Allyson Pascua
04-23-2013
Assignment 0411

The Complexity of Obstruction-Free Implementations

This paper explores obstruction-free implementation performance, which is the guarantee of progress for every thread that eventually executes alone even as threads are performing their own operations. It focuses on the cost of the program where no step contention is encountered and what the benefits are to that.

Its connection to the documents provided in class can be found near the end of the document. The paper claims there exists lock-based implementations not subject to the problem of processes causing delay without the awareness of step contention. This is possible through the use of semaphores, or some conditional variables, vs. busy waiting.

The Programming Language Concurrent Pascal

This paper describes a programming language called Concurrent Pascal that was designed specifically for concurrent programs and monitoring systems on shared memory computers. It focuses heavily on the utilization of monitors to synchronize concurrent processes and transmit data inbetween. However, Concurrent Pascal also seeks to extend the functionality of monitors with explicit access rights that can be checked at compile time.

Securing Interaction between Threads and the Scheduler in the Presence of Synchronization

This paper addresses the security risks associated with multithreaded programming, including permissiveness, scheduler-independence, realistic semantics, language expressiveness, and practical enforcement. It then details/develops a solution to a security type system that addresses such problems.

Its connection to the documents provided in class can be found in Section *9 – Synchronization Primitives*. It focuses on semaphores and directly quotes Dijkstra's *Cooperating Sequential Processes*. Semaphores are commonly used synchronization primitives. The problem with them is performance loss. With that comes the possibility of interference. The author details how well-written semaphores based on blocked waiting (thread is locked and waiting for a monitor lock) can avoid these problems and ensure all necessary information is protected.

Cited Works

Attiya, Hagit, Rachid Guerraoui, and Danny Hendler. "The Complexity of Obstruction-Free Implementations." . Journal of the ACM (JACM), n.d. Web. 23 Apr 2013. <http://www.cs.bgu.ac.il/~hendlerd/papers/OF-JACM.pdf>.

Hansen, Per Brinch. "The Programming Language Concurrent Pascal." . Department of Information Science, California Institute of Technology, Pasadena, n.d. Web. 23 Apr 2013. <http://brinch-hansen.net/papers/1975a.pdf>.

Russo, Alejandro, and Sabelfeld Andrei. "Securing Interaction between Theads and the Scheduler in the Presence of Synchronization." . IEEE Computer Security Foundations Workshop, n.d. Web. 23 Apr 2013. <http://www.cse.chalmers.se/~russo/publications_files/JLAP-russo-sabelfeld.pdf>.