Jimma university

INISTITUTE OF TECHNOLOGY
SCHOOL OF COMPUTING

Names	**ID**
1. Atsdaq Eshetu	Ru 0552/08
2. Ali Mohammed	Ru 0725/08
3. Binyam Ayana	Ru 0501/08

Group one

Distributed and parallel Optimization in Data Mining

Introduction

Data Mining and Knowledge Discovery in Database (KDD) is a new field which is a collection of statistics, machine learning, databases, and parallel and distributed computing. It has been cultivated by the phenomenal growth of data in all aspects of human endeavor and the economic and scientific need to extract useful information from a collected data. One of the main challenges in data mining is the extraction of knowledge and insight from massive databases.

Parallel data mining (PDM) deals with tightly-coupled systems including shared-memory systems (SMP), distributed-memory machines (DMM), or clusters of SMP workstations (CLUMPS) with a fast interconnect. Distributed datamining (DDM), on the other hand, deals with loosely-coupled systems such as a cluster over a slow Ethernet local-area network. It also includes geographically distributed sites over a wide-area network like the Internet. The main differences between PDM to DDM are best understood if view DDM as a gradual transition from tightly-coupled, fine-grained parallel machines to loosely-coupled medium grained LAN of workstations, and finally very coarse-grained WANs. There is in fact a significant overlap between the two areas, especially at the medium grained level where is it hard to draw a line between them.

Parallel and Distributed Data Mining

Parallel and Distributed computing is expected to solve the problem that is in the current mining methods from the sequential bottleneck, which is by providing the ability to scale to massive datasets, by improving the response time. It can achieve a good performance on today's multiprocessor systems in a non-trivial task. The main challenges include synchronization and communication minimization, work-load balancing, finding good data layout and data decomposition, and disk I/O minimization, which is especially important for data mining.

Parallel Design space

The parallel design space includes many systems and algorithmic components. These components include the hardware platform, the load balancing strategy, Parallelism, the data layout and the search procedure used.
DMMs synchronization is implicit in message passing, so the goal becomes communication optimization. There are different methods that is executed when synchronization is done. For shared-memory systems, synchronization happens via locks and barriers, and the goal is to minimize these points. Data decomposition is very important for distributed memory, but not for shared memory. While parallel I/O comes for "free" in DMMs, it can be problematic for SMP machines, which typically serialize I/O. The main challenge for obtaining good performance on DMM is to find a good data decomposition among the nodes, and to minimize communication. For SMP the objectives are to achieve good data i.e., minimize the ping-pong effect where multiple processors may be trying to modify different variables which coincidentally reside on the same cache line. For today's non-uniform memory access (NUMA) hybrid and/or hierarchical machines (e.g., cluster of SMPs), the optimization parameters draw from both the DMM and SMP paradigms.

There are different architectures based on the database literature. These include: -

- Task vs Data Parallelism
- Static vs Dynamic Load Balancing
- Horizontal vs Vertical Data layout
- Complete vs Heuristic candidate data generation

Task vs Data Parallelism

These are two main paradigms for exploiting algorithm parallelism. Data parallelism corresponds to the case where the database is partitioned among processors. Each processor works on its local partition of the database, but performs the same computation of evaluating candidate patterns/models. Task parallelism corresponds to the case where the processors perform

different computations independently, such as evaluating a disjoint set of candidates, but have/need access to the entire database. SMPs have access to the entire data, but for DMMs this can be done via selective replication or explicit communication of the local data. Hybrid parallelism combining both task and data parallelism is also possible, and in fact desirable for exploiting all available parallelism in data mining methods.

Static vs Dynamic Load Balancing

In static load blanching work is partitioned among the processors using some heuristic cost function, and there is no subsequent data or computation movement to correct load imbalances
which result from the dynamic nature of mining algorithms. Dynamic load balancing seeks to address this by stealing work from heavily loaded processors and re-assigning it to lightly loaded ones. Computation movement also entails data movement, since the processor responsible for a computational task needs the data associated with that task as well. Dynamic load balancing thus incurs additional costs for work/data movement, but it is beneficial if the load imbalance is large and if load changes with time. Dynamic load balancing is especially important in multi-user environments with transient loads and in heterogeneous platforms, which have different processor and network speeds. These kinds of environments include parallel servers, and heterogeneous, meta-clusters. With very few exceptions, most extant parallel mining algorithms use only a static load balancing approach that is inherent in the initial partitioning of the database among available nodes. This is because they assume a dedicated, homogeneous environment.

Horizontal vs Vertical Data layout

The standard input database for mining is a relational table having N rows, also called feature vectors, transactions, or records, and M columns, also called dimensions, features, or attributes. The data layout can be row-wise or column-wise. Many data mining algorithms assume a horizontal or row-wise database layout, where they store, as a unit, each transaction(tid), along with the attribute values for that transaction. Other methods use a vertical or column-wise database layout, where they associate with each attribute a list of all tids (called tidlist) containing the item, and the corresponding attribute value in that transaction. Certain mining operations a more efficient using a horizontal format, while others are more efficient using a vertical format.

Complete vs Heuristic candidate data generation

The final result of mining method may be sets, sequences, rules, trees, networks, etc., ranging from simple patterns to more complex models, based on certain search criteria. In the intermediate steps several candidate patterns or partial models are evaluated, and the final result contains only the ones that satisfy the (user-specified) input parameters. Mining algorithms can differ in the way new candidates are generated for evaluation. One approach is complete search, which is guaranteed to generate and test all valid candidates consistent with the data. Note that completeness doesn't mean exhaustive, since pruning can be used to eliminate useless branches in the search space. Heuristic generation sacrifices completeness for the sake of speed. At each step, it only examines a limited

number (or only one) of "good" branches. Random search is also possible. Generally, the more complex the mined model, the more the tendency towards heuristic or greedy search.

Candidate and Data Partitioning

We use this to describe the parallel and distributed mining methods in terms of the computation and data partitioning methods that are used. For example, the database itself can be shared (in shared-memory or shared-disk architectures), partially or totally replicated, or partitioned (using round-robin, hash, or range scheduling) among the available nodes (in distributed-memory architectures).

The candidate concept is also generated and evaluated in the different mining methods that can be replicated, partitioned or shared. Shared is the state when all processors evaluate a single copy of candidate set. Replication is the process of approaching the candidate concepts on each machine, it involves first evaluating the local data before global results are obtained by merging item. Finally, in the partitioned approach, at first each process generates results, then it tests a disjoint candidate concept set.

Application Areas In Which Distributed Mining Frameworks Used

In recent times there is a huge interest in distributed and wide-area data mining systems. This is because many global businesses and scientific endeavors need an access of multiple distributed, and most of the time heterogeneous databases.

An ideal platform for DDM is a cluster of machines at a local site, or cluster of clusters spanning a wide area, the so-called computational grids, connected via Internet or other high speed networks. As we noted earlier, PDM is best viewed as a local component within a DDM system. Further the main differences between the two is the cost of communication or data movement, and the fact that DDM must typically handle multiple (possibly heterogeneous) databases. Below we review some recent efforts in developing DDM frameworks.

Most DDM's take data as it is horizontally partitioned, and that it's homogeneous (share the same feature space). So the local data is mined by the sites and they generate locally valid concepts. This results to obtain globally valid concepts.

For example, there is a new distributive system called do-all primitive called D-DOALL, which allows easy scheduling of independent mining task on a network of work stations. The D-DOALL framework allows incremental reporting of results, and it intends to reduce communication via resource aware task scheduling principal.

In general, big data mining has become a big research area. Now a day it is very difficult to use traditional methods and data mining software tools for a single personal computer to efficiently deal with a very large dataset. For this problem the parallel and cloud computing platforms are a better solution for big data mining. Parallel computing is based on dividing a large problem into smaller ones and carrying each of them out by one single processor individually. In addition, these processes are performed concurrently in a distributed and parallel manner. For this there are two major ways to tackle this problem.

- The first one is the distributed procedure based on the data parallelism paradigm, where a given big dataset can be manually divided into n subsets, and n algorithms are respectively executed for the corresponding n subsets. And the final result of the corresponding n subset can be obtained from a combination of the outputs produced by the n algorithm.
- The second one is the MapReduce based procedure under which the cloud computing platform. This procedure composes of the map and reduce processes, it is basically where the former performs a summary operation in order to produce the final result.

But the second one is very complicated and new so most technologists still prefer to use the distributed procedure that is based on the data parallelism paradigm.

References

1. Simoudis, E.: Reality check for data mining. IEEE Expert: Intelligent Systems and Their Applications 11 (1996) 26–33.
2. Mueller, A.: Fast sequential and parallel algorithms for association rule mining: Technical Report CS-TR-3515, University of Maryland, College Park (1995).
3. www.gurobi.com
4. www.sciencedirect.com