Parallel Data Mining on Shared-Memory Multiprocessors

Rakesh Agrawal Ching-Tien Ho Leon Pauser Mohammed Zaki
IBM Almaden Research Center

In this talk, we will present parallel algorithms for generating decision-tree classifiers, association rules and sequential patterns on shared-memory systems. We will discuss the design, implementation issues, and performance results on AIX RS/6000 SMP's and OS/390 systems.

Classification is recognized to be a primary data mining task. The input to a classification system consists of a set of example tuples, called a *training set*, where each tuple consists of several *attributes*. One of the attributes, called the *class* attribute, indicates the class to which each example belongs. The goal of classification is to induce a model from the training set, that can be used to predict the class of a new tuple that does not have a class label.

In the association rules, the user is given a database of transactions, where each transaction consists of a set of items, and is interested in finding all rules such that the presence of one set of items in a transaction implies the presence of another set of items. In the sequential patterns, the user is given a set of transactions over a period of time and is interested in finding inter-transaction patterns such that the presence of a set of items is followed by another set of items.

The parallel classification algorithms we propose span the gamut of data and task parallelism. The data parallelism is based on attribute scheduling among processors—scheduling work associated with different attributes to different processors. This basic scheme is extended with task pipelining and dynamic load balancing to yield more efficient schemes. The task parallel approach uses dynamic subtree partitioning among processors. These algorithms are evaluated on two different AIX RS/60000 SMP configurations: one in which data is too large to fit in memory and must be paged from a local disk as needed and the other in which memory is sufficiently large to hold the whole input data and all temporary files. For the local disk configuration, the speedup ranged from 2.97 to 3.86 for the build phase and from 2.20 to 3.67 for the total time on a 4-processor SMP. For the large memory configuration, the range of speedup was from 5.36 to 6.67 for the build phase and from 3.07 to 5.98 for the total time on an 8-processor SMP. We will also present the implementation results on an OS/390 system, a 10-processor SMP.

The parallel algorithm for association rules is the data-parallel versions of Apriori algorithm. The input data file contains fix-size records of ASCII format: (transaction-id, item-id). They are read in, remapped and transformed into a logical file of variable-size records of binary format: (transaction-id, number-of-items, item-id, ..., item-id) during the first pass of the algorithms. In addition, the logical file is partitioned in a cyclic manner into p temporary files so that each of the p threads can work on a different file in parallel starting the second pass. The parallel algorithm for the sequential patterns is similar. One exception is that the records also contain customer-id. Different transactions of the same customer are partitioned into the same file to avoid unnecessary synchronization between threads. We will present experimental results for these algorithms on both AIX SMP and OS/390 systems.

All implementations use the POSIX threads (pthread) standard, making them portable to different UNIX platforms and to OS/390 systems which support an essential subset of pthreads.