Brief Announcement: Local-Spin Algorithms for Abortable Mutual Exclusion and Related Problems

Robert Danek and Hyonho Lee

Department of Computer Science
University of Toronto
{rdanek,hlee}@cs.toronto.edu

Introduction. A mutual exclusion (ME) algorithm consists of a *trying protocol (TP)* and *exit protocol (EP)* that surround a *critical section (CS)* and satisfy the following properties: **mutual exclusion**: at most one process is allowed to use the CS at a given time; **lockout freedom**: any process that enters the TP eventually enters the CS; and **bounded exit**: a process can complete the EP in a bounded number of its own steps. A First-Come-First-Served (FCFS) ME algorithm [1] additionally requires processes to enter the CS in roughly the order in which they start the TP. Once a process has started executing the TP of a ME algorithm, it has committed itself to entering the CS, since the correctness of the algorithm may depend on every process properly completing its TP and EP.

Abortable ME [2, 3] is a variant of ME in which a process may change its mind about entering the CS, e.g., because it has been waiting too long. A process can withdraw its request by performing a bounded section of code, called an *abort protocol (AP)*.

We discuss novel algorithms for abortable ME and FCFS abortable ME. These algorithms are local-spin, i.e., they access only local variables while waiting and perform only a bounded number of remote memory references (RMRs) in the TP, EP and AP. Using these algorithms, we obtain new local-spin algorithms for two other additional problems: group mutual exclusion (GME) [4] and k-exclusion [5].

Summary of Results. All our algorithms use only atomic reads and writes. We call these *RW algorithms*. Our main result is the first RW local-spin abortable ME algorithm. It has $O(\log N)$ RMR complexity per operation and $O(N \log N)$ (total) space complexity for N processes. It is a surprisingly simple modification of the RW local-spin ME algorithm of Yang and Anderson [6]: we allow a process waiting in an unbounded loop in the TP to abort by executing the EP.

We also have a transformation that converts any abortable ME algorithm that has $O(T)$ RMR complexity and $O(S)$ space complexity to an FCFS abortable ME algorithm that has $O(N + T)$ RMR complexity and $O(S + N^2)$ space complexity. Given an abortable ME algorithm, we add code to the beginning of its TP: a process p builds a “predecessor” set, which includes all processes that must enter the CS before it. Process p then waits for its predecessors to finish the CS, during which time it can abort. We also add code to the end of the EP and AP: p signals to other processes that may have p in their predecessor set. This transformation combined with the modified Yang and Anderson algorithm yields the first RW local-spin FCFS abortable ME algorithm. It has $O(N)$ RMR complexity and $O(N^2)$ space complexity. This also uses only bounded registers, so it yields a positive solution to an open problem mentioned by Jayanti [3].

Danek and Hadzilacos [7] presented a number of transformations using only reads and writes that convert any FCFS abortable ME algorithm that has $O(T)$ RMR complexity and $O(S)$ space complexity into a local-spin GME algorithm that has $O(N+T)$ RMR complexity and $O(S+N^2)$ space complexity. Together with our FCFS abortable algorithm, this leads to the first RW local-spin GME algorithm. It has $O(N)$ RMR complexity and $O(N^2)$ space complexity.

Lastly, we convert any abortable ME algorithm that has $O(T)$ RMR complexity and $O(S)$ space complexity to a k-exclusion algorithm that has $O(k \cdot T)$ RMR complexity and $O(k \cdot S)$ space complexity, but is not fault-tolerant. The transformation uses k instances of an abortable ME algorithm.

When a process enters the TP of the k-exclusion algorithm, it performs all k instances of the abortable mutual exclusion algorithm concurrently (for example, repeatedly performing one step of each in round-robin order) until it enters the CS of one of the instances. When the process enters the CS of the jth abortable ME algorithm, it finishes or aborts its execution of all other instances before entering the CS of the k-exclusion algorithm. When the process finishes the CS of the k-exclusion algorithm, it performs the EP of the jth abortable ME algorithm.

Applied to our abortable ME algorithm, this yields the first RW local-spin k-exclusion algorithm. It has $O(k \cdot \log N)$ RMR complexity.

Acknowledgments. We thank Faith Ellen and Vassos Hadzilacos for their numerous helpful suggestions during the writing of this paper.

References

1. Lamport, L.: A new Solution of Dijkstra's Concurrent Programming Problem. Communications of the ACM **17**(8) (August 1974) 453–455
2. Scott, M.L.: Non-blocking Timeout in Scalable Queue-based Spin Locks. In: The 21st Annual Symposium on Principles of Distributed Computing. (July 2002)
3. Jayanti, P.: Adaptive and Efficient Abortable Mutual Exclusion. In: Proceedings of the 22nd Annual ACM Symposium on Principles of Distributed Computing. (July 2003)
4. Joung, Y.J.: Asynchronous group mutual exclusion. Distributed Computing **13**(4) (2000) 189–206
5. Anderson, J.H., Moir, M.: Using local-spin k-exclusion algorithms to improve wait-free object implementations. Distributed Computing **11**(1) (1997) 1–20
6. Yang, J.H., Anderson, J.H.: A Fast, Scalable Mutual Exclusion Algorithm. Distributed Computing **9**(1) (August 1995) 51–60
7. Danek, R., Hadzilacos, V.: Local-spin group mutual exclusion algorithms. In: DISC. (2004) 71–85