ISSN No. 0976-5697

Volume 3, No. 1, Jan-Feb 2012

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

Performance Analysis of Frequent Closed Itemset Mining: PEPP Scalability over CHARM, CLOSET+ and BIDE

Kalli Srinivasa Nageswara Prasad*
Research Scholar in Computer Science
Sri Venkateswara University, Tirupati
Andhra Pradesh, India
kallisnprasad@gmail.com

Prof. S. Ramakrishna
Department of Computer Science
Sri Venkateswara University, Tirupati
Andhra Pradesh, India
profsramakrishnasvu@gmail.com

Abstract: Frequent itemsets play an essential role in association rule mining, The closed frequent itemset mining is an advancement to frequent itemset mining for association rules, which become an interesting topic for researchers. In this paper, we present an empirical study to evaluate the performance compatibility of the Frequent closed itemset mining algorithms that are in current state of the art for mining association rules.

Keywords: Frequent Closed Itemset Mining, CHARM, CLOSET+, BIDE, PEPP

I. INTRODUCTION

In the operation of association rule mining, frequent itemset mining is an important stage that has been aimed at and in which remarkable improvements have been made. The research varies from efficient and scalable algorithms to most research frontiers; including sequential, structured, correlative mining, associative classification and frequent itemset based clustering. Let us discuss present status of this step including the analyzed challenges.

Frequent itemsets are item sets or substructures which occur in a dataset more than specified minimum no. of times. A substructure can be a sub-graph or sub-tree. If such substructure occurs more than specified threshold, it is called a frequent structural itemset. Identifying frequent itemsets is important in mining associations and correlations. It contributed in data indexing, classification and clustering. It is proposed by Agrawal et al. [1] for market basket analysis which explores customer characteristics from the associations between objects in the basket. There are several proposed algorithms [2,3,4] for generating frequent item sets which vary in the way of traversing item set lattice, use of anti-monotone property and the way to handle database. Based on these variations, representative set of algorithms is explained.

Problem Definition: The task of discovering all frequent itemsets is quite challenging. The search space is exponential in the number of items occurring in the database. The support threshold limits the output to a hopefully reasonable subspace. Also, such databases could be massive, containing millions of transactions, making support counting a tough problem. The search space of all itemsets 2^i contains different itemsets. If 'i' is large enough, then the naive approach to generate and count the supports of all itemsets over the database can't be achieved within a reasonable period of time. To compute the supports of a collection of itemsets, we need to access the database. Since such databases tend to be very large, it is not always possible to store them into main memory. Hence the frequent itemset mining algorithm performance will be analyzed based on its ability of search and memory usage.

In section II explored the frequent closed itemset mining algorithms that are targeted to evaluate the performance. In section III the Dataset adoption explained. In section IV, results gained from a comparative study briefed and fallowed by conclusion of the study.

II. EXPLORATION OF THE ALGORITHMS OPTED FOR PERFORMANCE EVALUATION

A. *CHARM [5]:*

CHARM simultaneously explores both the itemset space and tidset space using the IT-tree, unlike previous methods which typically exploit only the itemset space. CHARM uses a novel search method, based on the IT-pair properties, that skips many levels in the IT-tree to quickly converge on the itemset closures, rather than having to enumerate many possible subsets. The pseudo-code for CHARM appears in Figure 5. The algorithm starts by initializing the prefix class [P], of nodes to be examined, to the frequent single items and their tidsets. We assume that the elements in [P] are ordered according to a suitable total order f. The main computation is performed in CHARM-Extend which returns the set of closed frequent itemsets C. CHARM-Extend is responsible for considering each combination of IT-pairs appearing in the prefix class [P]. For each IT-pair Xi × t(Xi) it combines with the IT-pairs Xj × t(Xj) . Each Xi generates a new prefix class [Pi] which is initially empty. The two IT-pairs are combined to produce a new pair X × Y, where X = Xi U Xj and Y = t(Xi) ∩ t(Xj). Then tests which of the four IT-pair properties can be applied by calling CHARM-Property. Note that this routine may modify the current class [P] by deleting IT-pairs that are already subsumed by other pairs. It also inserts the newly generated IT-pairs in the new class [Pi]. Once all Xj have been processed, then recursively explore the new class [Pi] in a depth-first manner. Then insert the itemset X, an extension of Xi, in the set of closed itemsets C , provided that X is not subsumed by a previously found closed set. At this stage any closed itemset containing Xi has already been generated. And then continues to process the next IT-pair in [P].

Charm Algorithm pseudo representation

```
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 f
21. else if t(Xi) = t(Xj) then //Property 4
22. Add X × Y to [Pi] //use orderin
```

B. CLOSET+ [6]:

CLOSET+ follows the popular divide-and-conquer paradigm and the depth-first search strategy which has been verified a winner for mining long patterns by several efficient frequent pattern mining algorithms. It uses FP-tree as the compression technique. A depth-first search and horizontal format-based method like CLOSET+ will compute the local frequent items of a certain prefix by building and scanning its projected database. Therefore, a hybrid tree-projection method will be introduced to improve the space efficiency. Unlike frequent itemset mining, during the closed itemset mining process there may exist some prefix itemsets that are unpromising to be used to grow closed itemsets. We should detect and remove such unpromising prefix itemsets as quickly as possible. Besides adopting the above mentioned item merging and sub-itemset pruning methods, CLOSET+ also having the item skipping technique to further prune search space and speed up mining.

a. ***Bottom up Physical Tree Projection:*** For dense datasets, their FP-trees can be hundreds (or even thousands) times smaller than their corresponding original datasets due to compression. Its conditional projected FP-trees are usually very compact as well. Each projected FP-tree is much smaller than the original FP-tree, and mining on such a compact structure will also be efficient. As a result, for dense datasets CLOSET+ still physically builds projected FP-trees and it is done recursively in a bottom-up manner.

b. ***Top Down Pseudo Tree Projection:*** Physically projecting FP-trees will introduce some over- head both in space usage and runtime (due to allocating and freeing memory for projected FP-trees), especially if the dataset is sparse, the projected FP-tree will not be very compact and does not shrink quickly. Instead of physically building projected FP-trees, a new method is developed for sparse datasets: top-down pseudo projection of FP-tree. Un- like bottom-up physical projection of FP-trees, the pseudo projection is done in the f list order (i.e., in support descending order). Similar to bottom-up physical projection, a header table is also used to record enough information such as local frequent items, their counts and side-link pointers to FP-tree nodes in order to locate the sub trees for a certain prefix item- set.

C. BIDE:

BIDE, an algorithm for discovering the complete set of frequent closed sequences. The contributions of this algorithm include: (1) A new paradigm for mining closed sequences without candidate maintenance, called Bidirectional Extension. The forward directional extension is used to grow the prefix patterns and also check the closure of prefix patterns, while the backward directional extension can be used to both check closure of a prefix pattern and prune the search space. (2) Under the BI-Directional Extension paradigm, opted to frequent closed sequence mining algorithm, BIDE. The BI-Directional Extension pattern closure checking scheme, the BackScan pruning method, and the ScanSkip optimization technique are proposed to speed up the mining and also assure the correctness of the algorithm.

D. PEPP [7]:

The algorithm PEPP [7] opt to Sequence Graph protruding that based on edge projection and pruning, an asymmetric parallel algorithm for finding the set of frequent closed sequences. The giving of this PEPP [7] is: (A) an improved sequence graph based idea is generated for mining closed sequences without candidate maintenance, termed as Parallel Edge Projection and pruning (PEPP) based Sequence Graph Protruding for closed itemset mining. The Edge Projection is a forward approach grows till edge with required support is possible during that time the edges will be pruned. During this pruning process vertices of the edge that differs in support with next edge projected will be considered as closed itemset, also the sequence of vertices that connected by edges with similar support and no projection possible also be considered as closed itemset (B) in the Edge Projection and pruning based Sequence Graph Protruding for closed itemset mining, PEPP contains Forward edge projection and back edge pruning algorithms.

As a first stage PEPP performs dataset preprocessing and itemsets Database initialization, which finds itemsets with single element, in parallel prunes itemsets with single element those contains total support less than required support.

E. Forward Edge Projection:

In this phase, PEPP selects all itemsets from given itemset database as input in parallel, then starts projecting edges from each selected itemset to all possible elements. The first iteration includes the pruning process in parallel, from second iteration onwards this pruning is not required, which we claimed as an efficient process compared to other similar techniques like BIDE.

III. COMPARATIVE STUDY

This section explores the performance compatibility of CHARM [5], CLOSET+ [6], BIDE [7] and PEPP [8]. All the experiments were performed on CPU: Intel E6850 with MOTHERBOARD: nvidia 680i chipset based with SLi, RAM: 4GB(2 x 2GB running in dual channel mode) 800Mhz DDR2 and HARDDRIVE: maxtor raptor 74GB 10,000RPM + 500GB western digital. We ran the four algorithms on the same environment. Because our performance study showed that other reputed algorithms such as OP [9] and CLOSET [10] cannot compete in high memory usage with algorithms [5, 6, 7, 8] those we opted, hence we compared only CHARM [5], CLOSET+ [6], BIDE [7] and PEPP [8] on peak memory usage.

IV. EXPERIMENTAL SETUP

Our performance study includes both synthetic and real datasets. We used couple of synthetic datasets that are generated using IBM-DG[11] and a real dataset Gazelle contains click stream. In Gazelle, we consider different products as different items and the page views as events. The characteristics of these datasets are shown in table 1.

Table1: Structure of datasets opted for performance check

Dataset Name	No. Seq	No. Items	avg. Seq. Len	Max. Seq. Len
SD1	150000	6532	42	61
SD2	150000	5092	71	114
GAZELLE	2,937	1,423	29	1,443

V. RESULTS ANALYSIS

To verify the performance on dense dataset we opt to synthetic dataset SD1 Figure1, table 2, Figure 2 and table 3 explores the performance of the mining algorithms [5,6,7,8] opted. Fig. 2 shows PEPP can be orders of magnitude faster than BIDE and other two algorithms CHARM and CLOSET+. When the support is not too low, CHARM, CLOSET+, BIDE and PEPP performed in similar way. When support threshold 10% and lower CHARM cannot run by reporting memory issues. The rest three algorithms CLOSET+, BIDE and PEPP performance is scalable up to the support value 4% , At the support lower than 4% PEPP and BIDE maintains their scalability, but the CLOSET+ took huge execution time for support value 4% and lower. The PEPP elevated as best in time taken for execution. In terms of memory usage CHARM is very poor. From Fig. 1 we can observe that BIDE and PEPP uses less memory than CLOSET+ For example, at support 85%, CLOSET+ consumes about 190MB while BIDE and PEPP consumes about 18MB.

Table 2: Memory used by algorithms on SD1 for various support values in %

Memory usage on SD1							
Support in %	0.1	0.08	0.06	0.04	0.03	0.02	0.01
CHARM	101	136	169	201	232	-	-
CLOSET +	74	81	99	126	155	181	201
BIDE	5.4	5.9	6.7	7.1	7.9	8.3	9.8
PEPP	5.6	6.1	6.9	7.7	8.3	9.1	10.4

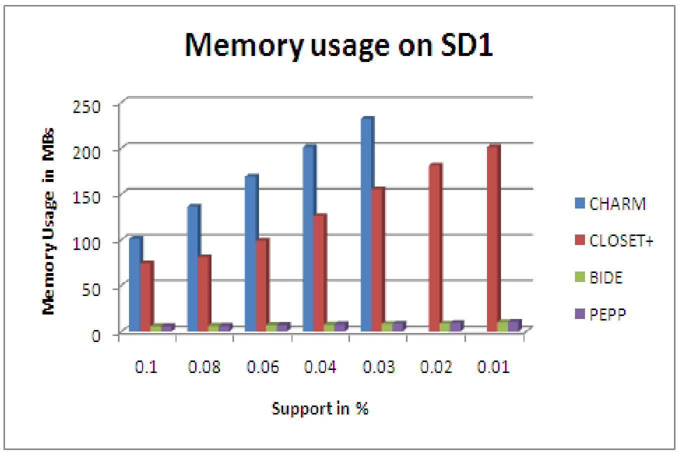

Figure 1: Graph representation of Memory used by algorithms on SD1 for various support values in %

Table 3: Time in seconds taken by algorithms to execute on SD1

Execution Time on SD1							
Support in %	0.1	0.08	0.06	0.04	0.03	0.02	0.01
CHARM	5	9	21	-	-	-	-
CLOSET+	5.9	9.4	23	32	79	92	341
BIDE	5.4	6.1	14	24	39	43	106
PEPP	2.3	3.2	8.6	11	18	21	49

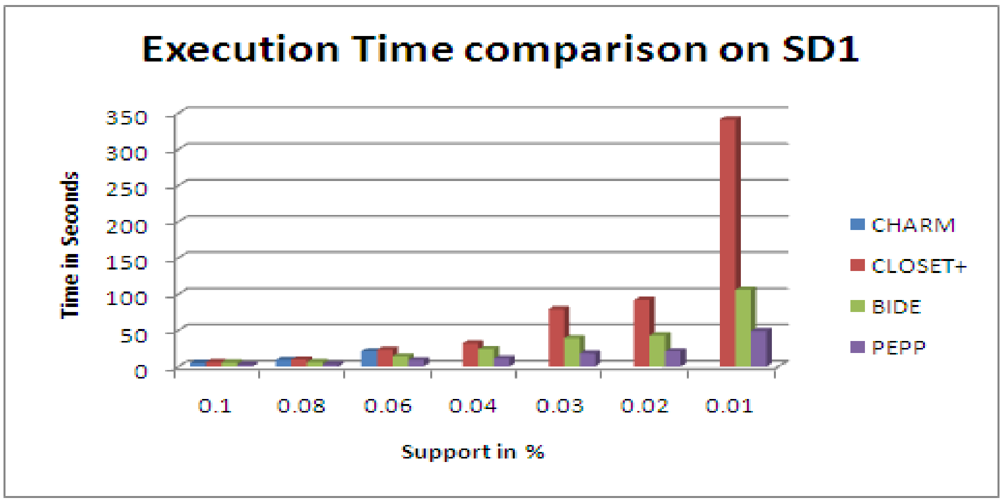

Figure 2: Graph representation of Time in seconds taken by algorithms to execute on SD1

To verify the performance on sparse dataset we opt to real time dataset GAZELLE. Figure3, table 4, Figure 4 and table 5 explores the performance in execution time and memory usage of the mining algorithms[5,6,7,8] opted. Fig. 3 shows PEPP can be orders of magnitude faster than BIDE and other two algorithms CHARM and CLOSET+. When the support is not too low, CHARM, CLOSET+, BIDE and PEPP performed in similar way. When support threshold 10% and lower CHARM cannot run by reporting memory issues. The rest three algorithms CLOSET+, BIDE and PEPP performance is scalable up to the support value 4% , At the support lower than 4% PEPP and BIDE maintains their scalability, but the CLOSET+ took huge execution time for support value 4% and lower. The PEPP elevated as best in time taken for execution. In terms of memory usage CHARM is very poor. From Fig. 4 we can observe that BIDE and PEPP uses less memory than CLOSET+.

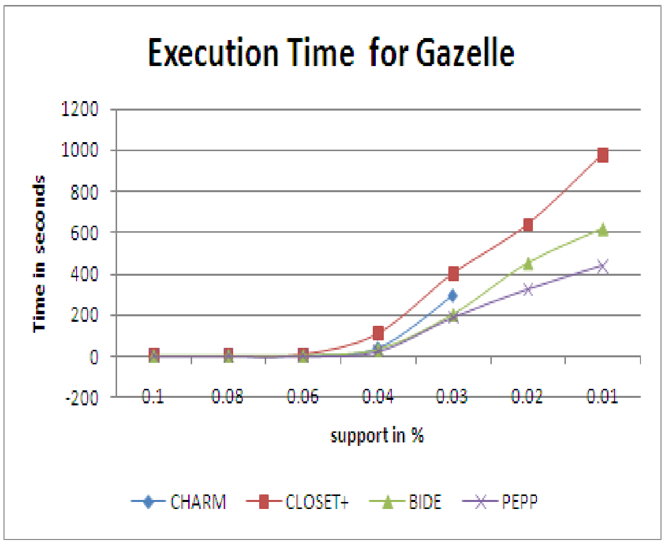

Figure 3: Graph representation of time in seconds taken by algorithms to execute on GAZELLE

Table 4: Time in seconds taken by algorithms to execute on GAZELLE

Execution Time on gazelle							
Support in %	0.1	0.08	0.06	0.04	0.03	0.02	0.01
CHARM	0.5	0.9	3	40	298		
CLOSET+	0.6	1	8	110	401	640	980
BIDE	0.4	0.8	1.9	31	201	452	621
PEPP	0.4	0.75	0.87	24	189	326	441

Table 5: Memory used by algorithms on GAZELLE for various support values in %

MEMORY USAGE on GAZELLE							
SUPPORT in %	0.1	0.08	0.06	0.04	0.03	0.02	0.01
CHARM	21	32	98	115	120	-	-
CLOSET+	11.2	13.4	14.1	18.4	19.3	21.1	24.9
BIDE	4.4	4.9	5.9	6.4	7.1	9.3	10.8
PEPP	3.3	3.8	4.2	4.8	5.1	5.7	6.1

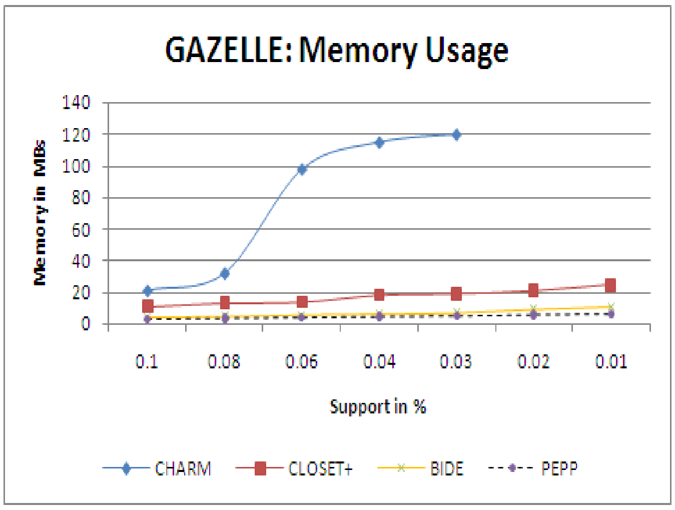

Figure 4: Graph representation of Memory used by algorithms on GAZELLE for various support values in %

We used the synthetic datasets SD1 and SD2 those generated by IBM-DG to test the scalability of CLOSET+ and compared it with both CHARM and CLOSET. We first tested the scalability in terms of database size using the dataset series SD1 with base size from 2000 tuple to 15000 tuple and support threshold set at 0.008%. From Figure 5 and table 6, we can see that, CHARM has the poorest scalability, and it even become worsen when the dataset contains more than 600K tuple. In comparison between CLOSET+, BIDE and PEPP, PEPP runs much faster and it also has much better scalability in terms of base size: the slope ratio for CHARM is much higher than other three and PEPP and BIDE are much scalable that compared to CLOSET+.

The scalability in terms of number of distinct items using SD2 series with number of distinct items set at 5092, 15845, 27550 and 31169, respectively, and minimum support set at 0.005%. From Figure 6 and table 7. , we can see that initially all algorithms have very similar performance when the number of distinct items is small, but once the number of distinct items increases, the runtime of CHARM has much bigger jump than CLOSET+, BIDE and PEPP. Out of CLOSET+, BIDE and PEPP, the PEPP run time is much scalable, which means PEPP also has better scalability than CHARM, CLOSET+ and BIDE in terms of the number of distinct items.

Table 6: Time taken by algorithms on SD1 for various tuple sizes

Scalability on SD1							
Base size in tuples(*10)	200	400	600	800	1000	1200	1500
CHARM	101	198	267	352	432	521	701
CLOSET+	94	118	123	144	156	164	181
BIDE	34	39	45	59	64	73	79
PEPP	18	23	28	32	41	59	71

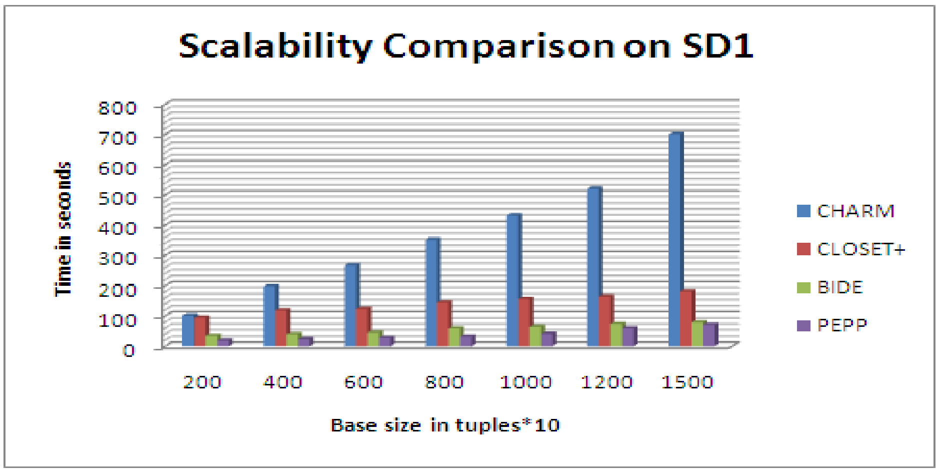

Figure 5: graph representation of Time taken by algorithms on SD1 for various tuple sizes

Table 7: Time taken by algorithms on SD2 for various count of distinct elements

Scalability on SD2							
Unique item count	1000	5000	10000	15000	20000	25000	30000
CHARM	18.2	23.2	40.1	56.4	67.3	79.1	91.2
CLOSET+	16.1	16.3	16.9	17.5	18.1	18.9	19.7
BIDE	14.3	14.7	15.1	15.7	16.2	16.8	17.1
PEPP	14.1	14.2	14.8	15.1	15.7	16.1	16.4

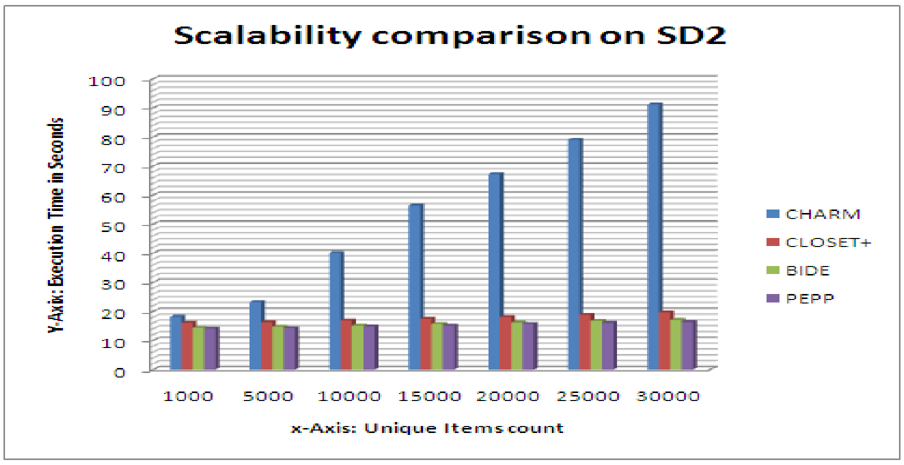

Figure 6: Graph representation of time taken by algorithms on SD2 for various count of distinct elements

VI. CONCLUSION

Frequent pattern mining has been studied extensively in data mining research. In this study, we have re-examined some previously proposed methodologies, and mainly focused on the new technique PEPP[8] developed for frequent closed itemset mining, a highly scalable and both runtime and space efficient algorithm for dense and sparse datasets, on different data distributions and support thresholds. In this paper we conducted empirical analysis of the performance differences between CHARM[5], CLOSET+[6], BIDE[7] and PEPP[8]. The aim of this comparative is to prove the performance in speed, memory usage and scalability of PEPP over CHARM, CLOSET+ and BIDE.

VII. REFERENCES

[1]. Agrawal, R., T, Imielinski and A, Swami, 1993, Mining association rules between sets of items in large databases, Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, May 25-28, ACM, New York, USA., pp: 207-216

[2]. Agrawal, R, and R, Srikant, 1994, Fast algorithms for mining association rules, Proceedings of the 20th International Conference on Very Large Data Bases, Sept, 12-15, San Francisco, CA., USA., pp: 487-499

[3]. Mannila, H., H, Toivonen and A, Inkeri Verkamo, 1994 Efficient algorithms for discovering association rules Proceedings of the AAAI Workshop on Knowledge Discovery in Databases, (KDD-94), IEEE, pp: 181-192

[4]. Han, J., J, Pei, Y, Yin and R, Mao, 2004, Mining frequent patterns without candidate generation: A frequent-pattern tree approach, Data Mining Knowledge Discovery, 8: 53-87

[5]. M. Zaki, and C. Hsiao, CHARM: An efficient algorithm for closed itemset mining. In SDM'02, Arlington, VA, April 2002.

[6]. J. Wang, J. Han, and J. Pei, CLOSET+: Searching for the Best Strategies for Mining Frequent Closed Itemsets. In KDD'03, Washington, DC, Aug. 2003.

[7]. J. Wang and J. Han. BIDE: Efficient mining of frequent closed sequences. In Proc. 20th International Conference on Data Engineering, pages 79–90, 2004.

[8]. Parallel Edge Projection and Pruning (PEPP) Based Sequence Graph protrude approach for Closed Itemset Mining kalli Srinivasa Nageswara Prasad, Sri Venkateswara University, Tirupati, Andhra Pradesh , India. Prof. S. Ramakrishna, Department of Computer Science, Sri Venkateswara University, Tirupati, Andhra Pradesh, India Vol. 9 No. 9 September 2011 International Journal of Computer Science and Information Security Publication September 2011, Volume 9 No. 9

[9]. J. Liu, Y. Pan, K. Wang, and J. Han. Mining frequent item sets by opportunistic projection. In SIGKDD'02, July 2002.

[10]. J. Pei, J. Han, and R. Mao. CLOSET: An efficient algorithm for mining frequent closed itemsets. In DMKD'00, May 2000.

[11]. IBM dataset generator for sequential patterns. http://www.almaden.ibm.com/software/quest/Resources.

Short Bio Data for the Author's

Kalli Srinivasa Nageswara Prasad has completed M.Sc(Tech)., M.Sc., M.S (Software Systems)., P.G.D.C.S. He is currently pursuing Ph.D degree in the field of Data Mining at Sri Venkateswara University, Tirupathi, Andhra Pradesh State, India. He has published Ten Research papers in International journals. He has also attend Three national conferences.

S.Ramakrishna is currently working as a professor in the Department of Computer Science, College of Commerce, Management & Computer Sciences in Sri Venkateswara university, Tirupathi, Andhra Pradesh State, India. He has completed M.Sc, M.Phil., Ph.D., M.Tech(IT).He is specialized in Fluid Dynamics & Theoretical Computer Science. His area of Research includes Artificial Intelligence, Data Mining & Computer Networks. He has an experience of 26 years in teaching field. He has published 41 Research papers in National & International Journals. He has also attended 13 national Conferences and 3 International Conferences. He has guided 17 Ph.D Scholars and 18 M.Phil Scholars.