# Distributed Computing Column 48
## *Annual Review 2012*

Idit Keidar
Dept. of Electrical Engineering, Technion
Haifa, 32000, Israel
idish@ee.technion.ac.il

As usual, I conclude the year with an annual review of distributed computing awards and conferences. I begin by reporting on two prestigious awards - the Dijkstra Prize and the Principles of Distributed Computing Doctoral Dissertation Award. I then proceed with reviews of the main two distributed computing conferences, PODC– the ACM Symposium on Principles of Distributed Computing– and DISC– the International Symposium on DIStributed Computing. Finally, the column includes a review of WTTM - The Fourth Workshop on the Theory of Transactional Memory.

The 2012 Edsger W. Dijkstra Prize in Distributed Computing was awarded to Maurice Herlihy, J. Eliot B. Moss, Nir Shavit, and Dan Touitou, (see picture), for their seminal work on transactional memory. The award recognizes two outstanding papers. The first, "Transactional Memory: Architectural Support for Lock-Free Data Structures" by Herlihy and Moss, appeared at the 20th Annual International Symposium on Computer Architecture (ISCA) in 1993; this paper introduced the idea of transactional memory and suggested how it could be supported in hardware. The second paper, "Software Transactional Memory" by Shavit and Touitou, appeared in the 14th ACM Symposium on Principles of Distributed Computing (PODC) in 1995 and in Distributed Computing in 1997; it showed how to realize the transactional memory concept in software when no hardware support is available.

The Dijkstra Prize is jointly awarded by PODC and DISC; it was awarded in PODC this year. The full award citation appears earlier in this issue of SIGACT News (and on the award's web page[1]), so I do not repeat it here. Instead, I include here the prize acceptance speech by Maurice Herlihy and Nir Shavit, where they review research on transactional memory and its deployment over the last two decades, and look ahead at its future evolution and adoption. Another glimpse at contemporary transactional memory research, at least on the theoretical side, can be found in

[1] http://www.podc.org/dijkstra/2012.html

the review of WTTM at the end of this column.

Left to right: J. Eliot B. Moss, Maurice Herlihy, Nir Shavit, and Dan Touitou receiving the Dijkstra Prize. Photo by Jukka Suomela.

Keren Censor-Hillel receiving the Doctoral Dissertation award from Faith Ellen. Photo by Chen Chen.

The Principles of Distributed Computing Doctoral Dissertation Award was given for the first time this year. Its recipient was Keren Censor-Hillel, for her 2010 thesis "Probabilistic Methods in Distributed Computing", supervised by Hagit Attiya at the Technion, Israel. The award statement appears below. Keren received the award at DISC this year (see picture).

Continuing a four-year tradition, I invited students who have won Best Paper or Best Student Paper Awards in PODC and DISC to review these conferences.

The review of PODC is by Siddhartha Sen of Princeton, who shared the Best Student Paper award with his co-author Alexander Jaffe and with Mika Göös. Siddhartha and Alexander were awarded for their paper "On the Price of Equivocation in Byzantine Agreement", co-authored with Thomas Moscibroda. Equivocation occurs when Byzantine processes send contradicting messages to different processes. When equivocation is ruled out, (e.g., using digital signatures or a global broadcast channel), the replication cost with $f$ faults can be reduced from $3f + 1$ to $2f + 1$. This paper constructs a range of models where the replication costs are $2f + 1, 2f + 2, \ldots, 3f$, by controlling the faulty processors' ability to equivocate. It does so by adding partial broadcast channels among sets of three processors (observing that equivocation is fundamentally an act between three parties). The paper gives asymptotically tight bounds on the number of necessary and sufficient 3-processor channels for Byzantine Agreement.

The PODC Best Paper Award was presented to George Giakkoupis and Philipp Woelfel for

"On the Time and Space Complexity of Randomized Test-And-Set".

The review of DISC is by Mika Göös of the University of Toronto. His review also covers the DISC tutorials and the co-located ADGA (Advances on Distributed Graph Algorithms) workshop. Remarkably, Mika has won awards both at PODC and DISC this year, for papers he wrote at the University of Helsinki.

Mika's award-winning paper in PODC is "Lower Bounds for Local Approximation", co-authored with Juho Hirvonen and Jukka Suomela. This work proves a general theorem showing that constant-time distributed algorithms cannot make use of numerical node identifiers when computing approximations to basic graph optimization problems, such as the maximum matching problem and the minimum dominating set problem. In fact, the symmetry breaking capabilities of these local algorithms are no better than the capabilities of algorithms that are run on anonymous port-numbered networks.

The Best Paper Award of DISC was presented to Mika Göös and Jukka Suomela for the paper titled "No Sublogarithmic-Time Approximation Scheme for Bipartite Vertex Cover". In this article, the authors consider distributed algorithms for computing close-to-optimal vertex covers on sparse 2-colored bipartite graphs. Though it is well known that the dual problem to minimum vertex cover (i.e., the maximum matching problem) admits constant-time approximation schemes, this paper shows, surprisingly, that no such fast algorithms exist for the vertex cover problem itself.

The Best Student Paper Award at DISC was awarded to Boris Korenfeld and Adam Morrison for their paper "CBTree: A Practical Concurrent Self-Adjusting Search Tree", co-authored with Yehuda Afek, Haim Kaplan, and Robert E. Tarjan.

The column concludes with a report on WTTM, the Fourth Workshop on the Theory of Transactional Memory, which was co-located with PODC this year. The review is by Vincent Gramoli and Alessia Milani. Many thanks to Maurice, Nir, Siddhartha, Mika, Vincent, and Alessia for their contributions!

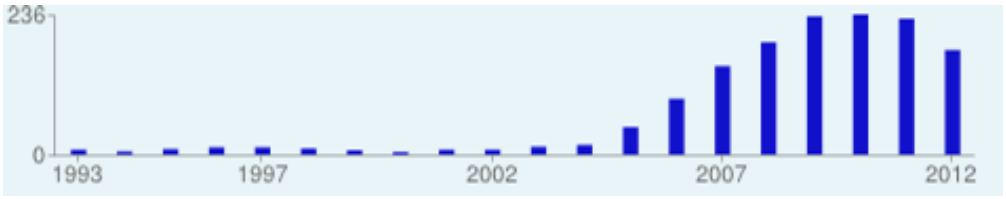
**Call for contributions:** I welcome suggestions for material to include in this column, including news, reviews, open problems, tutorials and surveys, either exposing the community to new and interesting topics, or providing new insight on well-studied topics by organizing them in new ways.

# Transactional Memory: Beyond the First Two Decades

Maurice Herlihy
Brown University
mph@cs.brown.edu

Nir Shavit
MIT and Tel-Aviv University
shanir@cs.tau.ac.il

The 2012 Dijkstra prize was awarded to two papers, *Transactional Memory: Architectural Support for Lock-Free Data Structures*, by Maurice Herlihy and Eliot Moss [4], published in 1993, and *Software Transactional Memory*, by Nir Shavit and Dan Touitou [8], published in 1995. Figure 1 shows the year-by-year citation graphs from Google Scholar for the past two decades. Though to some, these curves may look like hats, they actually represent the computer science community digesting a new idea. Transactional memory, to put it mildly, was slow to catch on.



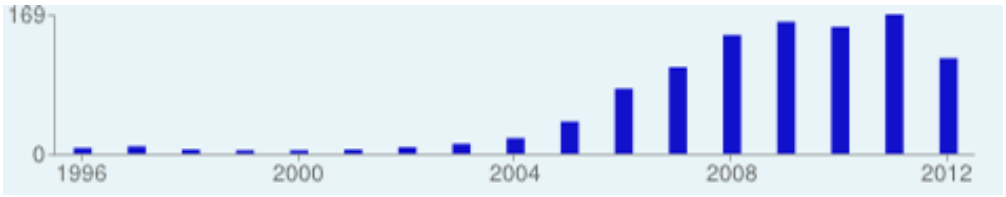




Figure 1: Hardware Transactional Memory (above) and Software Transactional Memory (below): citations per year.

Today, by contrast, transactional programming has become mainstream. Recently, Intel [5] and IBM [1] announced new processors with direct support for hardware transactional memory, and it seems likely that others will follow suit. Software transactional memory is present in a variety of languages, including Clojure [2], Scala [7] Haskell [3], Java [9], Python [6], and others, either through native language support or indirectly though libraries. An increasing number of compilers support transactional extensions to languages, most recently, in the new GNU C++ release [10].

This sweeping change did not happen by itself. It required a sequence of breakthrough ideas, by many in our community, to transform the ideas originating from those papers into a form ready for the market.

Transactional memory is a prime example of how basic research originating in the distributed computing community can influence the rest of the world. Just as important, however, it is an example of how the demands of the outside world have driven distributed computing research. Our community has worked long and hard to make these ideas a reality, and is to some extent in the position of the dog that spends all day chasing passing cars. Suddenly, one day, the dog catches a car. Now what?

Simply basking in success will not do. It should be the role of our community to ensure that the move to transactional models in hardware and software brings about a fundamental, positive change in the way people, especially non-specialists, program multicores.

The distributed computing community has the expertise to understand and invent the new algorithms and abstractions needed to to address this challenge. In particular, there is a need to develop new, transaction-friendly data structures and libraries that combine the best of both software and hardware techniques. There is a need to find ways for hardware transactions to compensate for software's drawbacks, and for software to compensate for the hardware's limitations. There is also a need to develop better transactional programming models, either as composable libraries of transactional objects, or as programming language abstractions. And, underneath it all, the effort requires a solid theoretical foundation, a task our our community is uniquely qualified to provide.

We have the car; it's time to chase trucks.

## References


[1] Bit-tech.net. IBM releases "world's most powerful" 5.5GHz processor. Retrieved from http://www.bit-tech.net/news/hardware/2012/08/29/ibm-zec12/1, 8 September 2012.

[2] Halloway, S. *Programming Clojure*. Pragmatic Programmers. Pragmatic Bookshelf, 2009.

[3] Harris, T., Marlow, S., Jones, S. L. P., and Herlihy, M. Composable memory transactions. *Commun. ACM 51*, 8 (2008), 91–100.

[4] Herlihy, M., and Moss, J. E. B. Transactional memory: architectural support for lock-free data structures. In *Proceedings of the 20th Annual International Symposium on Computer Architecture (ISCA'93)* (1993), ACM Press, pp. 289–300.

[5] Intel Corporation. Transactional Synchronization in Haswell. Retrieved from http:/http://software.intel.com/en-us/blogs/2012/02/07/transactional-synchronization-in-haswell/, 8 September 2012.

[6] Python Application Kit. Web. Trellisstm. Retrieved from http://peak.telecommunity.com/DevCenter/TrellisSTM, 20 November 2011.

[7] Scala STM Expert Group. Scalastm. web. Retrieved from http://nbronson.github.com/scala-stm/, 20 November 2011.

[8] Shavit, N., and Touitou, D. Software transactional memory. In *Proceedings of the fourteenth annual ACM symposium on Principles of distributed computing* (New York, NY, USA, 1995), PODC '95, ACM, pp. 204–213.

[9] The Deuce STM Group. Deuce stm - java software transactional memory. web. Retrieved from http://www.deucestm.org/documentation, 20 November 2011.

[10] Wiki, G. Transactional memory in gcc. Retrieved from http://gcc.gnu.org/wiki/TransactionalMemory, 8 September 2012.

## Principles of Distributed Computing Doctoral Dissertation Award

On Wednesday, October 17, 2012, the first Principles of Distributed Computing Doctoral Dissertation Award was given to Dr. Keren Censor-Hillel for her 2010 thesis "Probabilistic Methods in Distributed Computing", supervised by Professor Hagit Attiya at the Technion, Israel.

The main contribution of her thesis is proving that the total step complexity of asynchronous randomized consensus for $n$ processes is a quadratic function of $n$, solving a longstanding open problem. This result involved developing and analyzing a new shared coin algorithm that improved the previously best randomized consensus algorithm and creating a randomized valency technique that improved the lower bound. Another very interesting result in her thesis is the implementation of a counter with polylogarithmic individual step complexity, which she uses to derive a randomized consensus algorithm with linear individual step complexity. The thesis also contains a number of other related results, including the first randomized $k$-set agreement algorithm. As the nomination letter said, "The thesis has changed the way randomization is applied and analyzed in distributed computing, especially in the context of the consensus problem". The work in Keren's thesis has been presented in six conference papers: three at PODC and one at each of SODA, SPAA, and STOC. It has also led to five journal papers, including two in JACM and one in SICOMP. Her thesis is coherent and well organized. It is exceptionally well written, with clear explanations of technically sophisticated proofs.

Seventeen very strong theses were nominated for the award. The award committee consisted of Professor Faith Ellen (chair), Professor Pierre Fraigniaud, Professor Maurice Herlihy, Professor Friedhelm Meyer auf der Heide, Professor David Peleg, and Professor Sergio Rajsbaum. Dr. Tushar Chandra was a consultant to the committee.

# Review of PODC 2012

Siddhartha Sen
Department of Computer Science
Princeton University
sssix@cs.princeton.edu

The 31st Annual ACM Symposium on Principles of Distributed Computing (PODC) took place on July 16 - 18, 2012 in Madeira, Portugal, an archipelago in the Atlantic Ocean about 1,000 km off the European continent. This makes it arguably the most exotic location in PODC's 31-year history! When the view from your hotel and a stroll with your colleagues look like the pictures below, the conference transforms into a rejuvenating retreat, the kind that fosters great ideas and makes you forget (for a while) that you have to give that talk on the third day.

Figure 1: (left) A tourist-carrying galleon seen from the conference hotel, CS Madeira Atlantic Hotel and Sea Spa. (right) Walking by the water towards the conference banquet.

PODC was co-located with three interesting workshops: the 8th International Workshop on Foundations of Mobile Computing (FOMC), the 6th Workshop on Large Scale Distributed Systems and Middleware (LADIS), and the 4th Workshop on the Theory of Transactional Memory (WTTM). The workshops took place on July 18 - 19 and were attended by many of the PODC participants, who were happy to extend their stay in Madeira. This review focuses on the PODC conference.

In a nutshell, the PODC program consisted of the following: 2 keynote talks, one of which was held jointly with the LADIS workshop, 1 invited industry session, 11 full paper sessions, 3 brief announcements sessions, 3 lunches, 1 business meeting, 1 wine tasting, and 1 banquet. Amazingly, there was still time for participants to go for a swim in the ocean!

We begin by reviewing the research elements of the program, and then review the fun elements (including awards). We'll end by thanking the hardworking individuals who made PODC 2012 such a smooth and enjoyable experience.

## Research Program

While primarily a theoretical venue, in recent years PODC has become increasingly interested in the best practices of large-scale industry systems. This year's program featured an invited industry session with talks from Oracle Labs and Facebook, a keynote talk on bringing theoretical rigor to software-defined networking, and a keynote talk underscoring the difficulty of analyzing realistic wireless communication models. Indeed, bridging theory and practice is difficult to do in practice, and is something I am personally deeply interested in. The problem, as the invited speakers attested, is that often one side is more tractable or interesting than the other, leading to oversimplification or even neglect of the other side. Nevertheless, this kind of research is important because it keeps theory relevant and practice well-founded, and conferences like PODC are gradually making it more mainstream. All theory and no practice, and vice versa, make Jack a dull boy!

**Keynote talks.** David Peleg gave the first talk of the conference with his keynote "Towards an Algorithmically Usable SINR Model for Wireless Communication". In the signal to interference plus noise ratio (SINR) model, the energy of a wireless signal fades with some power of the distance, and a receiver successfully receives a message only if the signal is strong enough to overcome interference from simultaneous transmissions and background noise. SINR diagrams map the successful reception zones of transmitting stations in the plane; unfortunately, Peleg observed, they are theoretically not well understood. Most prior works study simplified models that abstract away any interference-related complications, making them easier to analyze but much less realistic. Peleg and his colleagues [2] brave the theory-practice gap and prove some basic properties about SINR diagrams, like the fact that reception zones are convex if transmissions have uniform power. They use this to develop an efficient approximation algorithm for answering point location queries.

Scott Shenker gave the second keynote, "Software-Defined Networking: History, Hype, and Hope", on the third day in a joint session with LADIS. This talk was not so much about bridging a theory-practice gap, as it was a call for theoretical rigor in a practice that has become too complicated and ad hoc. Software-defined networking (SDN) decouples the network's control plane logic from its data plane forwarding state, allowing software controllers like NOX [4] to program commodity network switches via interfaces like OpenFlow [7]. Shenker argues that the control plane has evolved to an ad hoc mess of protocols, and with networks hitting their complexity limits (a single datacenter can have 100,000 machines and 10,000 switches!), a theoretical intervention is needed. He believes the PODC community is well-poised to build abstractions for the control plane. Specifically, he wants abstractions for computing and specifying the distributed state of switches — state that correctly routes packets — subject to unreliable communication and failures.

Figure 2: (left) Peleg giving his keynote. (right) The daily lunch, with beautiful outdoor seating.

**Industry session** Two invited industry talks started the second conference day, and both presented interesting and sometimes unexpected facts from the field. Dave Dice from Oracle Labs spoke about practical synchronization techniques used in heavily-threaded Java software stacks. Locks are widely used here, techniques like barriers and lock-free structures are rare, and wait-free synchronization is nonexistent. Dice made an interesting observation that the locks used in practice are actually quite unfair; for example, admission may be strongly tied to cache coherence arbitration in hardware. These locks sacrifice short-term fairness for aggregate throughput, by favoring threads that are "hot" or resident in cache (context switches are expensive, costing 5,000+ cycles). One example are the cohort locks devised by Dice and his colleagues, which explicitly avoid lock migration across node sockets of a multi-core NUMA system.

Harry Li from Facebook gave the second talk, which was about practical consistency trade-offs at Facebook. Facebook has over a billion users, and uses an in-memory distributed cache that processes a billion operations per second. Their top priority is to ensure a fast, reliable, and consistent user experience. They use a geographically-diverse architecture of master and slave regions, where each region has its own web, cache, and database clusters. A slave region only serves read requests and uses MySQL replication to synchronize its database with the master, reducing a 70ms cross-country access to 2ms. Though Facebook only guarantees eventual consistency, they use a neat trick to ensure a user reads her writes: if the user recently updated data, her subsequent requests are redirected to the master region for some time. Li's talk reminded me of other geo-replicated storage systems like COPS [6] and Google's Spanner [3], which provide causal consistency and linearizability, respectively. Spanner runs Paxos over the wide area! It is fascinating how demands for stronger consistency are pushing traditionally local-area protocols to the Internet.

**Technical sessions** First, some statistics. PODC 2012 received 142 full paper submissions and accepted 35 (24.5%), compared to the 34/129 papers accepted in 2011. There were a total of 26 brief announcements. The most popular topics by submission were "fault tolerance", "wireless and mobile", and "graph algorithms". Topics like "cluster and cloud computing" and "internet and social networks" were quite low on the list, but will hopefully move up in future years. The top submitting countries by affiliation were the United States, Israel and France, the same as last year, but this year the 4th spot went to Spain instead of Switzerland.

We cannot possibly do justice to the wealth of ideas and contributions in the proceedings in this

short review. We refer the reader to [1] for the full papers and extended abstracts. What follows is a breeze through the various sessions with a few sprinkled comments from my perspective.

The first and last sessions of the conference were on shared memory. Aspnes gave an energetic talk on faster randomized consensus algorithms in a shared-memory model with an oblivious adversary. Reading the introductions of his paper was very useful, because the sheer number of models used in consensus can make your head spin otherwise. Giakkoupis and Woelfel study randomized test-and-set implementations in asynchronous shared memory models. Besides proving new time and space upper bounds, they also prove the first non-trivial space complexity lower bound for test-and-set. This paper won the Best Paper Award.

The session on information spreading and random walks included a paper on coalescing random walks, where independent random walks merge upon meeting at a graph vertex. The abstract idea of "coalescence" appears in a suprising number of fields, including population genetics and Bose-Einstein statistics.

The session on communication complexity discussed diverse problems, including distributed task allocation, multiparty communication subject to faults, and cooperative biological ensembles. The proofs in these papers are always quite neat because they involve a decoder Bob (with apparently psychic powers) who is able to do more than what is information-theoretically possible.

Continuing the practical spirit of the invited speakers, the session on wait freedom included an algorithm for solving a generalization of lattice agreement that facilitates building replicated state machines whose update commands commute. A variety of recent systems have leveraged the power of commutative data types [8] to avoid concurrency control and conflict resolution.

Given the rise of side-channel attacks on computer memory, such as the cold-boot attack [5], it was nice to see a paper on this topic in the session on game theory and security. Akavia *et al.* devise distributed public key schemes that are secure against continual memory leakage, even leakage that occurs while the secret keys are being refreshed.

In the session on locality, Göös *et al.* proved that for a large class of problems on bounded-degree graphs, including vertex covers and edge dominating sets, local algorithms (constant-time distributed algorithms) that achieve constant-factor approximations with the help of unique identifiers can do so without this help. Their paper was co-winner of the Best Student Paper award. Holzer and Wattenhofer also tackled a distributed graph problem in the session on distributed algorithms. They give an algorithm for all pairs shortest paths that runs on $O(n)$ rounds, which is optimal. They also present lower and upper bounds for approximating the diameter of a graph.

The session on ad-hoc networks covered problems in different wireless models. It is instructive to compare these models to the SINR model in Peleg's keynote, to understand the choices that were made to trade off of reality with tractability. The session on load balancing and scheduling also included a paper by Kesselheim on dynamic packet injection in wireless networks. The model he considers is quite general and covers virtually all interference models, including the SINR model.

The session on fault tolerance featured some elegant insights. Herlihy and Rajsbaum made the simple yet elegant observation that it is more important to know that one model of distributed computation simulates another, than to explicitly construct the simulation. They define simulation with respect to colorless tasks using combinatorial topology, and show how to prove they exist without finding them. Taubenfield generalized the traditional "all-or-nothing" notion of fault tolerance to count the number of correct processes that terminate properly. Studying a slew of classical problems with this new notion, he shows that some have solutions which guarantee that most correct processors properly terminate despite any number of faults, whereas others can't even


Figure 3: Bridging the theory-practice gap: (left) learning about Madeira wine; (right) tasting it.

guarantee that one correct processor properly terminates despite just one fault.

Moving to the malicious setting, the session on Byzantine Agreement analyzed the power of non-equivocation in reducing replication costs. Clement *et al.* showed that non-equivocation alone does not reduce the number of processors required for asynchronous reliable broadcast, but that the addition of transferable authentication (e.g. digital signatures) does. Sen *et al.* apply 3-processor partial broadcast channels to a variety of models and show that Byzantine Agreement is possible for all $n = 2f+1, 2f+2, \ldots, 3f$, where $n$ is the number of processors needed to tolerate $f$ faults. They give asymptotically tight bounds on the number of necessary and sufficient 3-processor channels. Their paper was the other co-winner of the Best Student Paper award.

There were three brief announcement sessions spread over the three conference days. I particularly enjoy these sessions because they are like a rapid fire of ideas and topics aimed at your brain. Some of the problems considered include computing without any communication, queuing in highly dynamic networks, scalable secure multiparty communication, Internet-scale and human computing, a tight lower bound for randomized mutual exclusion, network formation games that generate realistic networks, and failure detectors that equalize models of computation. If you are tired after reading that list, then you know what it's like to sit in one of these sessions!

## Fun Program

The organizers of PODC interspersed fun activities throughout the technical program, giving participants time to socialize, enjoy the outdoors, and digest their meals and theorems. Even the daily buffet lunches were exalted by excellent views from the hotel's seaside facade. Below is a photo-guided run-down of the fun events at PODC!

**Dijsktra award** Yes, awards are part of the fun program, because the hard work has already been done! This year, the prestigious Edsger W. Dijkstra Prize in Distributed Computing was awarded at PODC. The prize is given to oustanding papers on the principles of distributed computing whose impact has been evident for over a decade. The 2012 recipients were Maurice Herlihy and J. Eliot B. Moss for their paper "Transactional Memory: Architectural Support for Lock-Free Data Structures"; and Nir Shavit and Dan Touitou for their paper "Software Transactional Memory".

Figure 4: (left) Rooftop cocktail hour before dinner. (right) The dinner.

Given the pervasive impact of transactional memory today, from software runtimes to compilers like gcc to hardware implementations like Intel's, this award could truly not be more deserved.

**Wine tasting** At the end of the first day, PODC participants were escorted to a wine tasting at the Instituto Do Vinho Da Madeira. There, we tasted the famous Madeira wine, both sweet and dry versions, and heard a presentation about the wine's history. Madeira wine is uniquely known for its *estufagem* aging process, which simulates the effect of long sea voyages through tropical climates. Being a fortified wine, the estufagem process can last up to 100 years! A wine tasting before dinner implies rapid inebriation, so it was an interesting (random) walk (stumble) back home.

**Business meeting** Our PC Chair Alessandro Panconesi ran the business meeting, which was full of statistics about the submission process and the budget, accompanied by interesting local refreshments. The student grant program this year was particularly generous, more so than any conference (both theory and systems) I have been to.

At the meeting, Alexander Shvartsman was unanimously elected as chair of the steering committee, replacing Andrzej Pelc. Looking forward, Gadi Taubenfeld will be the PC Chair of PODC 2013, which will take place in Montreal, Quebec, Canada, on July 22 - 24.

**Banquet and awards** The conference banquet was held at the nearby Restaurante do Forte. The dinner began with a cocktail hour atop the restaurant, which had fantastic views of Madeira and the ocean. As with all other meals at the conference, the banquet was filled with good food and good company. Around dessert time, Alessandro Panconesi announced this year's award papers. The Best Student Paper award was given to two papers: Mika Göös, Juho Hirvonen, and Jukka Suomela received it for their paper "Lower Bounds for Local Approximation"; and Alexander Jaffe, Thomas Moscibroda, and Siddhartha Sen received it for their paper "On the Price of Equivocation in Byzantine Agreement". The awards were given by Maria da Graça Luís of Madeira's Regional Secretariat for Culture, Tourism, and Transportation, whose presence elevated the event.

Though technically not part of PODC's fun program, an interesting (but dramatic) event took place the following day during the LADIS banquet. A fire broke out atop the Madeira hills, and while the locals told us this was a common occurrence, it eventually spread to an unprecedented

Figure 5: A massive fire broke out on the island on July 18, seen here from the LADIS banquet.

size. Fortunately, no one was injured during the event, though several homes were affected. It was an amazing thing to see.

## Thank You

PODC would not have been the fantastic conference it was without the work of Alessandro Panconesi and the Program, Conference, and Steering Committee members. Special thanks go to Oksana Denysyuk and Luís Rodrigues for their amazing job with the local arrangements. They were visible at every corner, making things run smoothly for everyone. We can't thank all of you enough!

## References

[1] *PODC '12: Proceedings of the 2012 ACM symposium on Principles of distributed computing*, 2012. General Chair: Dariusz Kowalski; Program chair: Alessandro Panconesi.

[2] C. Avin, Y. Emek, E. Kantor, Z. Lotker, D. Peleg, and L. Roditty. SINR diagrams: Convexity and its applications in wireless networks. *Journal of the ACM*, 59(4):18, 2012.

[3] J. Corbett, J. Dean, M. Epstein, C. Frost, J. Furman, S. Ghemawat, A. Gubarev, C. Heiser, P. Hochschild, W. Hsieh, S. Kanthak, E. Kogan, H. Li, A. Lloyd, D. Mwaura, D. Nagle, S. Quinlan, R. Rao, L. Rolig, Y. Saito, M. Szymaniak, C. Taylor, R. Wang, and D. Woodford. Spanner: Google's globally-distributed database. In *Proc. Symposium on Operating System Design and Implementation*, 2012.

[4] N. Gude, T. Koponen, J. Pettit, B. Pfaff, M. Casado, N. McKeown, and S. Shenker. NOX: towards an operating system for networks. *ACM Computer Communication Review*, 38(3):105–110, 2008.

[5] J. A. Halderman, S. D. Schoen, N. Heninger, W. Clarkson, W. Paul, J. A. Calandrino, A. J. Feldman, J. Appelbaum, and E. W. Felten. Lest we remember: cold-boot attacks on encryption keys. *Communications of the ACM*, 52(5):91–98, 2009.

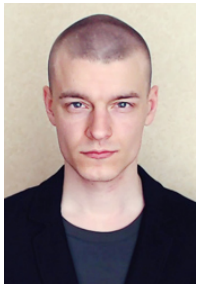
[6] W. Lloyd, M. J. Freedman, M. Kaminsky, and D. G. Andersen. Don't settle for eventual: scalable causal consistency for wide-area storage with COPS. In *Proc. ACM Symposium on Operating Systems Principles 2011*, pages 401–416, 2011.

[7] N. McKeown, T. Anderson, H. Balakrishnan, G. M. Parulkar, L. L. Peterson, J. Rexford, S. Shenker, and J. S. Turner. Openflow: enabling innovation in campus networks. *ACM Computer Communication Review*, 38(2):69–74, 2008.

[8] M. Shapiro, N. Preguica, C. Masquero, and M. Zawirski. Conflict-free replicated data types. In *International Symposium on Stabilization, Safety and Security of Distributed Systems*, pages 386–400, 2011.

# Review of DISC 2012

Mika Göös
Department of Computer Science
University of Toronto
mika.goos@mail.utoronto.ca



The 26th International Symposium on Distributed Computing (DISC 2012) was held on 16–18 October in sunny Salvador, Brazil. The conference was organized by the Distributed Systems Laboratory (LaSiD) of the Federal University of Bahia (UFBA). Some 90 people (local organization included) attended the conference.

The technical programme consisted of 4 tutorials, 27 regular paper presentations (27 papers were accepted out of 112 submissions), 24 brief announcements and 2 keynote presentations. This year a workshop on Advances on Distributed Graph Algorithms (ADGA) was also co-located with DISC.

**Tutorials.** On Monday, the day before the conference, some of the attendees who had arrived early—mostly junior members—were treated to a series of comprehensive 150 minute tutorials at the Mathematics Institute of UFBA.

Nicola Santoro gave the first tutorial on the subject of *mobile agents/robots* [4]. In the *discrete* setting, multiple asynchronous agents move about a predefined graph with the goal of accomplishing some global task. Example problems include detecting a "black hole" (a byzantine node) and network decontamination. In the *continuous* setting, the agents are navigating a geometric world (e.g., a plane) with a limited sense of direction. Example problems include pattern formation and sensor placement.

Paulo Veríssimo followed up with a systems-oriented account of *intrusion tolerance and resilience* [8]. Paulo argued that the classical Byzantine fault tolerance models are too simplistic to give practically useful guarantees for real-world systems: modern systems need to persist over long periods of time against continual attacks. Towards a solution, Paulo proposed a property called *exhaustion-safety* that gives new formal guarantees for real-time systems.

Elias Duarte gave an enthusiastic survey of *system-level diagnosis* [2, 6]. Introduced by Preparata, Metze, and Chien already in 1967 this high-level type of diagnosis views the system as consisting of indecomposable units that can perform *tests* on one another. The objective is to find out which

of the units are faulty with the minimum number of tests, even under the assumption that faulty units may report arbitrary test results.

Michel Raynal (whose tutorial was rescheduled to Thursday) talked about *concurrent programming* [7, 5] for asynchronous shared memory machines—the bread and butter of the "red" side of distributed computing. Topics covered included wait-freedom, linearizability, and basic problems such as renaming and set agreement. Michel has a new textbook coming out [7].

**Paper presentations.** The main venue for DISC was the Pestana Bahia conference hotel. Pestana Bahia stands uncontested overlooking the Rio Vermelho neighbourhood on the Atlantic coast, complete with postcard-like views from the hotel room windows.

What follows is my rather random (and biased) sample of the many papers presented at the conference.

The very first presentation on Tuesday morning was given by Boris Korenfeld who, together with his student co-author Adam Morrison, won the Best Student Paper Award. Their work (also co-authored by Afek, Kaplan, and Tarjan) described a new binary search tree, called CBTree (for *counting-based tree*), that makes frequently accessed items easily available while modifying the tree structure only infrequently. This allows for efficient *concurrent* implementations of the data structure: the authors show experimentally that CBTree outperforms existing concurrent search tree implementations on real-life workloads. (Outside of this work, concurrent search trees were popular this year: there were at least two brief announcements studying them.)

In other shared memory related news, the talk by Alexey Gotsman (with co-authors Musuvathi and Yang) on interoperability between high-level clients and low-level libraries stood out as being highly relevant to common programming practice. On the more theoretical side, Ami Paz (with Attiya) presented new alternative impossibility proofs for renaming and set agreement that avoid the usual algebraic topology framework. Transactional memory had its foothold with Alexander Matveev (with Afek and Shavit) introducing a new approach to replacing read-write locks with transactional code.

Robots and mobile agents were trending this year with their own dedicated session. Among the topics studied were: how to gather information in a graph when the robots have a limited battery life; oblivious algorithms for asynchronous pattern formation that work under any (feasible) initial positions of the robots; learning the initial positions in case the robots can sense their environment only through bouncing off of each other.

Wireless networks and the SINR model received attention in several works. For example, Keren Censor-Hillel (with Haeupler, Lynch, and Médard) gave an interesting talk on using linear network coding to allow error-free communication even under channel contention.

My own area of distributed graph algorithms was well-represented, too. In our paper (with Jukka Suomela) we established run-time lower bounds for finding small vertex covers on bipartite graphs; for this work we received the Best Paper Award. Christoph Lenzen (with Dolev and Peled) presented fast algorithms for finding triangles in dense graphs. Michal Hańćkowiak (with Czygrinow, Szymańska, and Wawrzyniak), in turn, designed local algorithms for computing approximations to the semi-matching problem. Merav Parter (with Fraigniaud, Korman, and Peleg) continued the study of randomness in the context of locally decidable problems. Finally, Fabian Kuhn (with Haeupler) gave a measured talk on their extensions to prior lower bounds on token dissemination in time evolving graphs.

**Keynotes.** Our first keynote speaker was Yehuda Afek (see picture). He shared his experiences in setting up a start-up company (called Riverhead Networks) that specialized in preventing DDoS attacks. In short, their solution was to integrate new hardware into the internet backbone, which, together with some sophisticated filtering software, is able to divert malicious traffic targeted at their customers' web sites. The start-up turned out very successful (it was subsequently acquired by Cisco). Here is Yehuda's 7-step programme to success: To start, one needs 1) an idea, and 2) some high quality people to implement it. An important factor is 3) timing, as it helps to be the first one on the scene—failing this, one needs 4) determination to pursue under pressure from competitors. Some other qualities that come in handy are 5) (social) networking skills, 6) experience, and 7) luck.


Yehuda Afek giving his keynote talk.
Photo by Chen Chen.

The second keynote was given by the energetic Simon Peyton-Jones whose research typically revolves around functional programming. Being a newcomer to DISC Simon assumed the role of an ambassador for Haskell as he described their first-ever message passing library, termed *Haskell Cloud*, for running Haskell programs on large-scale networks (e.g., data-centre computing). Their implementation is using Haskell's type system while borrowing ideas from Erlang [3]. Given that people in the DISC community sometimes skip implementing their algorithms, Simon invited the audience to consider Haskell Cloud as an appropriate easy-to-use implementation platform.

**Social event.** On Wednesday afternoon we boarded a buss towards the Pelourinho, the old city centre, which is a UNESCO world heritage site. Our tour guide was sure to point out the historical significance of the place, being, as it was, among the first cities founded by the Portuguese settlers. After touring around some of the impressive Pelourinho churches we made our way to *Restaurante Amado* where the conference banquet was held. During the banquet—and between *Caipirinhas*—the first Principles of Distributed Computing Doctoral Dissertation Award was granted to Keren Censor-Hillel (see picture above).

**The ADGA workshop.** On Friday, the day after the conference, a good many people turned out for the Advances on Distributed Graph Algorithms workshop organized by Amos Korman. The workshop consisted of 5 presentations.

The talk by Fabian Kuhn had the widest scope. Fabian surveyed the basic results on distributed graph algorithms. (He was careful to mention many of the works by people sitting in the audience.) Example problems include maximal independent set, approximating the maximum dominating set, vertex colouring, and network decomposition. Here is a major open problem: derandomize any of the polylogarithmic-time algorithms for, e.g., network decomposition—or prove lower bounds establishing that these tasks cannot be solved in deterministic polylogarithmic time.

Pierre Fraigniaud talked about their recent OPODIS article that studies whether unique node identifiers help for locally decidable problems. Yuval Emek was interested in applying tools from distributed computing to understand the "computations" that take place inside biological systems (for a motivational TED talk, see [1]). Christian Scheideler gave an introduction to self-stabilizing distributed data structures. Here, the data structures are represented as graphs that, after some link failures, eventually stabilize through a series of local operations on the graph topology. Cyril Gavoille reviewed the known bounds on labelling schemes. For example, how many bits per node are needed to decide whether two nodes are adjacent based only on their labels?

**Acknowledgements.** I thank Chen Chen for providing the photos.

## References

[1] Bonnie Bassler. How bacteria "talk". http://www.ted.com/talks/bonnie_bassler_on_how_bacteria_communicate.html, February 2009. TED.

[2] Elias P. Duarte, Jr., Roverli P. Ziwich, and Luiz C.P. Albini. A survey of comparison-based system-level diagnosis. *ACM Computing Surveys*, 43(3):22:1–22:56, April 2011.

[3] Jeff Epstein, Andrew P. Black, and Simon Peyton-Jones. Towards haskell in the cloud. In *Proceedings of the 4th ACM symposium on Haskell*, pages 118–129. ACM, 2011.

[4] Paola Flocchini, Giuseppe Prencipe, and Nicola Santoro. *Distributed Computing by Oblivious Mobile Robots*. Synthesis Lectures on Distributed Computing Theory. Morgan & Claypool Publishers, 2012.

[5] Maurice Herlihy and Nir Shavit. *The Art of Multiprocessor Programming*. Morgan Kaufmann, 2008.

[6] Gerald M. Masson, Douglas M. Blough, and Gregory F. Sullivan. System diagnosis. In Dhiraj K. Pradhan, editor, *Fault-Tolerant Computer System Design*, pages 478–536. Prentice Hall, 1996.

[7] Michel Raynal. *Concurrent Programming: Algorithms, Principles, and Foundations*. Springer, 2012.

[8] Paulo Veríssimo. Travelling through wormholes: a new look at distributed systems models. *SIGACT News*, 37(1):66–81, 2006.

# WTTM 2012, The Fourth Workshop on the Theory of Transactional Memory

Vincent Gramoli
University of Sydney
Australia
vincent.gramoli@sydney.edu.au

Alessia Milani
Université de Bordeaux-IPB-Labri
France
milani@labri.fr

**Abstract**

In conjunction with PODC 2012, the TransForm project (Marie Curie Initial Training Network) and EuroTM (COST Action IC1001) supported the 4th edition of the Workshop on the Theory of Transactional Memory (WTTM 2012). The objective of WTTM was to discuss new theoretical challenges and recent achievements in the area of transactional computing. The workshop took place on July 19, 2012, in Madeira, Portugal.

This year's WTTM was a milestone event for two reasons. First, because the same year, the two seminal articles on hardware and software transactional memories [15, 21] were recognized as outstanding papers on principles of distributed computing, whose significance and impact on the theory and practice of distributed computing have been evident for at least a decade. Second, the winners of this prestigious ACM/EATCS Edsger W. Dijkstra Prize, Maurice Herlihy, Eliot Moss, Nir Shavit and Dan Touitou, were present at the workshop and three of them discussed their current progress with other outstanding researchers from the field. This report is intended to give highlights of the problems discussed during the workshop.

Transactional memory is a concurrency control mechanism for synchronizing concurrent accesses to shared memory by different threads. It has been proposed as an alternative to lock-based synchronization to simplify concurrent programming while exhibiting good performance. The sequential code is encapsulated in transactions, which are sequences of accesses to shared or local variables that should be executed atomically by a single thread. A transaction ends either by *committing*, in which case all of its updates take effect, or by *aborting*, in which case, all its updates are discarded and never become visible to other transactions.

## 1 Consistency criteria

Since the introduction of the transactional memory paradigm, several consistency criteria have being proposed to capture its correct behavior. Some consistency criteria have been inherited from

the database field (e.g., serializability, strict serializability), others have been proposed to extend these latter to take into account aborted transactions, e.g., opacity, virtual world consistency; some others have been proposed to define the correct behavior when transactions have to be synchronized with non transactional code, e.g., strong atomicity.

Among all the criteria, opacity, originally proposed by Guerraoui and Kapalka [13], gained a lot of attention. In his first talk, Srivatsan Ravi showed that in any safe restriction of opacity, no transaction may read from a transaction that has not invoked *tryCommit*. He presented a definition of opacity that intuitively captures this feature and showed that it is a safety property in the formal sense, i.e., it is prefix-closed and limit-closed. This is a joint work with Hagit Attiya, Sandeep Hans and Petr Kuznetsov.

Victor Luchangco presented his joint work with Mohsen Lesani and Mark Moir provocatively titled "Putting opacity in its place" in which he presented the TMS1 and TMS2 consistency conditions and clarified their relationship with the prefix-closed definition of opacity. Broadly, these conditions ensure that no transaction observes the partial effects of any other transaction in any execution of the STM without having to force a total-order on transactions that participate in the execution. In particular, TMS1 is defined for any object with a well-defined sequential specification while TMS2 is specific for read-write registers. They further show using IO Automata [18] that every behavior allowed by TMS2 is allowed by TMS1 and formally prove that the NOrec algorithm satisfies TMS2. Moreover, algorithms that satisfy opacity also satisfy TMS1, and algorithms that satisfy TMS2 also satisfy opacity.

While opacity defines the correctness of transactional memories when shared variables are accessed only inside transactions, understanding the interaction between transactions and locks or between accesses to shared variables in and outside transactions has been a major question. This is motivated by the fact that the code written to work in a transactional system may need to interact with legacy code where locks have been used for synchronization. It has been shown that replacing a lock with a transaction does not always ensure the same behavior.

Srivatsan Ravi presented his joint work with Vincent Gramoli and Petr Kuznetsov on the locally-serializable linearizability (ls-linearizability) consistency criterion that applies to both transaction-based and lock-based programs, thus allowing to compare the amount of concurrency of a concurrent program [10]. In short, this consistency criterion captures the correctness of high-level data types (linearizability) with the correctness of their low-level implementations (thread-safety) by guaranteeing that the sequence of sequential events corresponding to each high-level operation is consistent with "some" sequential execution.

Stephan Diestelhorst presented his preliminary work with Martin Pohlack, "Safely Accessing Timestamps in Transactions". He presented scenarios in which the access to the CPU timestamp counter inside transactions could lead to unexpected behaviors. In particular, this counter is not a transactional variable, so reading it inside a transaction can lead to a violation of the single lock atomicity semantics: multiple accesses to this counter within transactions may lead to a different result than multiple accesses within a critical section. In this talk, a solution to prevent several transactions from accessing the timestamp concurrently was also sketched.

Annette Bienusa presented the definition of snapshot trace to simplify the reasoning on the correctness of TMs that ensure snapshot isolation. This is a joint work with Peter Thiemann [5].

Finally, Faith Ellen stated that despite the fact that a rich set of consistency criteria have been proposed there is no agreement on the way the semantics of a transactional memory has to be defined: operationally [14], as a set of executions [13, 22], or using automata [7].

## 2 Data structures for transactional computing

Eliot Moss' talk intended to explore how the availability of transactions as a programming construct might impact the design of data types. He gave multiple examples on how to design data types having in mind that these types are going to be used in transactions.

In a similar direction, Maurice Herlihy considered data types that support high-level methods and their inverse. As an example a set of elements supports the method to add an element, $add(x)$, and the method to remove an element from the set $remove(x)$. For these kind of data types, he discussed the possibility to apply a technique called *transactional boosting* which provides a modular way to make highly concurrent thread-safe data structures transactional. He suggested to distinguish the transaction-level synchronization with the thread-level synchronization. In particular, to synchronize the access to a linearizable object, non-commutative method calls have to be executed serially (e.g., $add(x)$ and $remove(x)$). Two method calls are commutative if they can be applied in any order and the final state of the object does not change. For example, $add(x)$ and $remove(x)$ do not commute, while $add(x)$ and $add(y)$ commute. Since methods have inverses, recovery can be done at the granularity of methods. This technique exploits the object semantics to synchronize concurrent accesses to the object. This is expected to be more efficient than STM implementations where consistency is guaranteed by detecting read/write conflicts.

## 3 Performance

Improving the efficiency of TMs has been a key problem for the last few years. In fact, in order for transactional memory to be accepted as a candidate to replace locks, we need to show that it has performance comparable to these latter.

In her talk Faith Ellen summarized the theoretical results on the efficiency of TMs. She stated that efficiency has been considered through three axes: properties that state under which circumstances aborts have to be avoided (permissiveness [11], progressiveness [12], etc); progress/liveness conditions and parallelizability. This latter is formalized through different variants of the disjoint access parallelism (DAP) property [16, 8, 2]. She also summarized the known impossibility results on TMs, parametrized by the consistency criterion, conditions under which transactions are allowed to abort and the nature of disjoint-access parallelism allowed by the implementation. The talk also summarized the lower bounds on the complexity to implement a combination of semantics, progress and DAP properties.

Mykhailo Laremko discussed how to apply known techniques (e.g., combining) to boost the performance of existing STM systems that have a central point of synchronization. In particular, in his joint work with Panagiota Fatourou, Eleftherios Kosmas and Giorgos E. Papadakis, they augment the NOrec transactional memory by combining and replacing the single global lock with a set of locks. They provide preliminary simulation results to compare NOrec and its augmented versions, showing that these latter perform better.

Nuno Diegues with João Cachopo [6] study how to extend a transactional memory to support nested transactions efficiently. The difficulty is to take into account the constraints imposed by the baseline algorithm. To investigate different directions in the design space (lazy versus eager conflict detection, multiversion versus single version etc), they consider the following transactional memories: JVSTM[9], NesTM [3] and PNSTM[4] . The performance of these algorithms were compared considering workloads without nested transactions and workloads with nested transactions. Nuno

Diegues shows that PNSTM's throughput is not affected by parallel nesting, while it is the case for the throughput of JVSTM and NesTM. In particular, NesTM shows the greater degradation of performance w.r.t. the depth of the nesting.

## 4 Legacy code and hardware transactional memory

The keynote by Nir Shavit was about his joint work with Yehuda Afek and Alex Matveev on "Pessimistic transactional lock-elision (PLE)" [1]. PLE is a non-speculative technique for automatic replacement of read-write locks by STM code in which each transaction is executed only once and never aborts. The main idea is to organize concurrency to avoid the synchronization overhead. In particular, write transactions execute sequentially and maintain a public undo log such that read operations can collect a snapshot of the memory using it. At commit time, writing transactions check that no transaction is reading the values they have to update. Then the old values are discarded. Their STM implementation offers significant performance improvements over read-write locks while performing only marginally worse than optimistic STMs like TL2 on some workloads. His talk further touched upon how to integrate such a mechanism to compliment hardware support like Intel's HLE. The inherent cost of lock elision remains an open problem.

Maged Michael gave a talk on IBM BlueGene/Q HTM features and performance characteristics [23]. Similar to the Intel HLE, he pointed out that there are no guarantees given on transaction progress and described the implementation of an STM on top of the BG/Q HTM.

## 5 Distributed transactional memory

Distributed transactional memory is the implementation of the transactional memory paradigm in a networked environment where processes communicate by exchanging messages. Differently from transactional memory for multicore machines, the networked environment needs to take into account the non negligible communication delays. To support local accesses to shared objects, distributed transactional memories usually rely on replication. Pawel T. Wojciechowski presented his joint work with Jan Konczak. They consider the problem of recovering the state of the shared data after some node crashes. This requests to write data into stable storage. Their goal was to minimize writes to stable storage, which are slow, or to do them in parallel with the execution of transactions. He presented a crash-recovery model for distributed transactional memory which is based on deferred update replication relying on atomic broadcast. Their model takes into account the tradeoff between performance and fault-tolerance. See [24, 17] for more information.

Sebastiano Peluso claimed that efficient replication schemes for distributed transactional memory have to follow three design principles: partial replication, genuineness to ensure scalability and support for wait-free read-only transactions. According to genuineness the only nodes involved in the execution of a transaction are ones that maintain a replica of an object accessed by the transaction. He claimed that genuineness is a fundamental property for the scalability of distributed transactional memory. This is a joint work with Paolo Romano and Francesco Quaglia [19]. Additional information can be found in [20].

## 6 Conclusion

While transactional memory has become a practical technology integrated in the hardware of the IBM BlueGene/Q supercomputer and the upcoming Intel Haswell processors, the theory of transactional memory still misses good models of computation, good complexity measures, agreement on the right definitions, identification of fundamental and useful algorithmic questions, innovative algorithm designs and lower bounds on problems. Upcoming challenges will likely include the design of new transactional algorithms that exploit the low-overhead of low-level instructions on the one hand, and the concurrency of high-level data types on the other hand.
For further information the abstracts and slides of the talks can be found at http://sydney.edu.au/engineering/it/~gramoli/events/wttm4.

**Acknowledgements** We are grateful to the speakers, to the program committee members of WTTM 2012 for their help in reviewing this year's submissions and to Panagiota Fatourou for her help in the organization of the event. We would like to thank Srivatsan Ravi and Mykhailo Laremko for sharing their notes on the talks of the workshop.

## References

[1] Y. Afek, A. Matveev, N. Shavit. Pessimistic Software Lock-Elision. In Proceedings of the 26th International Symposium on Distributed Computing, DISC 2012.

[2] H. Attiya, E. Hillel, and A. Milani. Inherent Limitations on Disjoint-Access Parallel Implementations of Transactional Memory. In Proceedings of the 21nd ACM symposium on Parallelism in algorithms and architectures, SPAA 2009, pages 69–78.

[3] W. Baek, N. Bronson, C. Kozyrakis, and K. Olukotun. Implementing and evaluating nested parallel transactions in software transactional memory. In Proceedings of the 22nd ACM symposium on Parallelism in algorithms and architectures, SPAA 2010, pages 253-262.

[4] J. Barreto, A. Dragojevic , P. Ferreira, R. Guerraoui, and M. Kapalka. Leveraging parallel nesting in transactional memory. In Proceedings of the 15th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming, PPoPP 2010, pages 91-100.

[5] A. Bieniusa. Consistency, isolation, and irrevocability in software transactional memory, Phd Thesis, 2011, http://www.freidok.uni-freiburg.de/volltexte/8382/.

[6] N. Diegues and J. Cachopo. Exploring parallelism in transactional workloads. Technical Report RT/16/2012, INESC-ID Lisboa, June 2012.

[7] S. Doherty, L. Groves, V. Luchangco, and M. Moir. Towards Formally Specifying and Verifying Transactional Memory, REFINE 2009, pages 245–261.

[8] F. Ellen, P. Fatourou, E. Kosmas, A. Milani, and C. Travers. Universal Constructions that Ensure Disjoint-Access Parallelism and Wait-Freedom. In Proceedings of the 31st Annual ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing, PODC 2012, pages 115–124.

[9] S. M. Fernandes and J. Cachopo. Lock-free and scalable multi-version software transactional memory. In Proceedings of the 16th ACM symposium on Principles and practice of parallel programming, PPoPP 2011, pages 179-188.

[10] V. Gramoli, P. Kuznetsov, S. Ravi. Brief announcement: From sequential to concurrent: correctness and relative efficiency. In Proceedings of the 31st Annual ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing, PODC 2012, pages 241–242.

[11] R. Guerraoui, T.A. Henzinger, and M. Kapalka, Permissiveness in Transactional Memories. In Proceedings of the 22nd International Symposium on Distributed Computing, DISC 2008, pages 305–319.

[12] R. Guerraoui and M. Kapalka. The Semantics of Progress in Lock-Based Transactional Memory. In Proceedings of the 36th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2009, pages 404–415.

[13] R. Guerraoui and M. Kapalka. Principles of Transactional Memory, Morgan Claypool, 2010.

[14] T. Harris, J. Larus, and R. Rajwar. Transactional Memory, Morgan Claypool,2010.

[15] M. Herlihy, E. Moss. Transactional Memory: Architectural Support for Lock-Free Data Structures. In Proceedings of the 20th Annual International Symposium on Computer Architecture, ISCA 1993, pages 289–300.

[16] Israeli and Rappoport, Disjoint-access-parallel implementations of Strong Shared Memory Primitives. In Proceedings of the 13th Annual ACM Symposium on Principles of Distributed Computing, PODC 1994, pages 151–160.

[17] J. Kończak, N. Santos, T. Żurkowski, P. T. Wojciechowski and A. Schiper. JPaxos: State Machine Replication Based on the Paxos Protocol. Technical report EPFL-REPORT-167765, Faculté Informatique et Communications, EPFL, july 2011.

[18] M. Lesani, V. Luchangco, M. Moir: A Framework for Formally Verifying Software Transactional Memory Algorithms. In Proceedings of the 23rd International Conference on Concurrency Theory, CONCUR 2012, pages 516–530.

[19] S. Peluso, P. Romano, F. Quaglia. SCORe: a Scalable One-Copy Serializable Partial Replication Protocol. In Proceedings of ACM/IFIP/USENIX 13th International Middleware Conference, Middleware 2012.

[20] S. Peluso, P. Ruivo, P. Romano, F. Quaglia, and L. Rodrigues. When Scalability Meets Consistency: Genuine Multiversion Update Serializable Partial Data Replication. In Proceedings of the 32nd International Conference on Distributed Computing Systems, ICDCS 2012, pages 455–465.

[21] N. Shavit, D. Touitou. Software Transactional Memory. In Proceedings of the 14th Annual ACM Symposium on Principles of Distributed Computing, PODC 1995, pages 204–213.

[22] M. Spear, L. Dalessandro, Marathe, and M. Scott, Ordering-Based Semantics for Software Transactional Memory. In Proceedings of the 12th International Conference on Principles of Distributed Systems, OPODIS 2008, pages 275–294.

[23] A. Wang, M. Gaudet, P. Wu, J. N. Amaral, M. Ohmacht, C. Barton, R. Silvera, M. M. Michael. Evaluation of blue Gene/Q hardware support for transactional memories. In Proceedings of the 21st International Conference on Parallel Architectures and Compilation Techniques, PACT 2012, pages 127-136.

[24] P. T. Wojciechowski, T. Kobus, M. Kokociski. Model-Driven Comparison of State-Machine-based and Deferred-Update Replication Schemes. In Proceedings of the 31st IEEE International Symposium on Reliable Distributed Systems, SRDS 2012.