Abstract

Scoped Behaviour for
Optimized Distributed Data Sharing

Chien-Ping Paul Lu
Doctor of Philosophy
Graduate Department of Computer Science
University of Toronto
2000

We introduce the novel *scoped behaviour* abstraction and examine how it is used to optimize distributed data-sharing patterns within the Aurora parallel programming system. Scoped behaviour is an application programmer's interface to a set of system-provided optimizations; it is also an implementation framework for the optimizations. Aurora is a distributed shared data system where shared-data objects are implemented in C++ as abstract data types. Aurora has been prototyped on a network of workstations connected by an ATM network.

The design, implementation, and evaluation of Aurora and scoped behaviour contributes to the field of parallel and distributed systems by demonstrating that:

1. Scoped behaviour can provide per-object and per-context (i.e., specific portion of the source code) flexibility when applying data-sharing optimizations. In contrast to some other systems, Aurora programs can be incrementally tuned and only a minimum number of error-prone changes to the source code are required in order to experiment with different optimization strategies.

2. Scoped behaviour can be implemented without language extensions and without special compiler support. Scoped behaviour's novel implementation framework can exploit both compile-time and run-time information about the parallel program.

3. A parallel programming system based on a high-level shared-data abstraction can achieve high performance. In a performance evaluation of four applications implemented using

three different types of parallel programming systems, Aurora usually matches or outperforms, sometimes by a wide margin, TreadMarks (a distributed shared memory system) and a message-passing system (either MPICH or PVM, depending on the application).