Analysis, Vol. 5, No.2, pp. 111-129, 2001.

[16] Amornchewin, R. and Kreesuradej, W. 2007. Incremental association rule mining using promising frequent itemset algorithm. In Proceeding 6th International Conference on Information, Communications and Signal Processing, pp.1-5.

[17] Amornchewin, R. and Kreesuradej, W. 2008. Probability-based incremental association rule discovery algorithm. The 2008 International Symposium on Computer Science and its Applications (CSA-08), Australia.

[18] R. Amornchewin and W. Kreesuradej, "Mining Dynamic Databases using Probability-Based Incremental Association Rule Discovery Algorithm", Journal of Universal Computer Science, vol. 15, no. 12 , 2009, pp. 2409-2428 .

[19] R. Amornchewin, "Probability-based Incremental association rules discovery algorithm with hashing Technique", Int. Journal of Machine Learning and Computing, vol. 1, no. 1, 2011, pp. 43-48.

[20] Zhu, Y. 2010. Improvement and Realization of Association Rules Mining Algorithm Based on FP-tree. 2nd International Conference on Information Science and Engineering (ICISE), China.

[21] J. Han, J. Pei, Y. Yin and R. Mao, "Mining Frequent Patterns without Candidate Generation: A Frequent-Pattern Tree", Data Mining and Knowledge Discovery, pp.53–87, IEEE 2004.

[22] Han, J., Pei, J. and Yin, Y. 2000. Mining frequent patterns without candidate generation. The ACM SIGMOD Int. Conference on Management of Data.

[23] Zeng, H. and Bangrong, S. 2010. An Improved Algorithm of FP - tree Growth Based on Mapping. International Conference on Computer Application and System Modeling (ICCASM).

[24] Jian-ping, L., Ying, W. and Fan-ding, Y. 2010. Incremental-Mining algorithm Pre-FP in association rules based on FP-tree. Networking and Distributed Computing (ICNDC), First Int. Conference, IEEE.

[25] Lin, C.-W., Hong, T. –P., & Lu, W. –H. "The Pre-FUFP algorithm for incremental mining" Journal of Expert Systems with Applications, 36, 2009.

[26] Hong, T. P., Lin, J. W. and Wu, Y. L. 2006. Maintenance of fast updated frequent pattern trees for record modification. The Int. Conference on Innovative Computing, Information and Control, pp. 570-573, IEEE.

[27] B. Nath, D K Bhattacharyya and A. Ghosh, "Discovering Association Rules from Incremental Datasets," *Int. J. of* Computer Science & Communication Vol. 1, No. 2, July-December 2010.

[28] Mahgoub, H. and Rösner, D. 2006. Mining association rules from unstructured documents. In Proc. 3rd Int. Conf. on Knowledge Mining, ICKM, Prague, Czech Republic, pp. 167-172.

[29] H. Mahgoub, D. Rösner, N. Ismail and F. Torkey, "A Text Mining Technique Using Association Rules Extraction" *Int. J. of Computational Intelligence,* Vol.4, Nr.1, 2007 *WASE.*