Parallel Association Rule Algorithm using Hybrid Parallel Computing

Negussie Zadig Serawit

Tianjin University of Technology and Education, Department of Applied Computer Technology
Dagu South Road, Hexi District, Tianjin, PRC - 300222, China

Abstract: ***With massive amount of data continuously being collected and stored, many industries are becoming interested in mining such patterns from their databases. The discovery of interesting correlation relationship among huge amount of business transactions records can help in many business decision-making processes. Simply adding more processors will not magically improve the performance of the vast majority of serial programs. Clearly, parallel computing can have an enormous impact on application performance, and OpenMP and MPI facilitates aces to this enhanced performance. Having them work in a hybrid model will result in a better efficiency supporting each other. As an example this paper is intended to provide an insight of hybrid parallel implementation for mining an association rule called Apriori.***

Keywords: Apriori algorithm, parallel computing, MPI, OpenMP, hybrid parallel computing, parallel Apriori

1. Introduction

Data mining is the way to extract strategic information from data. The interdisciplinary nature of data mining research and development contributes significantly on the success of data mining and its extensive applications. As a highly application-driven domain, data mining has incorporated many techniques from other domains such as statistics, machine learning, pattern recognitions, database and data warehouse system, information retrieval, visualization algorithms, high performance computing, expert systems, and many more application domain.

In general, we use data mining for looking hidden patterns that is not immediately apparent from summarizing the data and predict the future outcome to make better decision. In addition to accuracy, data mining research places strong emphasis on the efficiency and scalability of mining methods on large data set, as well as on ways to handle complex types of data and explore new alternative methods. The idea of data mining is to build models from existing data so that we can predict future events or outcomes and based on those predictions, we can improve our decision making process.

Data mining can be applied to any kind of data as long as the data are meaningful for a target application. The most basic forms of data for mining applications are database data, data warehouse data, and transactional data. It can also be applied to other forms of data such as stream data, ordered or sequential data, graph or networked data, spatial data, text data, multimedia data, and the World Wide Web. Considering all the different type of data, the information gathered in data mining process is categorized into two main categories, descriptive and predictive information; (i) ***descriptive information*** mainly focus on finding pattern that are human interpretable, some of the tasks in this modelling include; clustering, association rule discovery, sequential pattern discovery, etc. and, (ii) ***predictive information*** focuses on finding value of an attribute using values of other attributes. Some of the tasks include; classification, regression, deviation detection …

This paper mainly tries to emphasis on how it is possible to optimize Apriori algorithm which is an association rules discovery modeling by using parallel approach. Many have optimized parallel Apriori algorithm using OpenMP, but much less research done using MPI and hybrid OpenMP-MPI model.

Section two will describe in detail OpenMP, MPI, hybrid OpenMP-MPI, Apriori algorithm. Section three will demonstration the implementation details of the hybrid parallel Apriori algorithm. Section four will clarify application areas of interest that may require the use of huge data processing in relation to data mining. Finally, section five will conclude and arrange a list of some reference materials for further knowledge.

2. Basic Concept

2.1 Apriori algorithm

Apriori is one of the well-known association rule algorithm. It is a seminal algorithm for mining frequent item sets for Boolean association rules. The name of the algorithm is based on the fact that the algorithm rules prior knowledge for frequent item sets properties. It implies an iterative approach known as level-wise search. In general, association rule mining can be viewed as two-step process; (i) finding all frequent item sets: each of these item sets will occur at least as frequently as a predetermined minimum support count, and (ii) generate strong association rule from the frequent item sets: this rules must satisfy minimum support and minimum confidence.

High computation headache in Apriori algorithm is counting the frequent item sets, see fig 1. Data needs to be accessed from storage for processing which makes input/output performance slower. Parallelizing input/output allows applications to access in parallel to the data, providing better performance.

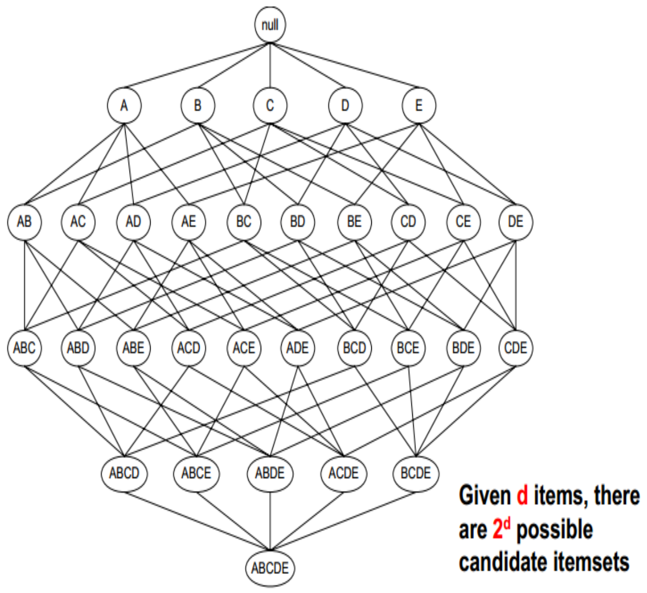

Figure 1: Frequent item set generation

Major challenge is mining frequent item sets from large data sets is the fact that such mining often generates a huge number of item sets satisfying the minimum support threshold, especially, when minimum support is low. This is because, if an item set is frequent, each of its subsets is frequent as well. A long item set will contain a combinatorial number of shorter frequent sub-item sets. Possible rules that can be generated in a given transaction can be calculated as:

$$R = \sum_{k=1}^{d-1}\left[\binom{d}{k} \times \sum_{j=1}^{d-k}\binom{d-k}{j}\right] = 3^d - 2^{d+1} + 1$$

If d =6, R=602 rules

For example, given d, unique item set of size 6, the total possible number of association rules that can be generated, using the above formula, will be 602 rules. Another example with frequent item set of length 100, such as$\{a_1, a_2, \ldots, a_{100}\}$ contains be as follows:

$$\binom{100}{1} + \binom{100}{2} + \cdots + \binom{100}{100} = 2^{100} - 1 \approx 1.27 \times 10^{30} FrequentItemSets$$

This is a huge number for any computer to compute or store, its computational complexity can be shown graphically in Fig 2. Closed frequent item set and maximal frequent item sets concept overcome this difficulty.

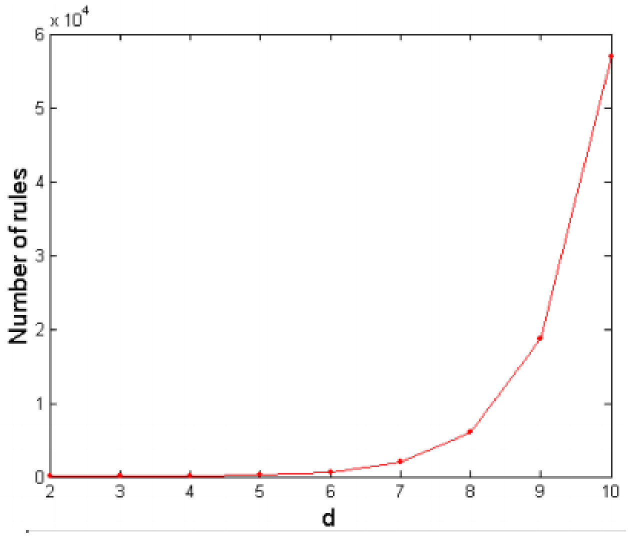

Figure 2: computational complexity, where d stands for unique item sets, total number of item sets will be 2^d on the x-axis and on the y-axis total number of possible association rules

An item X is closed in a data set D if there exists no proper super-item set Y^5 such that Y has the same support count as X in D. An item X is a ***closed frequent item set*** in set D if X is both closed and frequent in D, and it is ***maximal frequent item set*** if X is frequent and there exists no super-item set Y such that $X \subset Y$ and Y is frequent in D. These allow the user to choose how much item sets an association rule need to be generated with.

A candidate rule is generated in Apriori algorithm by merging two rules that share the same prefix, these process is known as join. And, using the support count that havebeen obtained during the frequent item set generation step, if a rule does not have high confidence, it will be cut-off, also known as pruning. These 2 important tasks are not discussed in this paper, because it is believed to be under-consideration all the way throughout the algorithm

2.2 MPI (Message Passing Interface)

MPI stands for Message Passing Interface. It is used for communication among processes that have different address spaces. It is single-threaded process, message passing library, but is neither a language, not a compiler specification nor a specific implementation. It can be implemented in both shared and/or distributed-memory parallel computing environment. In other words, is easily compatible with both distributed-memory multicomputer and shared-memory multiprocessors and the combinations of these elements.

2.3 Open MP (Open Multi-Processing)

It stands for Open Multi-Processing, supports multi-platform shared-memory parallel programming. It consists of compiler directives, library routines, and environmental variables that influence run-time behavior. The core elements of OpenMP are the constructs for thread creation, workload distribution, data environment management, thread synchronization, user level run-time routines and environmental variables. OpenMP makes better use of shared memory architecture, avoiding the overhead of message passing and provides two kinds of parallelism; coarse-granularity and fine-granularity.

2.4 Hybrid OpenMP and MPI

Adding an MPI code to OpenMP threading is an efficient way to run on multi-core processors and nodes. Since OpenMP operates in a shared memory space, and MPI can operate in both distributed and shared-memory architecture, it is possible to reduce the memory overhead associated with MPI tasks and reduce the replicated data across tasks.

To achieve MPI and OpenMP hybrid programming model, high performance computers can make well use of distributed memory high performance architecture. The main advantages of this hybrid programming model are:

- MPI can solve coarse-grained inter-processor communication
- OpenMP can solve the interaction among each of the processors within a multi-processor computers
- Fast intra-node shared memory access instead of message passing to reduce the number of message passing, communication overhead and the execution time

MPI and coarse-grained OpenMP hybrid parallelism is widely used due to the several advantages of coarse-grained OpenMP. These include; direct generation of threads after MPI processes creation and initialization; to communicate between nodes a thread is selected to be on duty for communication while others can continue their task.

3. Implementation

Each node executes a MPI process, and the MPI process contains multiple thread. For efficiency, the algorithm must combine the distributed shared memory system with multi-tier architecture. As the distributed shared memory system is shared within the storage node, and the nodes are connected, we construct a multi-level parallel algorithm, that uses the process-level parallel between nodes, and thread-level parallel in a node, MPI and OpenMP respectively.

The dataset will be controlled by the master node, and slave nodes will send local items to the master which will compute the support constantly. Therefore, there is no need of data exchange between nodes and communication overhead will be reduced.

3.1 Pseudo code

Generally, the pseudo code of Apriori algorithm for transaction database T, and support threshold of S is given below. Here note that C_k is the candidate set for level k. At each stage, it is assumed to generate the candidate sets from the large item sets of the preceding level, heeding the ***downward closure lemma*** $count[c]$ accesses a field of the data structure that represents candidate set c which is initially assumed to be zero. Many details have been omitted, usually the most important part of the implementation is the data structure used for storing the candidate sets, and counting their frequencies. But, this paper tries to show where parallelism can be of best use in the Apriori algorithm.

```
$\boldsymbol{Apriori}(T, S)$
  $L_1 \leftarrow \{large\ 1 - itemsets\}$
  $k \leftarrow 2$
  $\boldsymbol{while} L_{k-1} \neq \emptyset$
    $C_k \leftarrow \left\{a \cup \{b\} \middle| a \in L_{k-1} \wedge b \in \bigcup L_{k-1} \wedge b \notin a\right\}$
    $\boldsymbol{for}\ transactions\ t\ \in T$
      $C_t \leftarrow \{c | c \in C_k \wedge c \subseteq t\}$
      $\boldsymbol{for}\ candidates\ c \in C_t$
        $count[c] \leftarrow count[c] + 1$
    $L_k \leftarrow \{c | c \in C_k \wedge count[c] \geq S\}$
    $k \leftarrow k + 1$
$\textbf{return} \bigcup_k L_k$
```

3.2 Hybrid Apriori Parallel Algorithm

Step 1. Assign dataset to a master processor
Step 2. Divide item sets into n partitions, where n is the number of processors
Step 3. Loop
Step 4. Slave processors send single item to the master
  - Master processors task will loop
  - Master processor receives single items from slave processor
  - Computes frequency in a thread model (OpenMP data decomposition model using fork-join architecture) against dataset
  - Computes support
  - Sends result back to slave processor
  - Until all processors terminate

Step 5. Slave processor receive frequency of that item
Step 6. Each slave processors discard non frequent item sets
Step 7. Build a tree for storing the frequency of the frequent item sets
Step 8. Each processor generate candidate k-item sets
Step 9. Until candidate item sets are empty.

4. Area of interest

As our computational power increases, the number of problems that we can seriously consider solving also increases. The following are few areas of interest where parallelization is currently benefiting from parallel application in large scale:

- Atmosphere, earth, environment, climate modeling
- Physics, applied, nuclear, particle, condensed matter, high pressure, fusion, photonics,
- Bioscience, biotechnology, genetics, drug discovery
- Chemistry, molecular science
- Geology, seismology, energy research
- Mechanical engineering, spacecraft
- Electrical engineering, circuit design, micro electronics
- Computer science, mathematics, data analysis.

5. Conclusion

Where there is data, there must be data mining applications. Data mining is a dynamic and fast-expanding field with great strength that is important in our day to day activities, either consciously or unconsciously.

All parallelism is explicit, meaning it's the programmers' responsibility for correctly identifying parallelism and implementing parallel algorithm. Programmers can easily understand the high-level design of the program. However, in this program, there is communication overhead between the master and slave processors.

In parallel processing, distributed shared memory parallel machine is the main trends of development of high performance computer. Its main features are as follows: each node is a shared memory multi-processor, and they are connected. Memory access in the internal node and message passing between nodes, OpenMP & MPI respectively, can make better use of distributed shared memory system.

Many parallel algorithms divide the dataset into partitions for parallel processing among processors for finding frequent item sets. However, Apriori algorithm has weakness in performance with the exponential growth. The key factors in determining the performance of finding significant association rules in knowledge discovery is the size of the item set. In this program, the tree is incrementally created dependent on the frequency count of candidate item sets, load balancing is no taken under consideration, which initiate further research on finding ways for optimization.

Reference

[1] Pacheco, Peter S. University of San Francisco An introduction to parallel programming Elsevier Inc.2011
[2] Argonne National Laboratory Introduction to MPI 2014
[3] Argonne National Laboratory Advanced MPI 2014
[4] AnanthGrama, Anshul Gupta, George Karypis, Vipin Kumar Introduction to Parallel Computing, second edition Addison Wesley 2003
[5] HaoqiangJin, Dennis Jespersen, PiyushMehrotra, Rupak Biswas, Lei Huang and Barbara Chapman Parallel computing: High performance computing using MPI and OpenMP on multi-core parallel systems Elsevier 2011
[6] Aiichiro Nakano Hybrid MPI+OpenMP parallel MD University of Southern California
[7] Henry Kasim, Verdi March, Rita Zhang, and Simon See Survey on Parallel Programming Model National University of Singapore
[8] Chia-Chu Chiang Programming Parallel Apriori Algorithm for Mining Association Rules University of Arkansas 2010
[9] Jiawei Han, MichelineKamber and Jian Pei Data mining: Concepts and Techniques third edition Elsevier Inc. 2012
[10] Takao Terano, Huan Liu, ArbeeL.P.ChenKnowledge Discovery and Data Mining: current issues and new applications: 4th Pacific Asia Conference: Lecture notes in artificial intelligence Springer 2000

Author Profile

Negussie Zadig Serawit: born in Addis Ababa Ethiopia, and studied primary and secondary education in an international French school, I received my B.S. in Natural Resource Management from Hawassa University Wondogenet College of Forestry and Natural Resource in 2007. Afterwards, worked for few months in an international college, and a French tour agency while studying my postgraduate diploma in Computer Science at HILCOE School of Computer Science and Technology and graduate in 2010. Furthermore, worked at Ethiopian Airlines and LyceeGuebre Mariam Franco-Ethiopien international school. In 2012, certified in Chinese language from Tianjin University of Technology and Education, Tianjin China, which I am on my way to graduate my Ms in Applied Computer Technology from the same university by 2015.