# Contributions summary, TRAMP Workshop, February 2007

Michael L. Scott, Sandhya Dwarkadas, William N. Scherer III,
Virendra J. Marathe, Michael F. Spear, Arrvindh Shriraman,
Vinod Sivasankaran, Hemayet Hossain, Luke Dalessandro,
Athul Acharya, Chris Heriot, David Eisenstat, and Aaron Rolett

University of Rochester

Transactional Memory research at Rochester is the outgrowth of 16 years of work in synchronization and concurrency. A full publication list and on-line copies of papers can be found at *www.cs.rochester.edu/research/synchronization/pubs.shtml*.

Bill Scherer, now a Faculty Fellow at Rice University but then a graduate student intern, was part of the original team that developed the Dynamic STM (DSTM) system at Sun Labs [4]. As part of his thesis work, he developed and evaluated a large suite of pluggable contention managers for obstruction-free STM [14, 17, 18]. Virendra Marathe began his graduate career with an early analysis of design tradeoffs for practical STM [6, 7]. This work led to the Java-based ASTM system, which adapts its degree of eagerness and level of indirection based on characteristics of the offered workload [8]. Virendra also showed how to use load-linked/store-conditional to simplify and optimize Harris and Fraser's word-based STM (WSTM) [9]. More recently, together with Mark Moir of Sun Labs, he has developed a general approach to nonblocking copyback in word-based STM [12].

In 2005 we began to shift our focus from Java to C++, with the goals of (1) developing an open-source STM for unmanaged code, (2) achieving a better understanding of fundamental costs, without the complication of the JVM or JIT compiler, and (3) facilitating experimentation with hardware-software hybrid implementations. Our RSTM system [11] is available at *www.cs.rochester.edu/research/synchronization/rstm/* (development led by Mike Spear). We are also continuing work with Java and C#.

We have used RSTM to evaluate a variety of strategies for conflict detection and read-set validation [25]. Our work highlights the importance of validation (or sandboxing) for correctness, and suggests that the problem of erroneous behavior in doomed transactions is underappreciated in existing STM work. We have also used RSTM as the foundation for work in hardware acceleration of software TM. Our RTM system [23, 24] exploits two hardware assists: *alert on update* (AOU), which allows a thread to receive fast, asynchronous notification when previously-identified lines are written by other threads, and *programmable data isolation* (PDI), which allows a thread to hide its speculative writes from other threads until software decides to make them visible. We have also shown how AOU can be used by itself to eliminate most of the overhead of validation [26], or to implement simple, non-blocking STM with in-place updates (RTM-Lite) [27].

In other work, we have explored the scheduling of lightweight transactions in a bag-of-tasks programming model [1, 13] (in part with Tim Harris and James Larus); shown how to enable communication among nonblocking transactions [5]; and formalized the sequential semantics of transactions [22]. We also continue to work in nontransactional areas of synchronization and concurrency, with recent papers on preemption-adaptive queue-based spin locks [2, 20, 21]; *dual data structures*, which enable condition synchronization in ad hoc nonblocking data structures [15, 16, 19]; composite abortable locks (with Mark Moir and Nir Shavit) [10]; and lazy concurrent list-based sets (with Heller, Herlihy, Luchangco, Moir, and Shavit) [3].

# References


[1] T. Bai, X. Shen, C. Zhang, W. N. Scherer III, C. Ding, and M. L. Scott. A Key-Based Adaptive Transactional Memory Executor. In *Proc. of the NSF Next Generation Software Program Workshop*, Mar. 2007. Invited paper. Also available as TR 909, Dept. of Computer Science, Univ. of Rochester, Dec. 2006.

[2] B. He, W. N. Scherer III, and M. L. Scott. Preemption Adaptivity in Time-Published Queue-Based Spin locks. In *Proc. of the 2005 Intl. Conf. on High Performance Computing*, Goa, India, Dec. 2005. Earlier but expanded version available as TR 867, Dept. of Computer Science, Univ. of Rochester, May 2005.

[3] S. Heller, M. Herlihy, V. Luchangco, M. Moir, N. Shavit, and W. N. Scherer III. A Lazy Concurrent List-based Set Algorithm. In *9th Intl. Conf. on Principles of Distributed Systems*, Pisa, Italy, Dec. 2005.

[4] M. Herlihy, V. Luchangco, M. Moir, and W. N. Scherer III. Software Transactional Memory for Dynamic-sized Data Structures. In *Proc. of the 22nd ACM Symp. on Principles of Distributed Computing*, pages 92-101, Boston, MA, July 2003.

[5] V. Luchangco and V. J. Marathe. Transaction Synchronizers. In *Proc., Workshop on Synchronization and Concurrency in Object-Oriented Languages*, San Diego, CA, Oct. 2005. In conjunction with OOPSLA '05.

[6] V. J. Marathe and M. L. Scott. A Qualitative Survey of Modern Software Transactional Memory Systems. TR 839, Dept. of Computer Science, Univ. of Rochester, June 2004.

[7] V. J. Marathe, W. N. Scherer III, and M. L. Scott. Design Tradeoffs in Modern Software Transactional Memory Systems. In *Proc. of the 7th Workshop on Languages, Compilers, and Run-time Systems for Scalable Computers*, Houston, TX, Oct. 2004.

[8] V. J. Marathe, W. N. Scherer III, and M. L. Scott. Adaptive Software Transactional Memory. In *Proc. of the 19th Intl. Symp. on Distributed Computing*, Cracow, Poland, Sept. 2005.

[9] V. J. Marathe and M. L. Scott. Using LL / SC to Simplify Word-based Software Transactional Memory (poster paper). In *Proc. of the 24th ACM Symp. on Principles of Distributed Computing*, Las Vegas, NV, July 2005.

[10] V. J. Marathe, M. Moir, and N. Shavit. Composite Abortable Locks. In *Proc. of the 2006 Intl. Parallel and Distributed Processing Symp.*, Rhodes, Greece, Apr. 2006.

[11] V. J. Marathe, M. F. Spear, C. Heriot, A. Acharya, D. Eisenstat, W. N. Scherer III, and M. L. Scott. Lowering the Overhead of Software Transactional Memory. In *ACM SIGPLAN Workshop on Transactional Computing*, Ottawa, ON, Canada, June 2006. Held in conjunction with PLDI 2006. Expanded version available as TR 893, Dept. of Computer Science, Univ. of Rochester, Mar. 2006.

[12] V. Marathe and M. Moir. An Efficient Nonblocking Copyback Mechanism in Hybrid Transactional Memory (poster paper). In *Proc. of the 12th ACM Symp. on Principles and Practice of Parallel Programming*, San Jose, CA, Mar. 2007.

[13] V. Marathe, T. Harris, and J. Larus. Featherweight Transactions: Decoupling Threads and Atomic Blocks (poster paper). In *Proc. of the 12th ACM Symp. on Principles and Practice of Parallel Programming*, San Jose, CA, Mar. 2007.

[14] W. N. Scherer III and M. L. Scott. Contention Management in Dynamic Software Transactional Memory. In *Proc. of the ACM PODC Workshop on Concurrency and Synchronization in Java Programs*, St. John's, NL, Canada, July 2004.

[15] W. N. Scherer III and M. L. Scott. Nonblocking Concurrent Objects with Condition Synchronization. In *Proc. of the 18th Intl. Symp. on Distributed Computing*, Amsterdam, The Netherlands, Oct. 2004.

[16] W. N. Scherer III, D. Lea, and M. L. Scott. A Scalable Elimination-based Exchange Channel. In *Proc., Workshop on Synchronization and Concurrency in Object-Oriented Languages*, San Diego, CA, Oct. 2005. In conjunction with OOPSLA '05.

[17] W. N. Scherer III and M. L. Scott. Advanced Contention Management for Dynamic Software Transactional Memory. In *Proc. of the 24th ACM Symp. on Principles of Distributed Computing*, Las Vegas, NV, July 2005.

[18] W. N. Scherer III and M. L. Scott. Randomization in STM Contention Management (poster paper). In *Proc. of the 24th ACM Symp. on Principles of Distributed Computing*, Las Vegas, NV, July 2005.

[19] W. N. Scherer III, D. Lea, and M. L. Scott. Scalable Synchronous Queues. In *Proc. of the 11th ACM Symp. on Principles and Practice of Parallel Programming*, New York, NY, Mar. 2006.

[20] M. L. Scott and W. N. Scherer III. Scalable Queue-Based Spin Locks with Timeout. In *Proc. of the 8th ACM Symp. on Principles and Practice of Parallel Programming*, Snowbird, UT, June 2001.

[21] M. L. Scott. Non-blocking Timeout in Scalable Queue-based Spin Locks. In *Proc. of the 22nd ACM Symp. on Principles of Distributed Computing*, pages 31-40, Monterey, CA, July 2002.

[22] M. L. Scott. Sequential Specification of Transactional Memory Semantics. In *ACM SIGPLAN Workshop on Transactional Computing*, Ottawa, ON, Canada, June 2006. Held in conjunction with PLDI 2006.

[23] A. Shriraman, V. J. Marathe, S. Dwarkadas, M. L. Scott, D. Eisenstat, C. Heriot, W. N. Scherer III, and M. F. Spear. Hardware Acceleration of Software Transactional Memory. In *ACM SIGPLAN Workshop on Transactional Computing*, Ottawa, ON, Canada, June 2006. Held in conjunction with PLDI 2006. Expanded version available as TR 887, Dept. of Computer Science, Univ. of Rochester, Dec. 2005, revised Mar. 2006.

[24] A. Shriraman, M. F. Spear, H. Hossain, S. Dwarkadas, and M. L. Scott. An Integrated Hardware-Software Approach to Flexible Transactional Memory. In *Proc. of the 34th Intl. Symp. on Computer Architecture*, San Diego, CA, June 2007 (to appear, pending revisions). Expanded version available as TR 910, Dept. of Computer Science, Univ. of Rochester, Dec. 2006.

[25] M. F. Spear, V. J. Marathe, W. N. Scherer III, and M. L. Scott. Conflict Detection and Validation Strategies for Software Transactional Memory. In *Proc. of the 20th Intl. Symp. on Distributed Computing*, Stockholm, Sweden, Sept. 2006.

[26] M. F. Spear, A. Shriraman, H. Hossain, S. Dwarkadas, and M. L. Scott. Alert-on-Update: A Communication Aid for Shared Memory Multiprocessors (poster paper). In *Proc. of the 12th ACM Symp. on Principles and Practice of Parallel Programming*, San Jose, CA, Mar. 2007.

[27] M. F. Spear, A. Shriraman, L. Dalessandro, S. Dwarkadas, and M. L. Scott. Nonblocking Transactions Without Indirection Using Alert-on-Update. In *Proc. of the 19th Annual ACM Symp. on Parallelism in Algorithms and Architectures*, San Diego, CA, June 2007.