Roman Kazarin
Michael Nekrasov
CS290B – Peter Cappello

Paper Summary: CX: A Scalable, Robust Network for Parallel Computing

This paper introduces CX, a system for distributed programming over a network of private or public computers. It allows an application programmers to easily code distributed tasks without worrying about processor communication and fault tolerance. The system once installed is automatically upgradable. The system uses the java virtual machine to provide homogeneity across platforms.

The paper argues that computational power will continue to scale with time, while human labor and administration will remain costly. The system therefore focuses around reducing administrative overhead by exploiting the computational power available online. As communication latency evolves at a slower rate than computational power, reducing communication latency is central to the system's design.

The computational model of CX is based on non-blocking tasks that are connected in a directed acyclic graph. The application programmer controls the decomposition and composition of tasks. While the system is implemented as a network of servers, the application programmer can interact with it as if it was a single task server. To minimize communication overhead, tasks should decompose into chunks of sufficient computational complexity to hide communication latency.

Tasks are scheduled based on the number of times they have already been assigned and on their level in the acyclic graph. This behavior is exposed to the programmer, and can be modified overridden to set the relative priority of tasks. The load is distributed among the producers, and allows assigning a single task to multiple producers. When producers deplete their task cache they steal tasks from other producers.

To keep a single server from being a bottleneck, the server can be run as a network of servers. This way tasks get distributed out among the network and their producers. The network is tolerant of failures, maintaining a mirrors of transitional states of the server nodes. The code for the tasks is downloaded from the tree of servers. This prevents forming a bottleneck at code download.

All together, CX builds on much of the previous work in distributed platforms like Condor and Cilk and provides an elegant platform that is system agnostic and fault tolerant. The systems we have put together in class have many of the groundwork components of CX and give us a good insight into the system.