CS586 – Distributed Computing
Fall semester - Ac. Year 2011-12
Instructor: Panagiota Fatourou

Programming Projects

Deadline: February 13, 2012

Teaching assistant: Eleftherios Kosmas (ekosmas@csd.uoc.gr)

Each of the projects includes the study, the implementation, and the evaluation of some algorithm.

1) STM algorithms

In the literature several STM algorithms have been proposed and this course focuses to those that are considered as the state of the art, which are the following:

[1] DSTM: M. P. Herlihy, V. Luchangco, M. Moir and W. N. Scherer III. "Software Transactional Memory for Dynamic-Sized Data Structures", In Proceedings of 22nd Annual ACM Symposium on Principles of Distributed Computing (PODC), pages 92-101, July 2003.

[2] NZSTM: Fuad Tabba, Cong Wang, and Mark Moir, "NZTM: Nonblocking Zero-Indirection Transactional Memory", Proceedings of TRANSACT, 2007 (http://www.cs.rochester.edu/meetings/TRANSACT07/papers/tabba.pdf)

[3] NBSTM: Eleftherios Kosmas, «Software Transactional Memory», Dissertation, 2008.

[4] RingSTM: Michael F. Spear, Maged M. Michael and Christoph von Praun, "RingSTM: Scalable Transactions with a Single Atomic Instruction", Proceedings of the ACM Symposium on ParallelAlgorithmsandArchitectures, 2008 (http://www.cs.rochester.edu/u/spear/spaa08.pdf)

[5] OSTM: Keir Fraser, "Practical lock freedom", PhD thesis, Cambridge University Computer Laboratory (also available as Technical Report UCAM-CL-TR-579).

[6] TLII: D. Dice, O. Shalev and N. Shavit. "Transactional Locking II", In Proceedings of the 20th International Symposium on Distributed Computing (DISC), September 2006.

2) Concurrent data structures

[1] Maged M. Michael. 2002. High performance dynamic lock-free hash tables and list-based sets. In *Proceedings of the fourteenth annual ACM symposium on Parallel algorithms and architectures* (SPAA '02). ACM, New York, NY, USA, 73-82.

[2] Mikhail Fomitchev and Eric Ruppert. 2004. Lock-free linked lists and skip lists. In *Proceedings of the twenty-third annual ACM symposium on Principles of distributed computing* (PODC '04).

[3] Faith Ellen, Panagiota Fatourou, Eric Ruppert, and Franck van Breugel. 2010. Non-blocking binary search trees. In *Proceedings of the 29th ACM SIGACT-SIGOPS symposium on Principles of distributed computing* (PODC '10). ACM, New York, NY, USA, 131-140.

[4] Maged M. Michael. 2004. Hazard Pointers: Safe Memory Reclamation for Lock-Free Objects. *IEEE Trans. Parallel Distrib. Syst.* 15, 6 (June 2004), 491-504.

Project Description

The goal of this project is the study, the implementation, and the evaluation of existing STM algorithms and concurrent data structures. In each of the projects, one of the above algorithms should be studied and their pseudo code should be written (for those that is not presented). Moreover, complicated points of the description and any necessary assumptions or modifications, should be highlighted.

Then, each algorithm should be implemented in the C programming language using an additional library that provides the implementation of strong registers (i.e., the implementation of their atomic operations), e.g., CAS and LL/SC, which are required for implementing those algorithms.

Each STM algorithm should be implemented as a library providing an API through which the user (i.e., the naïve parallel programmer) can use in his/her code, in order to achieve executing it atomically. This interface is common for all the algorithms and it will be presented during the course's corresponding lecture. Writing appropriate code for each of the functions supported by the interface and any additional required procedures, is what you are requested to do for each algorithm.

Finally, the performance of each algorithm should be studied analytically. For the STM algorithms, performance measures that can be studied are: the number (or percentage) of committed and aborted transactions, the mean time of transaction's successful completion (which may be re-executed if it aborts), the mean number of aborts for each transaction before it commits, etc.

Deliverables

You should deliver the code of the library that implements the algorithm of your project. Moreover, you should deliver any code used to evaluate your implementation. Finally, a report is required that includes the pseudo code of the algorithm, explains any modifications or assumptions you have taken, includes the description of the experiments you have performed (for STM algorithms, it is required to present at least the implementation of some data structures, e.g., linked list, sorted linked list, etc, using the STM interface you implemented; also, it is recommended to use STAMP benchmarks), and presents your evaluation of the algorithm.