Comparative Analysis of Frequent Closed Itemset Mining Algorithms

S. Sharmila, Ph.D. Research Scholar, Department of Computer Science, Bharathiar Coimbatore. University, India. sharmilasathyanathan@gmail.com

Dr. S.Vijayarani, Assistant Professor, Department of Computer Science, Bharathiar University, Coimbatore, India. vijimohan_2000@yahoo.co

Abstract Association rule is a popular data mining technique, helps to identify the relationship between attributes in large databases. It is determined to discover frequent patterns and rules generation from the databases. Closed itemset is defined as frequent itemset whereas none of its immediate supersets has the same support in the itemset. Closed frequent itemset removes some redundant rules, provide compact representations and helps to determine the support of their subset. The main objective of this research work is to find the frequent closed items from frequent items by using four existing algorithms namely Apriori Close, DCI closed, LCM and Charm. From this analysis, it is observed that DCI closed algorithm has produced better results.

Keyword - Apriori Close, CHARM, DCI Closed, Frequent closed item mining and LCM Algorithm

I. INTRODUCTION

Association Rules Mining is one of the data mining technique, which finds a correlation among items of the database. Market Basket Analysis is a good example of ARM. It discovers items which are frequently purchased together by the customers [1]. Association rules have been mostly used in many applications areas like library circulation data, protein composition, population and economic census etc [2]. Frequent itemset is an itemset which satisfies the minimum threshold value. The occurrence of an item should be equal to or greater than the threshold value, those items are frequent items

A Closed frequent itemset first identifies all frequent itemsets based on the minimum support. From this, candidate itemsets are generated and frequent itemsets are identified [1]. An itemset is defined as closed itemset whose minimum support of superset should not be equal to the support count of original itemset [2].Closed Frequent Itemsets are the compact representation of the Frequent Itemsets which can save memory and time for large, dense data. Closed frequent itemsets can be mined by pruning the search space. Pruning strategies consists of three strategies i.e. item merging, sub itemset pruning and item skipping [3].

In item merging, for example every transaction containing a frequent itemset A also contains an itemset B but not any proper superset of B, then AUB forms a frequent closed itemset. [4]. In Sub-itemset pruning, If a frequent itemset A is a proper subset of frequent closed itemset B and support count(A) = support count(B), then A and all of A's child node in the set enumeration tree cannot be frequent closed itemsets and thus can be pruned. In item skipping, at each level of depth-first mining of closed itemsets, there will be a prefix itemset A associated with a header table and a projected database. If a local frequent item has the same support in several header tables at different levels, prune that item from the header tables at higher level [5].closed frequent itemset is explained with a small example given below.

Example

Database consists of 6 items and 6 transactions

T id	Items
T1	A B C E
T2	A C D E
T3	A B D
T4	C D E
T5	A D E
T6	A C D E

Pass 1

Count the occurrences of each itemset. We assume that Min_support = 3, hence, the items whose occurrence is 3 or greater than 3 are considered as frequent items.

Item	Occurrence
A	5
B	2
C	4
D	5
E	5

→ Frequent items

 DOI : 10.18231/2454-9150.2018.0616

.A, C, D, E → frequent items, since its occurrences are greater than min_support.
Frequent closed itemset

To find the frequent closed itemsets, it is necessary to generate the candidate itemsets i.e. 2-itemsets from the frequent 1-item set.

Pass 2: Candidate Generation

Item	{A C}	{A D}	{AE}	{ C D}	{C E}	{ D E}
Occurrence	3	4	4	3	4	4

Now, consider each frequent item and find the occurrences of that item with other items. i.e. Consider item A, then the occurrences of (A,C) (A,D) and (A,E) is (A,C)=3, (A,D)=4, (A,E)=4. From this, the closed frequent items are identified by using the given condition.

(Occurrence of item A in 1-itemset) ≠ (Occurrences of items in candidate (A, C) (A, D) and (A, E))

i.e. 5 ≠ {3, 4, 4} → True. This condition is satisfied, hence item A is considered as frequent closed item.

Next consider Item C, The occurrence of item C is 4, the occurrences of (C, D) =4 (C, E) =5

C ≠ (C, D), (C, E)

4 ≠ 4, 5 → False.

Hence item C is not a frequent closed item.

Next consider Item D, The occurrence of item D is 5, the occurrences of (D, E) =4

D ≠ (D, E)

5 ≠ 4, True, so, Item D is a frequent closed itemset

Frequent closed items in 1-itemset are {A} and {D}

In order to find frequent closed 2-itemset, it is necessary to generate candidate 3-itemset from the frequent 2-itemsets.

2-itemset

Frequent 2-item sets are (A, C), (A, D), (A, E), (C, D), (C, E), (D, E), since their occurrences are greater than minimum support

Item	{A C D}	{ A C E}	{A D E}	{C D E}
Occurrence	2	3	4	3

Now, find the occurrences with other itemsets for itemset (A,C) (A,D) (A,E) (C,D) (C,E) and (D,E). Frequent 3-itemsets are {A C E} {A D E} {C D E}. From this, frequent closed itemsets are identified, by using the above condition.

Consider (A, C), its occurrences are 4
(A, C) ≠ {A C E}
3 ≠ {3} → False.
Itemset {A, C} is not frequent closed itemset.
Next Itemset {A, D}
(A, D) ≠ {A D E}
4 ≠ {4} → False.
Itemset {A, D} is not frequent closed itemset.
Next Itemset {A, E}
(A, E) ≠ {A C E} {A D E}
4 ≠ {3} {4} → False.
Itemset {A, E} is not frequent closed itemset.
Next Itemset {C, D}
(C, D) ≠ {C D E}
3 ≠ {3} → False.
Itemset {C, D} is not frequent closed itemset.
Next Itemset {C, E}
(C, E) ≠ {A C E}, {C D E}
4 ≠ {3} {3} → True
Itemset {C, E} is frequent closed itemset.
Next Itemset {D, E}
(D, E) ≠ {A D E}, {C D E}
4 ≠ {4}, {3} → False.
Itemset {D.E} is not a frequent closed itemset.
Here Frequent closed items in 2-itemset is {C, E}.
3-itemset
Frequent 3-item sets are (A, C, E), (A, D, E), (C, D, E) since their occurrences are greater than minimum support.

Item	{A C D E}
Occurrence	2

Iteration has stopped here because the occurrences of 4-itemset (A C D E) is less than minimum support, it is not a frequent itemset.

Pass	Frequent Itemset	Frequent Closed Itemset
1	A, C, D, E	{A} {D}
2	(A,C), (A,D), (A,E), (C,D), (C,E), (D,E),	{C, E}.
3	(A, C, E), (A, D, E), (C, D, E)	-

Frequent Closed itemset are {A} {D} {C, E}.

II. LITERATURE REVIEW

Ferenc Bodon et.al [1] examined the relationship between closed itemset mining, the complete pruning technique and item ordering in the Apriori algorithm. Author had proposed intersection-based technique and explained about complete pruning technique .From the analysis the proposed techniques gives better result in, memory consumption and run-time.

Mohammed et.al [2] presented an efficient algorithm CHARM, for mining all frequent closed itemsets. It had enumerated closed sets using a dual itemset-tidset search tree, in this research work efficient hybrid search was used

 DOI : 10.18231/2454-9150.2018.0616

to skips many levels. Author had used a technique called diffsets to reduce the memory footprint of intermediate computations. Finally fast hash-based approach was used to remove "non-closed" sets found during computation. Author had proved that proposed algorithm had given better results and linearly scalable in the number of transactions.

Vikram Pudi et.al [3] proposed a new framework, namely, the generalized closed (or g-closed) itemset framework. This had allowed for a small tolerance in the accuracy of itemset supports, author had analysed that number of redundant rules more than previously estimated. The framework had integrated into both level wise algorithms (Apriori) and two-pass algorithms (ARMOR). Experimental results had proved that gclosed itemsets provides significant performance improvements with variety of database.

Maryam Shekofteh et.al [6] Reviewed and compared the FCI algorithms with other algorithms. Results had showed that each algorithm has some advantages and disadvantages for mining in dense and sparse datasets based on its applied strategy. However, DCI-Closed algorithm has produced better results than other ones.

Takeaki Uno et.al [9] had proposed an efficient algorithm LCM (Linear time Closed pattern Miner) for mining frequent closed patterns from large transaction databases. Prefix-preserving closure extension of closed patterns was the main theoretical contribution in this research work. Algorithm had enabled to search all frequent closed patterns in a depth-first manner, in linear time for the number of frequent closed patterns. The proposed algorithm do not occupy more storage space for obtained patterns. The existing algorithms were compared with proposed algorithms and it has observed that proposed algorithm gives best result.

Ansel Y. Rodríguez-González et.al [15] had proposed a novel closed frequent similar pattern mining algorithm (CFSP-Miner). The algorithm discovers frequent patterns by traversing a tree that contains all the closed frequent similar patterns. Many lemmas were used to prune the search space. The results had shown that CFSP-Miner is more efficient than the other frequent similar pattern mining algorithm. However, CFSP-Miner was able to find the closed similar patterns, reduced size of the discovered frequent similar pattern set without information loss.

III. METHODOLOGY

3.1 THE A-CLOSE ALGORITHM

The A-Close algorithm consists of two main steps. First, a level-wise search is implemented to discover the generators which have sufficient support. The generators are then used as inputs and the outputs are resultant as closed sets [1]. The second step of the A-Close algorithm includes the generators found in the first step and inputting them into f(x) to obtain the closed sets. In Apriori-Close the infrequent candidate deletion is extended by a step, where the subsets of the frequent candidate are checked. All subsets are marked as closed by default, which is changed if the subsets support equals to the candidate's actually examined. Consequently, in Apriori-Close all subsets of the frequent candidates are generated [2]. The closed itemset filtering is done in the candidate generation phase and intersection-based pruning is applied.. In this method the subsets are already determined; hence checking support equivalence does not require any extra travels. [3].

```
Algorithm A-Close algorithm
1) Generators in G1← {1-item sets};
2) G1 ←Support-Count (G1);
3) forall generators p ∈ G1 do begin
4)        if (support(p) < minsup) then delete p from
G1; // Pruning infrequent
5) end
6) level ← 0;
7) for (i ← 1; Gi .generator ≠ Ø i++) do begin
8)        Gi+1← AC-Generator(Gi); // Creates (i+1)-
generators
9) end
10) if (level > 2) then begin
11)       G ← ∪ Gj ∈ | j < level-1; // Those generators
are all closed
12)       forall generators p ∈ G do begin
13)               P.closure ← p.generator;
14)       end
15) end
16) if (level ≠ 0) then begin
17) G'←∪ {Gj | j ≥ level-1}; // some of those
generators are not closed
18) G'← AC-Closure (G');
19) end
20) Answer F C ← {c.closure, c.support | c ∈ G ∪
G'};
```

3.2 THE CHARM ALGORITHM

The CHARM [4] algorithm is a more efficient approach to solving this problem. This algorithm performs a search for closed frequent sets over a novel IT-tree search space. Each node in the IT-tree, represented by an itemset-tidset pair, X ×t(X), is in fact a prefix-based class. All the children of a given node X, belong to its equivalence class, since they all share the same prefix X. an equivalence class is defined as [P] = {l1, l2, · · · , ln}[5], where P is the parent node (the prefix), and each li is a single item, representing the node Pli × t(Pli). For example, the root of the tree corresponds to the class [] = {A, C, D, T,W} [6]. The leftmost child of the root consists of the class [A] of all itemsets containing A as the prefix, i.e. the set {C,D, T,W}. As each class member represents one child of the parent node. A class represents items that the prefix can be extended with to obtain a new frequent node [7].

 DOI : 10.18231/2454-9150.2018.0616

```
CHARM (D, min sup):
1. [P] = {Xi × t(Xi) : Xi ∈ I∧ σ(Xi) ≥ min sup}
2. CHARM-Extend ([P], C = )
3. return C //all closed sets CHARM-Extend ([P], C):
4. for each Xi × t(Xi) in [P]
5. [Pi] = and X = Xi
6. for each Xj × t(Xj ) in [P], with Xj ≥f Xi
7. X = X ∪ Xj and Y = t(Xi) ∩ t(Xj )
8. CHARM-Property([P], [Pi])
9. if ([Pi] = ) then CHARM-Extend ([Pi], C)
10. delete [Pi]
11. C = C ∪ X //if X is not subsumed CHARM-
Property ([P], [Pi]):
12. if (σ(X) ≥ minsup) then
13. if t(Xi) = t(Xj ) then //Property 1
14. Remove Xj from [P]
15. Replace all Xi with X
16. else if t(Xi) ⊂ t(Xj ) then //Property 2
17. Replace all Xi with X
18. else if t(Xi) ⊃ t(Xj ) then //Property 3
19. Remove Xj from [P]
20. Add X × Y to [Pi] //use ordering
21. else if t(Xi) = t(Xj ) then //Property 4
22. Add X × Y to [Pi] //use ordering f
```

3.3 DCI Closed

The DCI Closed: a Fast and Memory Efficient Algorithm to Mine Frequent Close [8]. One of the main problem occurs at the time of mining the frequent closed itemsets is the duplication. General technique of this algorithm is to find and remove the duplicate closed itemsets, without the need of storing the whole closed itemsets in main memory. This is one of the very important features of this algorithm their approach can be exploited with substantial performance benefits by any algorithm that adopts a vertical representation of the dataset. This algorithm contains mainly three functions CLOSED SET, PRE SET, POST SET. From CLOSED SET new closed set, new generators and corresponding closed sets can be building. While the composition of POST SET guarantees that the various generators will be produced according to the lexicographic order. The composition of PRE SET guarantees that duplicate generators will be pruned by function is dup () [9].

DCI-Closed a Fast and Memory Efficient Algorithm to Mine Frequent Closed Itemsets algorithm gives best result because this technique is finding and removing duplicate itemsets without keeping to store all itemsets in memory [10]. Because this technique used vertical bitmap for represent the dataset. So this technique is very effective.

```
1: procedure DCI Closed(CLOSED SET, PRE SET,
POST SET)
2: for all i ∈ POST SET do . Try to create a new
generator
3: new gen ← CLOSED SET ∪ i
4: if supp(new gen) ≥ min supp then . new gen is
frequent
5: if is dup(new gen, PRE SET) = FALSE then .
Duplication check
6: CLOSED SETNew ← new gen
7: POST SETNew ← ∅
8: for all j ∈ POST SET, i ∈ j do . Compute closure of
new gen
9: if g(new gen) ⊆ g(j) then
10: CLOSED SETNew ← CLOSED SETNew ∪ j
11: else
12: POST SETNew ← POST SETNew ∪ j
13: end if
14: end for
15: Write out CLOSED SETNew and its support
16: DCI Closed(CLOSED SETNew, PRE SET, POST
SETNew)
17: PRE SET ← PRE SET ∪ i
18: end if
19: end if
20: end for

21: end procedure
23: function is dup(new gen, PRE SET)
24: for all j ∈ PRE SET do . Duplicate check
25: if g(new gen) ∈ g(j) then
26: return FALSE
27: end if
```

3.4 LCM

LCM (Linear time Closed itemset Miner) algorithms are based on backtracking algorithms, and use efficient techniques for the frequency counting[11].LCM algorithms compute the frequency efficiently without keeping previously obtained itemsets in memory [12]. Database reduction performs well when the minimum support is large, and many existing algorithms use it. LCM algorithms also use database reduction [13]. LCM uses prefix preserving closure extension (ppc) for generating closed itemsets [14].

 DOI : 10.18231/2454-9150.2018.0616

```
Algorithm LCM()
1.  X := I(T (∅)) /* The root ⊥ */
2. For i := 1 to |E|
3. If X[i] satisfies (cond2) and (cond3) then
Call LCM Iter( X[i], T (X[i]), i ) or
Call LCMd Iter2( X[i], T (X[i]), i, DJ )
based on the decision criteria
4. End for
LCM Iter( X, T (X), i(X) ) /* occurrence deliver */
1. output X
2. For each T ∈  T (X)
For each j ∈  T, j > i(X), insert t to J [j]
4. For each j,J [j] _= ∅ in the decreasing order
5. If |J [j]| ≥ α and (cond2) holds then
LCM Iter( T (J [j]),J [j], j )
6. Delete J [j]
7. End for
LCM Iter2( X, T (X), i(X), DJ ) /* diffset */
1. output X
2. For each i,X[i] is frequent
3. If X[i] satisfies (cond2) then
4. For each j,X[i] ∪ {j} is frequent,
DJ _[j] := DJ [j] \ DJ[i]
5. LCM Iter2( T (J [j]),J [j], j, DJ _ )
6. End if
7. End for
```

IV. RESULT AND DISCUSSION

The dataset was taken from UCI repository; it consists of two different data sets, i.e. mushroom and chess. Whereas mushroom dataset consists of 14275 transactions and 22 items, and chess dataset consists of 8126 transactions and 18 items. The performance metrics like a number of frequent closed items, execution time and memory usage are compared .This work is done in an Intel Core i5 processor running at 3.30 GHz, 4 GB RAM and 32 bit Windows 8. From the above analysis DCI closed algorithm gives better results than other algorithms. Table 1 gives the frequent closed items count for existing algorithms for different data sets.

Table. 1 Number of Frequent Closed Itemsets

Dataset	Algorithms	Frequent Closed Itemset
Chess	Apriori Close	1197
	DCI closed	2305
	Lcm	1234
	Charm	1467
Mushroom	Apriori Close	2361
	DCI closed	3202
	Lcm	2311
	Charm	1985

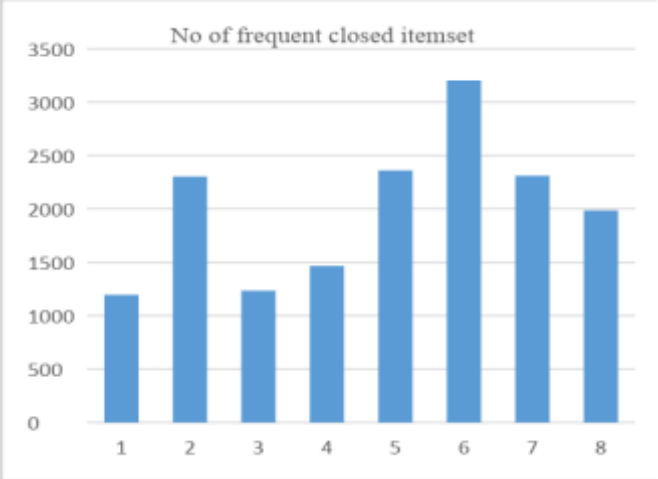

Figure.1 Analysis of frequent items

Figure 1 describes the analysis of frequent closed items count for existing algorithms. It is observed that DCI closed algorithm gives best results.

Table 2 gives the result of execution time in milliseconds using existing algorithms for different data sets.

Table.2 Execution Time in Milliseconds

Dataset	Algorithms	Total time
Chess	Apriori Close	1972
	DCI closed	1743
	Lcm	1804
	Charm	1961
Mushroom	Apriori Close	4202
	DCI closed	3452
	Lcm	3578
	Charm	3662

Figure.2 Analysis of Execution time

Figure 2 describes the analysis of Execution Time in milliseconds for algorithms for different data sets. From the above analysis it is concluded that DCI Closed algorithm gives better result other algorithms.

 DOI : 10.18231/2454-9150.2018.0616

Table 3 depicts the outcome of Memory Usage using existing algorithms for different data sets.

Table.3 Memory Usage in Kilobytes

Dataset	Algorithms	Memory usage
Chess	Apriori Close	4967
	DCI closed	3674
	Lcm	4109
	Charm	4881
Mushroom	Apriori Close	6463
	DCI closed	5731
	Lcm	6247
	Charm	6161

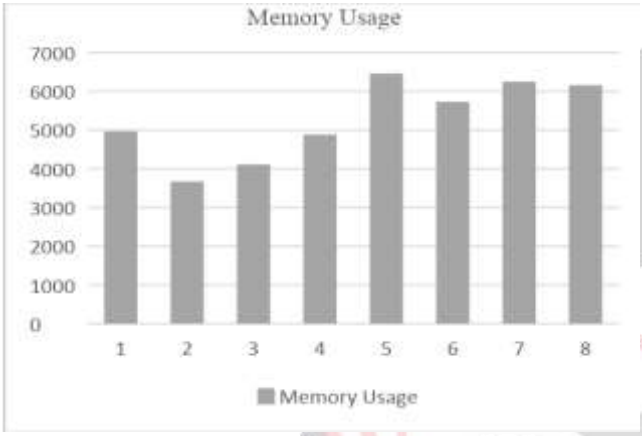

Figure.2 Analysis of Memory usage

Figure 3 shows the analysis of Memory Usage in Kilobytes for existing using different data sets. From the above analysis it is observed that DCI closed algorithm gives better result than other algorithms.

V. CONCLUSION AND FUTURE TRENDS

Association rule mining is the most effective data mining technique to discover hidden pattern from large volume of data. The main idea of this research work is to find the frequent closed itemset. In this research work four existing algorithms and two different datasets were compared with Performance metrics like total execution time, memory usage, and number of frequent closed item count. From the above analysis it has observed that DCI Closed algorithm has found more closed frequent itemset with minimum execution time and memory comparing to other algorithms. In future, this work will be implemented with different size and types of dataset like medical, bioinformatics, CRM, telecommunication etc.

REFERENCES

[1] Ferenc Bodon and Lars Schmidt Thieme The Relation of Closed Itemset Mining, Complete Pruning Strategies and Item Ordering in Apriori-based FIM algorithms (Extended version) Department of Computer Science and Information Theory, Budapest University

[2] Vikram Pudi _ Jayant R. Haritsa Generalized Closed Itemsets for Association Rule Mining Database Systems Lab, SERC, Indian Institute of Science, Bangalore.

[3] Rakesh Agrawal, Ramakrishnan Srikant, Fast Algorithms for Mining Association Rules in Large Databases, Proceedings of the 20th International Conference on Very Large Data Bases (VLDB'94), pages 487-499, 1994.

[4] Mohammed J. Zaki ∗ and Ching-Jui Hsiao Charm: An Efficient Algorithm for Closed Itemset Mining pg 451-47

[5] Xin Ye, Feng Wei, Fan Jiang and Shaoyin Cheng An Optimization to CHARM Algorithm for Mining Frequent Closed Itemsets. 978-1-5090-0154-5/15 $31.00 © 2015 IEEE DOI 10.1109/CIT/IUCC/DASC/PICOM.2015.33

[6] Maryam Shekofteh, A survey of algorithms in FCIM 2010 International Conference on Data Storage and Data Engineering 978-0-7695-3958-4/10 $26.00 © 2010 IEEE DOI 10.1109/DSDE.2010.32.

[7] Claudio Lucchese Salvatore Orlando DCI Closed: a Fast and Memory Efficient Algorithm to Mine Frequent Closed Itemsets. 978-0-7695-3958-4/10 $26.00 © 2010 IEEE DOI 10.1109/DSDE.2010.32 29.

[8] .Dungarwal Jayesh Neeru Yadav A Review paper for mining Frequent Closed Itemsets International Journal of Advance Research in Computer Science and Management Studies.

[9] Takeaki Uno, Tatsuya Asai, YuzoUchida, and Hiroki Arimura An Efficient Algorithm for Enumerating Closed Patterns in Transaction Databases. DS 2004, LNAI 3245, pp. 16–31, 2004.

[10] Takeaki Uno, Tatsuya Asai, Yuzo Uchida, Hiroki Arimura LCM: An Efficient Algorithm for Enumerating Frequent Closed Item Sets

[11] Takeaki Uno, Masashi Kiyomi, Hiroki Arimura LCM ver: Efficient Mining Algorithm for Frequent/Closed/Maximal Itemsets National Institute of Informatics 2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo, 101-8430, Japan.

[12] Nicolas Pasquier, Yves Bastide, Ra_k Taouil, and Lot_ Lakhal Discovering Frequent Closed Itemsets for Association Rules

[13] Dao-I Lin, Zvi M. Kedem. Pincer Search: A New Algorithm for Discovering the Maximum Frequent Set, in Advances in Database Technology - EDBT'98, 6th International Conference on Extending Database Technology, 1998, pages 105-119, 1998.

[14] Mohammed J. Zaki y Mining Closed & Maximal Frequent Itemsets Computer Science Department Rensselaer Polytechnic Institute Troy NY 12180 USA, November 16, 2003.

[15] Ansel Y. Rodríguez-González a,∗ , Fernando Lezamaa , Carlos A. Iglesias-Alvarez b , José Fco. Martínez-Trinidada , Jesús A. Carrasco-Ochoaa , Enrique Munoz de Cotea, Closed frequent similar pattern mining: Reducing the number of frequent similar patterns without information loss, Expert Systems With Applications, 96 (2018) 271–283.

 DOI : 10.18231/2454-9150.2018.0616