Roman Kazarin
Michael Nekrasov
HW5 Paper Summary

A Development and Deployment Framework for Distributed Branch & Bound – Summary

Branch and bound is a technique used to find the optimal solution without actually looking at every single solution. However, because the number of feasible solutions grows exponentially as you increase the size of the input, using branch and bound becomes computationally infeasible. A search tree is formed by partitioning the space of feasible solutions. Each node of the tree contains a partial feasible solution. A node can be pruned if we see that the set of solutions it represents is worse than a solution we have already found. This paper talks about using branch and bound in a distributed setting.

A problem is first decomposed into tasks, which are computed on a compute server. If a computer server, after finishing a task, discovers a better solution than it currently knows about, it will let every other compute server know about this solution, so that they can begin pruning nodes with worse solutions than this new solution.

The paper introduces JICOS, a scalable, fault-tolerant, parallel computing network computing service, written in Java. JICOS is made up of a Hosting Service provider with which clients communicate tasks to, a Task Server which stores Task objects, and a Host which request and executes tasks. JICOS tries to reduce communication latency and in turn improve performance by Task caching and Task pre-fetching. It also computes some tasks right on the Task server, simply because it would take longer to send a smaller task to a host than to just compute it right on the task server.

A DAG in which nodes represents tasks and arcs represent direction of information flow models the computation. At any time, the currently known least cost solution is stored inside the computation's *shared* object. Again, we use this object to decide whether we need to perform a computation or if we can prune an entire subtree instead. In a TSP problem, a Solution object could be a partial tour. The object has methods to *getChildren, getLowerBound, getUpperBound, isComplete,* and *reduce.* The *reduce* method is what actually omits edges whose cost is bigger than the current best solution.

For the experiments, a 200-city TSP problem instance on 64 computer instances was run. Some measurements that were used were: Tp – the time for *p* processors to run the application, Tinf – the maximum time path from source task to sink task (the critical path). The paper reports superlinear speedups using 4, 8, 16, and 32 processors – therefore the JICOS system is a success.

The JICOS system helps develop distributed branch and bound computations. Because the programmer mostly focuses on implementing lower/upper bound and branching factor computation, the likelihood of getting an error elsewhere is low.