High Performance Frequent Pattern Mining on Multi-Core Cluster

Lan Vu and Gita Alaghband
Department of Computer Science and Engineering
University of Colorado Denver
Denver, USA
Lan.Vu@ucdenver.edu
Gita.Alaghband@ucdenver.edu

POSTER PAPER

***Abstract*— Mining frequent patterns is a fundamental data mining task with numerous practical applications such as consumer market-basket analysis, web mining, and network intrusion detection. When database size is large, executing this mining task on a personal computer is non-trivial because of huge computational time and memory consumption. In our previous research, we proposed a novel algorithm named FEM which is more efficient than well-known algorithms like Apriori, Eclat or FP-growth in discovering frequent patterns from both dense and sparse databases. However, in order to apply FEM to applications with large-scale databases, it is essential to develop new parallel algorithms that are based on FEM and deploy this mining task on high performance computer systems. In this paper, we present a new method named PFEM that parallelizes the FEM algorithm for a cluster of multi-core machines. Our proposed method allows each machine in the cluster execute an independent mining workload to improve the scalability. Computations within a multi-core machine use shared memory model to reduce communication overhead and maintain load balance. With the collaboration of both distributed memory and shared memory computational models, PFEM can adapt well to large computer systems with many multi-core.**

Keyword – data mining; frequent pattern mining; association rule mining; multi-core cluster; parallel algorithm; transactional databases.

I. INTRODUCTION

Frequent pattern mining is an important problem in data mining which is aimed to search for groups of itemsets, subsequences, or substructures that co-occur in a database with their frequency no less than a user-specified minimum support threshold. This mining task can be used to discover many types of relationships in large databases such as associations [1], correlations [2], causality [3], sequential patterns [4], episodes [5] and partial periodicity [6]. In addition to its numerous practical applications, it is also applied in data indexing, classification, clustering, especially association rule mining as well [7], [8].

Motivation: Although frequent pattern mining has a simple computational model, this task is computationally intensive, I/O intensive, and requires large computing resources especially memory [9]. In our previous research [27], [28], we proposed a novel approach for frequent pattern mining that combines mining strategies of two well-known algorithms Eclat [10] and FP-growth [11]. The FEM algorithm developed from this mining approach performs better than many popular algorithms like Apriori [1], Eclat or FP-growth on both dense and sparse databases. However, our experiments show that mining on very large databases (e.g. web document databases or integrated biological databases) requires parallel frequent pattern mining methods to efficiently utilize computing resources of large high performance computer systems such as clusters.

For over a decade, many parallel and distributed algorithms have been proposed [8], [23], [30], [31]. However, most of these methods were developed for shared memory systems or distributed memory systems alone. Clusters of multi-core machines is the current trend in high performance computing which requires new algorithm and system design that can take advantage of both the shared and distributed memory environments. Although clusters offer a much higher computing power, they pose major challenges in design and implementation of efficient high performance algorithms. The goal here is to design an efficient and fast parallel and distributed frequent pattern mining task for large-scale applications.

Contribution:

In summary, the contributions of this paper are as follows:

(1) We propose PFEM, a parallel method based on FEM [27] for large-scale frequent pattern mining on multi-core clusters. PFEM can improve the scalability by distributing independent mining workload over the multi-core machines of the cluster and utilizing shared memory computational model to reduce communication overhead as well as maintain the balance of the workload.
(2) Because of the similarities of FEM and FP-growth, our proposed method of combining distributed memory and

978-1-4673-1382-7/12/$31.00 ©2012 IEEE

shared memory computational models can be also applied to parallelize FP-growth-like algorithms [11].

II. Overview of The FEM Algorithm

A. Frequent Pattern Mining Problem

The frequent pattern mining problem aims to search for groups of itemsets, subsequences, or substructures that co-occur in a database with their frequency no less than a user-specified minimum support threshold. For example, a set of items (itemset), such as milk and bread that appear frequently together in a database is a frequent itemset or frequent pattern. In a typical transactional database, the number of distinct single items and their combinations are usually very large. For a small minimum support threshold, the number of generated itemsets can be extremely large. Hence, it is a great challenge to design algorithms for mining frequent patterns that scale with memory size and run in reasonable time [9].

B. FEM: An Adaptive Method for Mining Frequent Patterns from Dense and Sparse Databases

Many algorithms have been proposed for frequent pattern mining [1], [10], [11], [12], [13], [14], [15], [16], [17], [18], [19], [20], [21], [22], [23], [24], [25], [26]. However, most of them behave differently on different databases making it difficult for users to select a suitable method for their applications. To address this issue, we have developed a new algorithm called FEM that combines mining strategies of two well-know algorithms Eclat and FP-growth to unify their merits and adapt mining behavior to characteristics of databases. Our experimental results show that FEM performs efficiently on both dense and sparse databases [27], [28]. The FEM algorithm includes three main tasks:

- *FP-tree construction*: Database is scanned for the first time to find the frequent items and create the header table. A second database scan is conducted to get frequent items of each transaction. Then, these items are inserted into the FP-tree in their frequency descending order.
- *FP-tree mining*: This task uses the mining solution of FP-growth to construct the conditional FP-trees and recursively mines these trees to find the frequent patterns. However, before a conditional FP-tree is constructed, it will check the size of the appropriate conditional pattern base. If its size is smaller or equal to a threshold K (e.g. K=128), the conditional pattern base will be transformed into TID bit vectors and a weight vector and the mining process switches to the TID-bit-vector mining task.
- *TID-bit-vector mining*: This task obtains the TID bit vectors and continues searching for frequent patterns recursively by logical ANDing these bit vectors. The new patterns are constructed by concatenating the suffix pattern of previous steps with the newly generated frequent patterns. This mining task is inspired by Eclat's mining strategy. However, TID bit vectors are used instead of the TID-lists for their efficiency as shown in [8].

III. PFEM: Parallel FEM For Multi-Core Cluster

A. Multi-core Cluster Architecture

A typical multi-core cluster is usually a network of many multi-core machines (nodes) with Gigabit Ethernet/Infiniband interconnection. Each node can have one or many multi-core chips which share main memory. Cores on a single chip can have private and shared caches. Figure 1 shows an example architecture of a typical dual six-core cluster in which two chips of a node share main memory; six cores of a chip share L3 cache and each core has private L1 and L2 caches. Communication latency among cores on the same chips is smaller than among cores on different chips because of the impact of shared cache. Communication latency between cores on different nodes is largest because of large latency of interconnection network among the nodes.

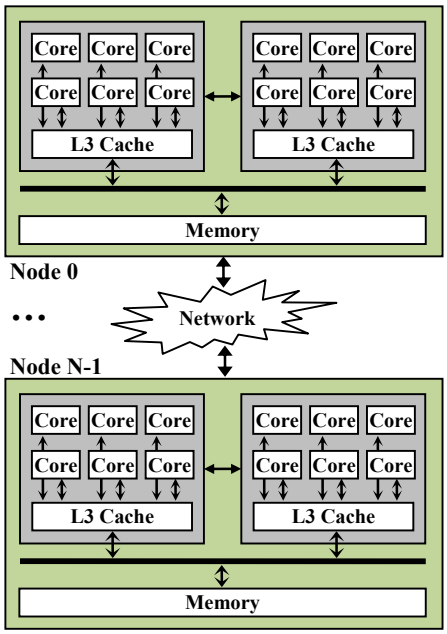

Figure 1: Architecture of a Typical Cluster with Dual Six-core nodes

B. Challenges in Developing a Parallel Algorithm bBased on FEM

Scalability and load balancing are two major desirable characteristics of a parallel frequent pattern mining algorithm used in large-scale mining applications. In this research, we aim to develop a parallel algorithm for FEM that runs fast and scales well on multi-core clusters. There are several challenges to be solved in order to obtain high scalability and speedup [9].

- *Memory consumption*. For large databases or mining with very low minimum support, the corresponding FP-trees are very large and may not fit in main memory. Hence, it is necessary to partition the database into smaller ones that fit in the main memory of each of the nodes.
- *Data communication*: parallel frequent pattern mining algorithms like FEM usually requires huge communication and synchronization to construct FP-trees.

- *Load balancing*: FEM works in a depth-first manner with recursive computational model. Thus, load balancing is important to avoid unfair computational distribution.
- *Other elements* like cache architecture and I/O utilization can affect the program performance and scalability that also needs to be considered.

In addition, our observations on many experimental results conducted on our cluster [29] show that a parallel algorithm that maximizes the use of shared memory, both main memory and the shared caches, will minimize the communication overhead and result in better speedup and scalability.

C. PFEM: Design of Parallel FEM For Multi-Core Cluster

PFEM is designed with consideration of challenges presented in previous section. Its computational model combines features of both distributed memory and shared memory systems where communication among nodes in the cluster will use message passing and communication among cores in a node is done via shared memory. In the context of this paper, we use *core* to indicate thread/process running on one core of a node.

PFEM divides mining workload into M parts where frequent pattern generation of each part is independent. M should be much larger than the number of nodes in the cluster. Data and computation of M parts will be distributed to nodes in a dynamic manner to improve load balance. Cores in a same node share same memory space and working data. Each core will generate frequent patterns from shared data and store them in its own local file. Finally, all frequent pattern sets are combined for final mining results.

The detailed design of PFEM consists of two stages:

Stage 1 *- Parallel FP-tree construction*:

This stage executes the *FP-tree construction* task of FEM.

1. Create a local copy of database on each node in the cluster.
2. Each core in the cluster scans its data partition to compute local counts of all items. Then, global counts of all items are computed using all local count values.
3. Frequent items are specified and grouped into M groups. M is large enough for good load balancing.
4. Each node will be assigned a group of frequent items to construct a set of FP-trees where each tree is associated with an item in the group. There are two ways to do this task: (1) each core reads the whole database and create FP-trees of one or several item in the group; (2) all cores in a node read their portions of the database to construct a set of FP-trees where each tree is associated with one item in the group.
5. When all FP-trees of items in the group are constructed, the mining process starts by switching to Stage 2 to find all frequent patterns from these FP-trees.

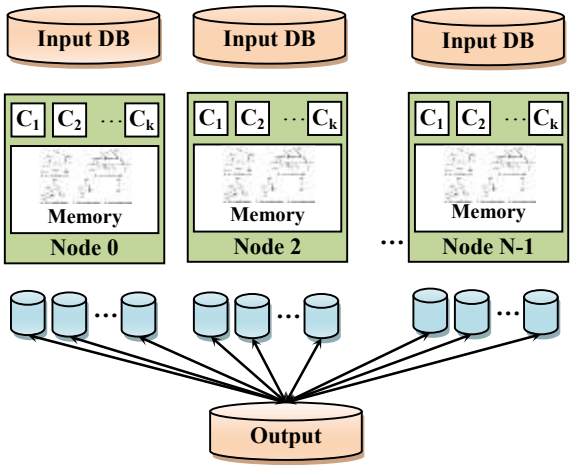

Figure 2: Overview computational model of PFEM

Stage 2 *- Parallel Frequent Pattern Generation*

This stage executes two tasks *FP-tree mining* and *TID-bit-vector mining* of FEM.

1. Each node creates a queue that contains all FP-trees in the set of FP-trees generated in Stage 1.
2. Each core in a node obtains a FP-tree from this queue and starts mining using FEM approach. Frequent patterns generated by each core are stored in its local output file.
3. During the recursive mining process, new conditional FP-trees are generated and added to the queue. When all frequent patterns of the FP-tree set are found, mining process of the node will stop.
4. The algorithm will check if another group of frequent items created in Stage 1 is available. If there is an unprocessed group, the algorithm will switch to Step 3 of Stage 1 to work on this group.
5. When all frequent patterns are found and the frequent pattern generation completes, all local output files will be combined into a global output file containing the datasets.

IV. CONCLUSION

In this paper, we present PFEM, a parallel version of the FEM algorithm for mining frequent pattern on multi-core clusters. By addressing the computing resource challenges, PFEM can solve the computation and memory bottleneck as well as minimize the data communication and automatically balance the workload. This algorithm is expected to scale well and result in fast performance on the parallel systems. The proposed method of combining distributed memory and shared memory computational model can be also applied to parallelize FP-growth like algorithms and other related problem.

REFERENCES

[1] R. Agrawal, R. Srikant, "Fast Algorithms for Mining Association Rules," *Proc. of the 20th Int. Conf. on Very Large Databases*, pp. 487-499, 1994.

[2] S. Brin, R. Motwani, C. Silverstein, "Beyond Market Basket: Generalizing Association Rules to Correlations," *Proc. ACM SIGMOD Management of Data*, vol. 26, issue 2, pp. 265-276, Jun. 1997.

[3] C. Silverstein, S. Brin, R. Motwani, J. Ullman, "Scalable Techniques for Mining Causal Structures," *J. Data Mining and Knowledge Discovery*, vol. 4, issue 2-3, pp. 163–192, July 2000.

[4] R. Agrawal, R. Srikant, "Mining Sequential Patterns," *Proc. Data Engineering*, pp. 3–14, 1995.

[5] H. Mannila, H. Toivonen, and A. I. Verkamo, "Discovery of Frequent Episodes in Event Sequences," *J. Data Mining and Knowledge Discovery*, vol. 1, issue 3, pp. 259-289, Sep. 1997.

[6] J. Han, G. Dong, Y. Yin, "Efficient Mining of Partial Periodic Patterns in Time Series Database," *Proc. IEEE Data Engineering*, pp. 106-115, Mar. 1999, doi: 10.1109/ICDE.1999.754913.

[7] J. Han, H. Cheng, D. Xin, X. Yan, "Frequent Pattern Mining: Current Status and Future Directions," *J. Data Mining and Knowledge Discovery*, vol. 15, issue 1, pp. 55-86, Aug. 2007.

[8] D. Burdick, M. Calimlim, J. Flannick, J. Gehrke, T. Yiu, "MAFIA: A Maximal Frequent Itemset Algorithm," *IEEE Trans. on Knowledge and Data Engineering*, vol. 17, no. 11, pp. 1490–1504, Nov. 2005, doi: 10.1109/TKDE.2005.183

[9] H. Li, Y. Wang, D. Zhang, M. Zhang, E. Chang, "PFP: Parallel FP-Growth for Query Recommendation," *Proc. 2008 ACM Recommender systems*, pp. 107-114, 2008.

[10] M. Zaki, S. Parthasarathy, M. Ogihara, W. Li, "New algorithms for fast discovery of association rules," *Proc. Knowledge Discovery and Data Mining*, pp. 283-286, 1997.

[11] J. Han, J. Pei, Y. Yin, "Mining Frequent Patterns without Candidate Generation," *Proc. Management of Data*, vol. 29, issue 2, pp. 1-12, Jun. 2000.

[12] JS. Park, MS. Chen, P. Yu, "An Effective Hash-based Algorithm for Mining Association Rules," *Proc. ACM SIGMOD Management of Data*, vol. 24, issue 2, pp. 175–186, May 1995.

[13] H. Toivonen, "Sampling Large Databases for Association Rules," *Proc. Very Large Databases*, pp. 134–145, 1996.

[14] S. Brin, R. Motwani, JD. Ullman, S. Tsur, "Dynamic Itemset Counting and Implication Rules for Market Basket Analysis," *Proc. ACM SIGMOD Management of Data*, vol. 26, issue 2, pp. 255–264, 1997.

[15] A. Fiat, S. Shporer, "AIM: Another Itemset Miner," *Proc. Frequent Itemset Mining Implementations*, 2003.

[16] M. J. Zaki, K. Gouda, "Fast Vertical Mining Using Diffsets," *Proc. ACM SIGKDD Knowledge Discovery and Data Mining,* pp. 326-335, 2003.

[17] C. Borgelt, "An Implementation of the FP-growth Algorithm," *Proc. OSDM Frequent Pattern Mining Implementations*, Aug. 2005.

[18] G. Grahne, J. Zhu, "Efficiently Using Prefix-trees in Mining Frequent Itemsets," *Proc. Frequent Pattern Mining Implementations*, pp 123–132, 2003.

[19] J. Pei, J. Han, H. Lu, S. Nishio, S. Tang, D. Yang, "Hmine : Hyper-structure Mining of Frequent Patterns in Large Databases," *Proc. IEEE Data Mining*, pp. 441–448, Nov. 2001, doi: 10.1109/ICDM.2001.989550

[20] B. Racz. "nonordfp: An FP-growth Variation Without Rebuilding the FP-tree," *Proc. IEEE ICDM Workshop on Frequent Itemset Mining Implementations*, Nov. 2004.

[21] S. Shporer, "AIM2: Improved Implementation of AIM," *Proc. IEEE Frequent Itemset Mining Implementations*, Nov. 2004.

[22] L. Schmidt-Thieme., "Algorithmic Features of Eclat," *Proc. IEEE Frequent Itemset Mining Implementations*, Nov. 2004.

[23] L. Liu, E. Li , Y. Zhang, Z. Tang, "Optimization of Frequent Itemset Mining on Multiple-core Processor," *Proc. of the 33rd Int. Conf. on Very Large Databases*, pp. 1275-1285, 2007.

[24] W. Li, A. Mozes, "Computing Frequent Itemsets Inside Oracle 10g," *Proc. of the 30th Int. conf. on Very Large Databases*, pp. 1253–1256, 2004.

[25] C. Utley, "Introduction to SQL Server 2005 Data Mining," *Microsoft SQL Server 9.0 technical articles*, available at: http://technet.microsoft.com/en-us/library/ms345131.aspx, Jun. 2005.

[26] T. Yoshizawa, I. Pramudiono, M. Kitsuregawa, "SQL Based Association Rule Mining Using Commercial RDBMS (IBM db2 UBD EEE)," *Proc. Data Warehousing and Knowledge Discovery*, pp. 301–306, 2000.

[27] L. Vu, G. Alaghband, "A Fast Algorithm Combining FP-Tree and TID-List for Frequent Pattern Mining," *Proc. IEEE Information and Knowledge Engineering*, pp. 472-477, Jul. 2011.

[28] L. Vu, G. Alaghband, "Efficient Algorithms Combining FP-Tree and TID-List for Frequent Pattern Mining," submitted *to the IEEE Transaction on Knowledge and Data Engineering in Mar. 2012*.

[29] Parallel and Distributed System Laboratory, University of Colorado Denver, http://pds.ucdenver.edu

[30] M. J. Zaki, "Parallel and Distributed Association Mining: A Survey", *IEEE Concurrency Journal,* vol. 7, issue 4, pp. 14- 45, Oct-Dec 1999.

[31] R. Garg, P. K. Mishra, "Some Observations of Sequential, Parallel and Distributed Association Rule Mining Algorithms", *IEEE Proc. Of the 2009 Int. conf. on Computer and Automation Engineering*, pp. 336-342, 8-10 March 2009.