C^3: A System for Automating Application-level Checkpointing of MPI Programs

Greg Bronevetsky, Daniel Marques, Keshav Pingali, Paul Stodghill

To appear at the *16th Workshop on Languages and Compilers for Parallel Computing (LCPC03)*, College Station, TX, 2-4 October 2003

Abstract

Fault-tolerance is becoming necessary on high-performance platforms. Checkpointing techniques make programs fault-tolerant by saving their state periodically and restoring this state after failure. *System-level* checkpointing saves the state of the entire machine on stable storage, but this usually has too much overhead. In practice, programmers do manual *application-level* checkpointing by writing code to (i) save the values of key program variables at critical points in the program, and (ii) restore the entire computational state from these values during recovery. However, this can be difficult to do in general MPI programs.

In ([2],[3]) we have presented a distributed checkpoint coordination protocol which handles MPI's point-to-point and collective constructs, while dealing with the unique challenges of application-level checkpointing. We have implemented our protocols as part of a thin software layer that sits between the application program and the MPI library, so it does not require any modifications to the MPI library. This thin layer is used by the C^3 (Cornell Checkpoint (pre-)Compiler), a tool that automatically converts an MPI application in an equivalent fault-tolerant version. In this paper, we summarize our work on this system to date. We also present experimental results that show that the overhead introduced by the protocols are small. We also discuss a number of future areas of research.