Distributed Computing Column 40
Annual Review 2010

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

As another year comes to a close, it is time for my traditional summary of this year's distributed computing awards and conferences. As reported in this column a year ago, Nancy Lynch received the prestigious 2010 IEEE Emanuel R. Piore Award[1] for her contributions to foundations of distributed and concurrent computing. Congratulations again Nancy!

The 2010 Edsger W. Dijkstra Prize in Distributed Computing was awarded to Tushar D. Chandra, Vassos Hadzilacos, and Sam Toueg, for their seminal work on unreliable failure detectors. The award recognizes two outstanding papers. The first, "Unreliable Failure Detectors for Reliable Distributed Systems", published in PODC in 1991 and in J. ACM in 1996, introduces the abstraction of an unreliable failure detector, and the concept of reducibility among classes thereof. A failure detector is a distributed oracle that provides processes with (possibly unreliable) information about failures. Such information is necessary for solving problems like fault-tolerant consensus in asynchronous systems, which we know from FLP to be otherwise impossible. Intuitively, this impossibility appears related to the fact that, in an asynchronous system, processes cannot distinguish a slow process from a faulty one. While earlier work on circumventing FLP introduced (partial) synchrony assumptions to enable such distinguishability, the failure detector abstraction provides processes with such information directly, thus "abstracting away" the need to assume synchrony. Furthermore, consensus is solvable with unreliable failure detectors, which may provide inaccurate information about failures. The second, paper "The Weakest Failure Detector for Solving Consensus", published in PODC in 1992 and in J. ACM in 1996, pinpoints the weakest failure detector for solving consensus, and thus identifies the minimal amount of information about failures required for circumventing the FLP result.

Both papers have made a tremendous impact on research in distributed computing; the first was cited over 2200 times, and the second has over 800 citations. The Dijkstra Prize is jointly awarded by PODC and DISC; Tushar, Vassos and Sam received it in PODC this year (see picture). The full Award Citation appears

[1] http://www.ieee.org/about/awards/tfas/piore.html

Left-to-right: Sam Toueg, Vassos Hadzilacos, and Tushar Chandra receive the Edsger W. Dijkstra Prize from Rachid Guerraoui. Photo by Jukka Suomela.

earlier in this issue of SIGACT News (and on the award's web page[2]), so I do not repeat it here. Instead, I include here the remarks made by Tushar Chandra at the Edsger W. Dijkstra Prize awarding ceremony.

The Prize for Innovation in Distributed Computing was awarded for the second time this year. The prize was established by SIROCCO to "recognize individuals whose research contributions on the relationships between information and efficiency in decentralized computing have expanded the collective investigative horizon by formulating new problems, or identifying new research areas, that were at the time of their introduction unorthodox and outside the mainstream." The prize is awarded in SIROCCO, and its second recipient is Jean-Claude Bermond. Bermond is recognized for a series of results showing the impact that network structure has on the efficiency of distributed algorithms running over these networks. These results span a wide variety of problems, for example, routing, broadcast, gossip, and fault tolerant network design. The full award citation appears below.

We proceed with reviews of the two leading conferences in distributed computing: PODC– the ACM Symposium on Principles of Distributed Computing– and DISC– the International Symposium on DIStributed Computing. Historically PODC was the North American of the two, and DISC European. But this year introduced a novelty, as PODC was held in Europe (Zurich, Switzerland), while DISC was held North America (Boston, USA). Continuing with past tradition, I invited students who have won Best Paper or Best Student Paper Awards in the respective conferences to review them.

The review of PODC is by Leonid Barenboim from Ben-Gurion University, who co-won the Best Paper Award for his paper "Deterministic Distributed Vertex Coloring in Polylogarithmic Time", co-authored with his advisor Michael Elkin. This paper solves a long-standing open problem, by showing, for the first time, a deterministic polylogarithmic vertex coloring algorithm that uses a number of colors $\ll \Delta^2$, where Δ is the maximum vertex degree in the graph. The authors use the same technique to show a number of other related results. Leonid and Michael shared the award with Valerie King and Jared Saia, who were recognized for their paper "Breaking the $O(n^2)$ Bit Barrier: Scalable Byzantine Agreement with an Adaptive Adversary". The results of the latter are discussed, among others, in Valerie and Jared's article in the previous Distributed

[2] http://www.podc.org/dijkstra/2010.html

Computing column (SIGACT News 41(3), September 2010, pp. 89–104).

DISC is reviewed by François Bonnet, who has won the Best Student Paper Award for his paper "Anonymous Asynchronous Systems: the Case of Failure Detectors" co-authored with his advisor Michel Raynal. While previous work on failure detectors focused on overcoming asynchrony, this paper extends their use to address also anonymity. Specifically, the paper introduces failure detectors for *anonymous* asynchronous systems, uses them for solving consensus, and discusses the notion of weakest failure detectors in this context. In both reviews, you will find fun information about the venues, as well as technical content.

The review of DISC is followed by a review of one of the workshops collocated with DISC this year—The Second Workshop on the Theory of Transactional Memory— by Srivatsan Ravi, Vincent Gramoli, and Victor Luchangco. Many thanks to Leonid, François, Srivatsan, Vincent, and Victor for their contributions!

Meanwhile, in this column, you may have read in 2010 about reconfiguring state machines (in March), the history of common knowledge (also in March), models for wireless networks (in June) and scalable Byzantine Agreement with many processes and among multiple clouds (in September). In 2011, stay tuned for future columns on dynamic graph models, round-based models for distributed quantum computing, and a game-theoretic approach to modeling various user behaviors. Best wishes for a fruitful 2011!

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.

Edsger W. Dijkstra Prize Acceptance Speech

Tushar D. Chandra

We would like to thank the distributed computing community for its support and the award committee for selecting our papers for the Edsger W. Dijkstra Prize. We are honored to join the list of highly distinguished researchers who previously received this award. We would briefly like to describe how this work grew on foundations built by the distributed computing community.

One of the core challenges in fault-tolerant distributed computing is handling the inherent uncertainty associated with one computer guessing whether another has failed. We felt that we needed an abstraction that captures this and came up with failure detectors.

We were surprised to find that a failure detector didn't have to be reliable to be useful - we found a surprisingly weak failure detector which we called $\Diamond W$ that can be used to solve Consensus. Our algorithm for solving consensus with this failure detector borrows from prior work, and in particular it is similar to several other rotating coordinator algorithms.

Regarding our weakest failure detector lower bound: from the onset we realized that failure detectors that seemed to be different could solve Consensus. We also realized that these seemingly different failure detectors could be compared using the well established notion of algorithmic reduction. The notion of reducibility allowed us to say what it means for a failure detector A to be stronger than another failure detector B. This led to a natural question - is there a weakest failure detector for solving Consensus? Or alternatively are there many incomparable minimal failure detectors for this problem? When we started our work, we were doubtful that we would find a weakest failure detector for solving Consensus - prior research had identified several seemingly incomparable models of partial synchrony in which consensus is solvable, but not a weakest model.

To prove the weakest failure detector result, we observed that any failure detector that solves Consensus has to break the bivalency argument of Fischer, Lynch and Paterson. Intuitively, given a failure detector and a corresponding Consensus algorithm, we simulated runs of the algorithm using the failure detector and observed where the bivalency argument of Fischer, Lynch and Paterson broke down. This breakdown allowed us to extract a leader that gave us the failure detector that we wanted.

In summary, we would like to thank the award committee for recognizing our work and also thank the distributed computing community for providing the foundation that enabled our work.

Prize 2010 for Innovation in Distributed Computing
awarded to
Jean-Claude Bermond

The Prize for Innovation In Distributed Computing is awarded by the Colloquium on Structural Information and Communication Complexity (SIROCCO). It is established to recognize individuals whose research contributions on the relationships between information and efficiency in decentralized computing have expanded the collective investigative horizon by formulating new problems, or identifying new research areas, that were at the time of their introduction unorthodox and outside the mainstream. The prize recognizes originality, innovation, and creativity. The recipient of the Prize is chosen among the nominated persons for the current year.

The Award Committee has selected **Jean-Claude BERMOND** as the recipient of this year's Prize for Innovation In Distributed Computing.

The prize is given in recognition of Bermond for his contribution to the study of the impact of structure of networks on the efficiency of parallel or distributed algorithms, as illustrated by several papers, including some that appeared in the proceedings of past SIROCCO meetings. These papers tackled a wide variety of problems including routing, broadcasting, gossip protocols, traffic grooming, fault tolerant network design, monopolies, and other topics, illustrated, in particular, by the following papers:

- J.-C. Bermond, L. Chacon, D. Coudert, and F. Tillerot. Cycle Covering. In Proc. SIROCCO 2001: 21-34.
- J.-C. Bermond, B. Jackson, F. Jaeger. Shortest coverings of graphs with cycles. J. Comb. Theory, Ser. B 35(3): 297-308 (1983)
- J.-C. Bermond, C. Peyrat. De Bruijn and Kautz networks: a competitor for the hypercube? In Proc. 1st European Workshop on Hypercubes and Distributed Computers: 279-293 (1989).
- J.-C. Bermond, Stephane Perennes. Efficient Broadcasting Protocols on the de Bruijn and Similar Networks. In Proc. SIROCCO 1995: 199-209
- J.-C. Bermond, L. Gargano, A. Rescigno, U. Vaccaro. Fast Gossiping by Short Messages. SIAM J. Comput. 27(4): 917-941 (1998)
- J.-C. Bermond, P. Fraigniaud. Broadcasting and Gossiping in de Bruijn Networks. SIAM J. Comput. 23(1): 212-225 (1994)

The paper *De Bruijn and Kautz networks: a competitor for the hypercube?* is a pioneering paper investigating the design of networks for parallel and distributed computers. This paper promoted the de Bruijn graph as a suitable network for efficient communication and management. Under the reign of the hypercubes and meshes tightly-coupled multi-processors, this paper was definitively unorthodox. It was visionary too, as the de Bruijn graph was later found particularly rich in applications, especially in the framework of overlay network design for P2P systems.

The paper *Cycle Covering* appeared in SIROCCO 1995 deals with protection by cycles in wavelength division multiplexing networks, a subject originated in the operation of telecommunication networks. The most important contribution is the use of design theory to tackle the questions at hand, using tools developed in the paper *Shortest coverings of graphs with cycles* (J. Comb. Theory, Ser. B). This clever perspective can be used to show that other combinatorial problems in communication networks are special cases of traffic grooming, like, for instance, the problem of the minimization of the number of ADMs for all-to-all traffic in unidirectional rings. This allowed the tools and techniques developed in this article to be extended to answer all-to-all traffic grooming in unidirectional rings with larger grooming factors. Most of the techniques used in the literature to prove lower bounds in related problems were established in this article.

Last but not least, Bermond was among the first to identify the importance of designing efficient protocols for structured communication problems, including one-to-all broadcasting, multicasting, and all-to-all broadcasting (a.k.a. gossiping). His many contributions in this field enabled deep understanding of the capabilities and limitations of networks in terms of efficiently propagating informations and data.

By his results and ideas, Jean-Claude Bermond has enriched Distributed Computing considerably, in demonstrating the importance of network structural properties on the performances of distributed algorithms, with applications ranging from fundamental aspects of distributed computing to network design.

The prize has been officially delivered at the the Business meeting of the 17th edition of the Colloquium on Structural Information and Communication Complexity (SIROCCO), Sirince, Turkey, June 7-11, 2010.

Award Committee 2010:

Pierre Fraigniaud	CNRS and University Paris Diderot, France
Shay Kutten	Technion, Israel
Nicola Santoro	Carleton University, Canada
Alexander A. Shvartsman	University of Connecticut, USA
Shmuel Zaks	Technion, Israel

A Review of PODC 2010

Leonid Barenboim
Computer Science Department
Ben-Gurion University of the Negev
leonidba@cs.bgu.ac.il

The ACM Symposium on Principles of Distributed Computing (PODC) deals with algorithmic aspects, design and analysis of distributed systems. Such systems include communication networks, multi-core systems, multi-agent systems, and virtually any system in which computation can be performed in parallel. Since according to current trends these systems are used by more and more electronic devices [1], theoretical and practical research in this area is of highest interest in Computer Science. The most important research results are presented annually in the PODC conference.

Figure 1: The Polybahn and the view from ETH terrace.

This year, for the first time in twenty-nine years since its foundation, the PODC conference was held outside of North America. It took place in the beautiful city of Zurich, Switzerland, on July 25 - 28, 2010. The main building of ETH, the Swiss Federal Institute of Technology, served as a venue for the conference. The institute is among the most prestigious universities in Europe, with more than twenty Nobel laureates that are associated with it, including Albert Einstein. The main building of ETH is located in the center of Zurich, not far from the main train station. A short walk from the station over a bridge that crosses Limmat River leads to the Polybahn - a funicular railway that arrives directly to the terrace behind the ETH building.

[1] It is likely that in a few years even your toaster will be connected to the internet, and be powered by a sixteen-core processor.

From this terrace an amazing view of the city is observed. Inside the building, which preserves the old style but includes all the latest technological innovations [2], one can easily locate the Auditorium Maximum in which the talks took place.

This year the conference attracted a record number of participants. Specifically, among the 176 registrants, more than fifty arrived from North America, and more than a hundred came from Switzerland, Israel, France, and Germany. Clearly, organizing the event in Europe significantly helped with increasing the participation rate of these countries. The highest participation rate of an institution belongs to EPFL Lausanne that was represented by 15 participants. It is followed by the Technion with 8 participants, and Ben-Gurion University with 7 participants. The number of submitted papers this year was also impressive. Among the 179 regular paper submissions, 39 were accepted. In addition, 57 brief announcements were submitted, and 48 of them were selected to appear in the conference.

The conference started with a birthday celebration of Danny Dolev and Eli Gafni. During the celebration Dolev and Gafni received warm greetings from family, friends, students, and colleagues. The latter also emphasized the great contribution of Dolev and Gafni to the field of distributed computing. The next three days were organized according to the following scheme. Each day started with a keynote talk. These talks were given by Hagit Attiya, Pierre Fraigniaud, and Eric Brewer. The presentation of regular papers was split into eleven sessions according to their themes. Most regular paper sessions were followed by brief announcement sessions. The first day evening was devoted to the business meeting. The next day ended with a banquet that last almost until midnight, during which the Dijktra prize was awarded to Tushar D. Chandra, Vassos Hadzilacos, and Sam Toueg. The last day of the conference included a special session called "the best paper session", which was longer than usual, and consisted of seven talks. The first two papers in this session shared the best paper award. I was happy to present the first paper, coauthored with Michael Elkin, "Deterministic Distributed Vertex Coloring in Polylogarithmic Time". The second paper that received the award, is authored by Valerie King and Jared Saia, entitled "Breaking the $O(n^2)$ Bit Barrier: Scalable Byzantine Agreement with an Adaptive Adversary".

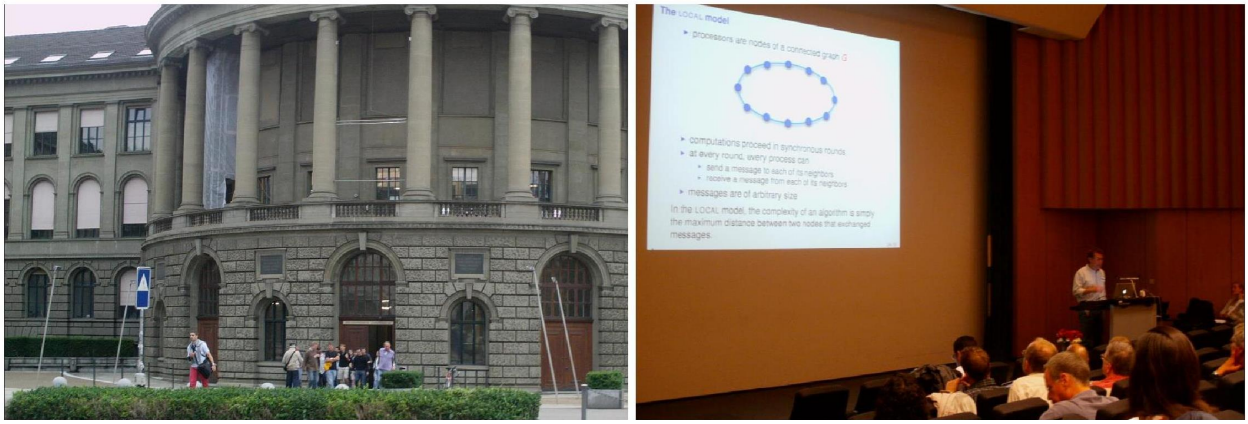


Figure 2: (a) The participants gathering at the entrance to the ETH building. (b) Pierre Fraigniaud describing the "blue" (message-passing) model.

The Program

[2] This is best illustrated by the heavy wooden doors that are automatically opened once you approach them.

Keynote Talks The first talk, entitled "On the inherent complexity of transactional memory and what to do about it", was given by Hagit Attiya. Transactional memory is a concurrency control mechanism that aims to avoid the drawbacks of the alternative mechanisms: lock-based synchronization and wait-free synchronization. The former mechanism suffers from a penalty in running time, while the latter is extremely hard to program. The introduction of transactional memory promised to overcome these problems by taking the responsibility of dealing with concurrency issues from the programmer. However, this approach did not prove itself. As pointed out by Attiya, the programmer must have a certain level of involvement. To this end, a new approach is proposed that is based on mini-transactions. It is currently intensively studied, and already has hardware support by several leading manufacturers.

The second talk, entitled "On distributed computational complexities: are you Volvo-driving or NASCAR-obsessed?", was given by Pierre Fraigniaud. He described the problems that attract the two main communities in distributed computing. The first community, to which he referred as the "red" community, deals with shared memory systems (e.g., multi-core computers). The second, "blue", community, deals with message-passing systems (e.g., communication networks). Fraigniaud pointed out that usually the tools and techniques used by the two communities are completely distinct, which makes it difficult to find common grounds. Nevertheless, he presented a topic that is studied in both communities, namely, the notion of oracles. Hopefully, additional "purple" themes will be found and accepted by both communities. This will increase the interest of a community in the work of the other one, and lead to collaboration.

The last talk, entitled "On a certain freedom: exploring the CAP space", was given by Eric Brewer. The CAP theorem states that any distributed system may satisfy at most two of the following three properties: consistency, availability, and partition tolerance. This theorem was presented in PODC 2000 by Brewer, and formally proved two years later by Nancy Lynch and Seth Gilbert. The theorem directly implies that a system that must satisfy both availability and partition tolerance cannot be consistent. Brewer explained that in this case weaker consistency rules can be applied, such as delayed restore of consistency. Therefore, system designers should choose the strategy of recovery rather than inconsistency prevention.

Paper Presentations Numerous works that were presented in the current PODC improve the understanding of fundamental problems in distributed computing. Despite that the approaches of the two main communities in the field are quite different, as demonstrated by Fraigniaud, it appears that the fundamental problems have things in common. For example, in the synchronization problem of the "blue" community, as well as in the consensus problem of the "red" community, the processors start with distinct values, and the goal is assigning the same (or roughly the same) value to all processors. Another example is the similarity of the problems of coloring and renaming. In both problems the goal is opposite from synchronization and consensus, namely, assigning distinct values to processors. Therefore, it may be beneficial for each community to take a look at the work of the other one once in a while. In what follows, a survey of the latest research on such problems is given, as evident from the conference.

Clock synchronization is one of the most fundamental problems in the "blue" community. In the paper "Optimal Gradient Clock Synchronization in Dynamic Networks", Kuhn, Lenzen, Locher, and Oshman presented an algorithm for synchronization in highly dynamic networks, where communication links may appear and disappear at any time. The goal of gradient clock synchronization is minimizing the skew of nodes that remain close to each other. The presented algorithm has the same bounds as that of the optimal algorithm in the static case, which implies its optimality. In the "red" community the most related problem to synchronization is consensus. Achieving consensus is a universally challenging task, and it is even more challenging in the field of distributed computing. In his paper "A Modular Approach to Shared-memory Consensus, with Applications to the Probabilistic-write Model", Aspnes presents two new classes of shared

memory objects: "ratifiers", for detecting agreement, and "conciliators", for ensuring agreement with some probability. He shows that consensus can be solved by an alternating sequence of these objects.

Another paper that discusses the problem of agreement is "Breaking the $O(n^2)$ Bit Barrier: Scalable Byzantine Agreement with an Adaptive Adversary" by King and Saia. Byzantine Agreement is related to consensus, but it is a much harder problem. The goal is deciding the winner of an election, in which each participant selects from several choices, but some participants are malicious, and provide incorrect data. The authors present the first algorithm with $o(n^2)$ bit communication complexity against an adaptive adversary, which can take over up to a constant fraction of processors at any time. Interestingly, the problem is defined in terms of the "red" community, but the authors' techniques involve graph-theoretic structures, a typical attribute of the "blues".

The coloring problem is another fundamental problem in the field of communication networks. Schneider and Wattenhofer presented a new randomized vertex-coloring algorithm in their paper "A New Technique in Distributed Symmetry Breaking". Their algorithm is extremely efficient for graphs with sufficiently large maximum degree. The authors achieve this result by increasing the number of color selection trials per round. Another paper on coloring is "Deterministic Distributed Vertex Coloring in Polylogarithmic Time", by Barenboim and Elkin. In this paper we devised a deterministic algorithm that employs $\Delta^{1+\epsilon}$ colors (for an arbitrarily small positive constant ϵ), and runs in polylogarithmic time. Despite that randomized logarithmic algorithms have been known for decades, the question of the existence of deterministic polylogarithmic algorithms that employ significantly less than Δ^2 colors remained open. In the current paper we answered this long-standing open question in the affirmative. We also presented efficient algorithms for $O(\Delta)$-coloring, and improved coloring and maximal independent set algorithms for graphs of bounded arboricity.

The mutual exclusion problem, which is intensively studied by the "reds", is somewhat related to the maximal independent set problem. The similarity of these problems is in their constraints. In the shared memory model a pair of processors is not allowed to enter the critical section at the same time, and in the message-passing model a pair of neighboring processors is not allowed to join an independent set. Quite a few papers this year have dealt with topics related to mutual exclusion. One such paper is "The k-Bakery: Local-spin k-Exclusion Using Non-atomic Reads and Writes", by Danek. The author presented the first known shared-memory k-exclusion algorithms that use only atomic reads and writes, have bounded RMR complexity, and tolerate crash failures.

A closely related problem was studied by Bhatt and Huang, and presented in their paper "Group Mutual Exclusion in $O(\log n)$ RMR". The authors devise an algorithm for the cache-coherent model whose RMR complexity matches the known $\Omega(\log n)$ RMR lower bound for the group mutual exclusion problem. Another related problem is the Reader-Writer exclusion. In this problem multiple readers (but not multiple writers) may enter the critical section at the same time. Bhatt and Jayanti presented new algorithms for Reader-Writer exclusion in their paper "Constant RMR Solutions to Reader Writer Synchronization". This is the first known constant RMR complexity algorithm for this problem. Finally, Hendler and Woelfel presented an efficient adaptive algorithm for the classical mutual exclusion, which succeeds even against a strong adversary, in their paper "Adaptive Randomized Mutual Exclusion in Sub-Logarithmic Expected Time".

Randomization is also heavily used by the "blue" community, as evident from the numerous works that employ randomized algorithms and protocols. For example, the paper "Efficient Threshold Detection in a Distributed Environment", by Emek and Korman, deals with a randomized protocol for threshold detection with significantly improved message complexity. The protocol detects an occurrence of at least k events

in a distributed system, for a given threshold k. In particular, it can be used for flooding notification [3]. Randomization is also useful in networks without unique vertex identifiers. An example of such a use can be found in the paper "Partial Information Spreading with Application to Distributed Maximum Coverage", by Censor-Hillel and Shachnai. The authors introduce a new notion of partial information spreading, where only a fraction of the nodes needs to receive each message, but each node needs to receive a fraction of all messages. They also introduce a new tool of weak conductance that generalize classic conductance.

Another area in which randomization is (obviously) essential is random walks. Several papers concern this topic. One such paper is "Expansion and the Cover Time of Parallel Random Walks", by Sauerwald. The cover time lower bound presented by the author matches an upper bound on binary trees up to logarithmic factors. Another paper is "Efficient Distributed Random Walks with Applications", by Das Sarma, Nanongkai, Pandurangan, and Tetali. The authors present two main applications of their algorithms: a fast distributed algorithm for computing random spanning trees, and a fast decentralized algorithm for computing various parameters of the underlying network. A different kind of problem was tackled by Pandit and Pemmaraju in "Rapid Randomized Pruning for Fast Greedy Distributed Algorithms". The authors' technique can be used to speed up greedy approximation algorithms for problems such as minimum dominating set and facility location.

The mentioned papers are just a tip of the iceberg of the dozens of excellent works that appeared at the conference. They covered such "pure red" topics as software transactional memory, asymmetric progress conditions, linearizability, shared memory adversaries, non-blocking search trees, and consensus numbers. The "pure blue" topics that were covered include distance-labeling, multi-party computation, fault-tolerant gossip, dominating sets, load balancing, rendezvous in graphs, target location, online set packing, and mobile, sensor, and radio networks. In addition, several papers concern issues that are not associated with the traditional "red" or "blue" research. These issues include robotics, Bayesian games, and large-scale hosting infrastructures.

Figure 3: The banquet and the view from Uetliberg.

The Banquet The banquet took place in a very special location, in a restaurant on the top of mount Uetliberg, one of the highest points in Zurich area. The mountain rising to 870 meters over sea level, offers a spectacular panoramic view of the entire city and the lake. A unique feature of the place is that

[3] Here flooding means heavy rain, rather than the broadcast algorithm. Flooding can be detected if at least k drops are spotted in a given area. If a system with such a protocol could have been used during the conference, it would have been very helpful, since sudden heavy rains are not a rare phenomenon in Zurich, even in the summer.

it can be accessed only by train. Other types of transportations, except bicycles, are not allowed to get near it. To allow convenient transportation, the organizers of the conference reserved a special train for the participants of the event. During the evening a wide assortment of delicious food and excellent wine was served. The culmination of the banquet was the Dijkstra award ceremony. The Dijkstra prize is awarded for outstanding papers on the principles of distributed computing with evident impact in at least a decade. This year, the prize was given to Tushar D. Chandra, Vassos Hadzilacos, and Sam Toueg, for their seminal work published in two articles in the Journal of ACM in 1996: "Unreliable Failure Detectors for Reliable Distributed Systems", and "The Weakest Failure Detector for Solving Consensus". The work introduces and defines the notion of failure detectors in a distributed system, establishing a theory of failure detectors grounded on a general and precise framework.

Finally, special thanks go to all the people who participated in the organization of such an excellent event: The program committee headed by Rachid Guerraoui, the steering and conference committees, the local arrangements team headed by Roger Wattenhofer that performed an outstanding job, and all the others who helped. We are grateful to all of them!

The next PODC will be held in San Jose as part of the FCRC Mega-event, consisting of more than a dozen conferences, including STOC and SPAA. In 2012 the PODC conference will take place in Madeira- a Portuguese archipelago in the Atlantic Ocean!

Acknowledgments
Photos are courtesy of Jukka Suomela and Leonid Barenboim.

Review of DISC 2010

François Bonnet
School of Information Science
Japan Advanced Institute of Science and Technology
f-bonnet@jaist.ac.jp

Organization

The 24th International Symposium on DIStributed Computing (DISC 2010) took place on September 13-15 in Cambridge, Massachusetts, USA. The conference is an international symposium on the theory, design, analysis, implementation and application of distributed systems and networks. DISC is organized in cooperation with the European Association for Theoretical Computer Science (EATCS).

Since the first symposium (initially called WDAG) in 1985, it was the first time DISC had been organized outside of Europe. This was entirely successful and will certainly happen again in the next few years, maybe in Brazil in 2012.

Back to this year, in addition to the main conference, there were also three co-located workshops the days before and after the conference. Since I have personally attended only the first one, there will be no precise description of their contents:

- 2nd Workshop on Theoretical Aspects of Dynamic Distributed Systems, organized by Roberto Baldoni and Alexander A. Shvartsman,
- 2nd Workshop on the Theory of Transactional memory, organized by Victor Luchangco and Vincent Gramoli,
- 6th International Workshop on Foundations of Mobile Computing, organized by Thomas Moscibroda and Andrea W. Richa.

Scientific Program

Each morning of the three days of conference started with an invited lecture:

1. On Monday, Barbara Liskov, from MIT, gave a lecture on the Power of Abstraction.
2. On Tuesday, Rachid Guerraoui, from EPFL, presented the notion of Speculating in Distributed Computing.

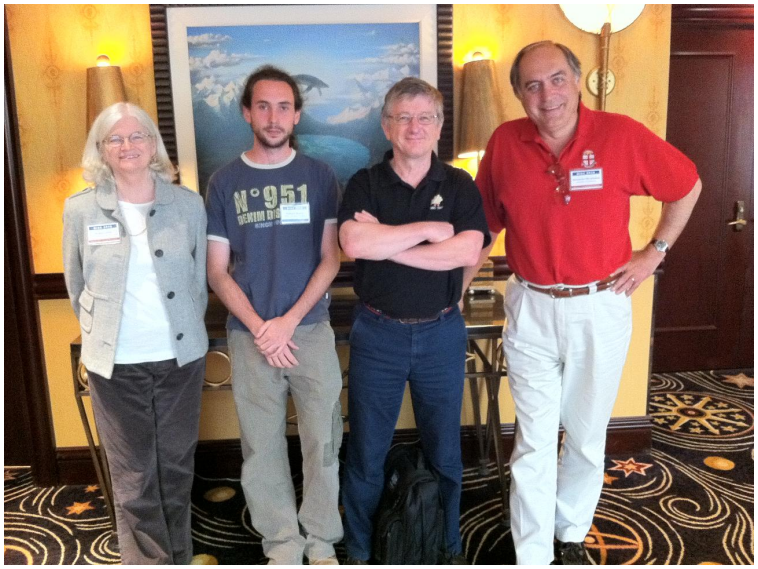


Figure 1: Program co-chairs and authors of the awarded paper.

3. On Wednesday, Nitin Vaidya, from the University of Illinois at Urbana-Champaign, explained how Distributed Algorithms could benefit from Network-Awareness in Wireless Networks.

Regular presentations were split into eight sessions containing 32 papers. As usual in DISC conferences, the program was various, which reflects the good diversity of our community: it went from "classical" distributed systems (shared memories, consensus, ...) to dynamic distributed systems (mobile, wireless networks, ...) without forgetting self-stabilizing or graph algorithms. In addition to the regulars papers, there were also four sessions devoted to brief announcements including 14 presentations.

This year, there was no award for best paper but the Program Committee awarded to François Bonnet the best student award for his paper on Anonymous Asynchronous Systems written in collaboration with his advisor Michel Raynal. Figure 1 shows the two program co-chairs (N. Lynch on the left and A. Shvartsman on the right) around the two authors of the awarded paper.

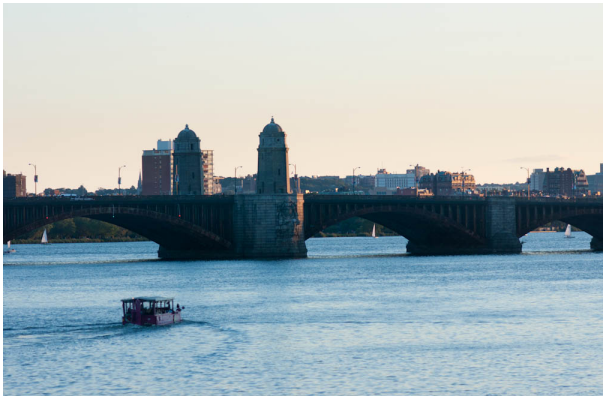
Extra-scientific Informations

We, as participants, were kindly received in the very nice Hotel Marlowe for the conference. From the first reception on Sunday evening to the end of Symposium on Wednesday, the organization was without any hitch. At any time, cold beverages (jugs of iced water with strawberry on each table!) were available, and between sessions, coffee breaks were served including hot drinks and various pieces of food.

On Tuesday afternoon, we had a great social event, the commented visit of Boston in a Duck[1]! Additionally to the discovery of many interesting historical facts on the old city of Boston, we had the opportunity to make the visit on this hybrid vehicle: the Duck is indeed a strange mix of car and boat. It can circulate on land but also floats on water! Figure 2 proposes a comparison of the two kinds of ducks you can find in Boston whereas Figure 3 gives you a brief overview of the city center of Boston.

[1]For anybody interested by this original visit of Boston, see www.bostonducktours.com.

(a) Real ducks.



(b) A strange Duck.

Figure 2: Ducks in Boston.

Figure 3: From the Duck.

Finally the social event ended with a banquet in the Museum of Science[2] for the banquet. That museum looks great, but unfortunately we did not have time to visit it.

Business Meeting

The Business Meeting was held on Monday evening. After traditional statistics on papers and participants, Chryssis Georgiou was re-elected to be part the Steering Committee of DISC. Then there were some strong debates concerning the organization of next years' conferences. For sure, the next event will be organized in Roma, Italy, by Roberto Baldoni from the University of Rome "La Sapienza". The future events may be organized in Brazil (2012) and/or in Israel (2013). However we still need to reach a (distributed) consensus...

Acknowledgments

I would like to thank Antonio Fernández Anta and Sébastien Tixeuil for their photos.

[2]More information available on the website www.mos.org.

Transactional Memory, linking Theory and Practice

Srivatsan Ravi
TU Berlin
Deutsche-Telekom Labs
ravi@net.t-labs.tu-berlin.de

Vincent Gramoli
EPFL
University of Neuchâtel
vincent.gramoli@epfl.ch

Victor Luchangco
Sun Labs, Oracle
victor.luchangco@oracle.com

Abstract

Transactional memory appears promising for democratizing concurrent programming. Its potential lies in producing code that is extensible as atomicity is preserved under composition of transactions. This new abstraction raises several challenges such as the compliance of transactions with alternative synchronization techniques of legacy code and memory consistency models that favor transactional programming. The objective of the Second Workshop on the Theory of Transactional Memory was to discuss new theoretical challenges and recent achievements in the context of transactional computing. We report on some of the issues raised during this workshop.

1 Introduction

The Second Workshop on the Theory of Transactional Memory, organized by Vincent Gramoli and Victor Luchangco, was held on 16 Sep 2010, in conjunction with the DISC 2010 in Cambridge, Massachusetts, USA. It consisted of a panel session on high-level issues about research on the theory of transactional memory (TM), and three technical sessions consisting of talks grouped along common themes. The main goal of the workshop was to foster discussion about the issues raised in these sessions. To achieve this goal, talks were kept brief, and over half the time was spent in vigorous discussion among the speakers and the audience.

The tone of the workshop was set by the panel discussion moderated by Michael Scott, which featured brief talks by Ali-Reza Adl-Tabatabai, Rachid Guerraoui and Mark Moir. The speakers were encouraged to present their points in a provocative manner to stimulate discussion, and a few themes emerged.

Scott emphasized in his introduction that the attraction of TM lay in great part on its perceived simplicity, and that a theory of TM must retain that essential simplicity. While Guerraoui agreed with Scott, he argued that for TM to become popular, it must be acceptable to expert programmers who require more control. Thus, more sophisticated models are needed, which requires that we respect users and trust them to tune the model appropriate to the level of programming.

TM theory should also encompass the interaction of TM with other mechanisms, such as locks and exceptions. Adl-Tabatabai noted that in a recent study with Victor Pankratius [11], students using TM in a project also used other means of synchronization, so a TM theory must specify the behavior of programs with

transactions and other synchronization primitives. Models for TM are also important for designing tools to aid in writing programs that use TM. Currently, there is not even a rough performance model to help guide the choice of programming approaches for a programmer trying to use TM. In addition to a performance model, TM-aware debuggers and profilers are necessary if TM is to be widely adopted. Adl-Tabatabai noted that the lack of such tools was a source of frustration for the students in their study.

One area of controversy among the speakers (and the audience) was the degree to which TM theory should consider practical issues. Guerraoui argued that for TM theory to mature, we must treat it as a "first-class citizen" and grant it the same status as other theoretical topics in the distributed systems community. In contrast, Moir argued that although an abstract model does elide some relevant details of the underlying hardware, it should not impose artificial restrictions on implementations, and that encouraging research on TM models with artificial restrictions saps resources from more important research. One cited example of such a restriction was strict disjoint-access-parallelism, a property that is violated by the most efficient conflict-free algorithms as observed at the former workshop [8]. Unfortunately, conclusions drawn from research on unrealistic models could be, and have been, interpreted wrongly to apply to other settings because the assumptions are not always spelled out clearly, a problem that is exacerbated by the differing uses of terminology between theoreticians, systems researchers and implementors.

The discussion from the invited speakers provided the impetus for subsequent discussion of workshop talks covering topics like the question of composition and concurrency, automated proofs and verification of TM, relaxed memory and message passing models.

2 Composition and concurrency

Composition is an appealing feature of transactions which make them reusable. Although transactional composition guarantees intuitively that the semantics of transactions is preserved despite their concurrency [10], the properties of this semantics are unclear and it is crucial to identify them. For instance, livelock-freedom cannot be preserved under composition. Michael Spear presented a barrier counter-example, derived from [14], involving n threads waiting for each other by first incrementing a counter in a transaction t_1, and then reading the counter value and retrying until the counter value equals n in a subsequent transaction t_2. This program terminates provided that one thread, while executing t_2, sees that the counter value has been changed by another thread incrementing it while running t_1. The cooperation between transaction t_1 of one thread and transaction t_2 of another is impossible if one tries to encapsulate them into a composite transaction t, using a simple closed-nesting technique (a technique discussed by Sathya Peri and Vidyasankar [12]). More specifically, the inherent isolation of t prevents threads from seeing the concurrent counter increments, forcing them to retry t_2 forever.

Composition, as defined by Mihai Letia in his work with Gramoli and Guerraoui [9], is the ability to encapsulate two transactional operations π_1 and π_2 into a transactional operation $\pi_3 = \pi_1 \circ \pi_2$ that preserves their atomicity and deadlock-freedom—but not livelock-freedom. Livelock-freedom cannot even be ensured in presence of contention if transactions are naively scheduled—this is the task of the contention manager to schedule transactions adequately to avoid livelocks. On the topic of contention management, Gokarna Sharma presented his work with Busch on the tight bounds of a randomized algorithm for workloads in which update and read-only transactions are balanced [13]. With any contention manager, relaxed transactions, which enable greater concurrency, tend to violate this composition definition. For example, the elastic model must comprise both elastic and regular transactions to ensure composition [6]. Adam Morrison presented his work on *view transactions* with Afek and Tzafrir [1]. In contrast with elastic transactions, such relaxed transactions use view pointers to indicate which memory locations must remain consistent for

the transaction to commit. Two view transactions that insert and delete can be encapsulated into a move transaction if the programmer extends the scope of the original view pointers to the context of the outermost move transaction. This prevents a programmer from reusing the insert and delete code without modifying it, hence violating composition as defined in [9]. This observation outlines a potential tradeoff between high concurrency and composition of transactions.

3 Verifying TM

Verifying concurrent programs is notoriously difficult because the number of possible interleavings of operations of different threads is exponential in the number of operations. TM can make this easier because a transaction can be treated as a single atomic operation, greatly reducing the number of interleavings that must be considered. However, even verifying the safety of TM algorithms is a complex task due to some (potentially) unbounded parameters like transaction delays and behavior of aborted transactions. Indeed, even specifying what it means for a TM to be correct is not entirely settled: there are several correctness conditions that differ in various ways.

Justin Gottschlich presented his work with Siek and Vachharajani on verifying their InvalSTM [7], using I/O automaton to model the STM and conflict graphs to specify the correctness condition. Their model currently treats commit as an atomic event, and after they complete the proof for that model, they intend to refine it to consider the actual commit procedure.

Victor Luchangco presented work with Doherty and Moir on developing completely formal (i.e., machine-checkable) specifications and verifications of transactional memory, also using I/O automaton, in this case expressed using the PVS language so that the proofs can be checked by the PVS prover [5]. They formalized two specifications for TM, one that captures the guarantees of TM in a very abstract and general way and one that is closer to an implementation, modeling read and write sets explicitly, and proved that the latter implements the former. Next they intend to write formal models of TM algorithms such as *NOrec* [3] and *TL2* [4], and prove that these implement the specifications.

An alternative to having a TM system that guarantees the atomicity of a transaction regardless of the operations done within the transaction is to support "coarse-grained transactions", which capture methods on an abstract data structure that appear to be atomic. Coarse-grained transactions provide potential for improved performance but they increase the programmer's burden because conflicts among these transactions must be correctly specified. Trek Palmer presented work with Eliot Moss introducing the *ACCLAM* language to specify the application-level abstractions of the concurrent program. The resulting specification helps in verifying automatically that low-level conflicts do not trigger false conflicts at the application level and that, upon abort, the involved abstractions are rolled back to a consistent state even though low-level operations are not compensated.

4 Weakening semantics

Several models have been proposed to relax the semantics of transactions to allow more efficient implementations, and also to make explicit the guarantees between transactional and nontransactional operations. Lock-based semantics are inadequate because they do not specify, for example, the behavior of "zombie" transactions that are doomed to abort due to conflicts, but continue to run for some time before the conflict is discovered. These doomed transactions are rescued by TwilightSTM [2], a software TM presented by Annette Bieniusa that aims at relaxing transaction semantics and uses an enriched interface to accept irrevocable twilight regions inside transactions. Luke Dalessandro also suggested to enrich the TM interface in his work with Scott, but for exposing speculation independently from atomicity.

Relaxation issues related to the semantics of nontransactional stores and more generally to the specification of AMD's Advanced Synchronization Facility were discussed by Sean White in a joint work with Spear, and by Stephen Diestelhorst in a joint work with Hohmuth and Polhack. Should the level of consistency be adjustable to resolve contention points and cater to application specific needs? What are the alternatives to "abort on inconsistency"? Can the programmer be oblivious to speculation and rollback? Ideally, the semantics should specified and formalized in a way that encompasses both hardware and software TM implementations, with differing levels of detail depending on the needs of the users and implementors. Hierarchical specifications could be useful for this.

5 Extensions to message passing

Moving from multicore to manycore systems requires reconsideration of our computational models and in particular, communication through message passing rather than shared memory.

Junwhan Kim and Bo Zhang argued in their work with Ravindran that a distributed message-passing system exporting a TM interface may help avoiding inherent synchronization issues of existing RPC-based distributed control-flow programming models. They investigated the support and progress of a data-flow distributed TM where a non-replicated object is moved from node to node.

Alternatively, Eric Koskinen proposed with Herlihy to treat the TM abstraction as a fully-replicated distributed system that allows every thread to optimistically apply single-operation transactions on local copies of transactional objects, and achieves consistency across the threads through an atomic broadcast protocol.

6 Conclusion

To conclude, two main themes emerged repeatedly throughout the workshop. First, it was argued that TM should provide greater flexibility and control to the programmer, and various extensions to do so were proposed. This control could enhance concurrency by letting users give hints to the TM, to exploit best-effort hardware TM, to tolerate irrevocable actions within transactions, or to help differentiate nontransactional stores, all while ensuring composition. Second, greater precision and rigor is necessary to specify the guarantees of TM and the behavior of TM implementations. Several people presented work that advanced the current state, but more work is necessary, particularly to handle the extensions to TM that are contemplated. Although additional control might benefit advanced programmers, many expressed the need to preserve the appealing simplicity of the transactional programming paradigm to facilitate its adoption. Thus, one challenge is to produce a model for TM that satisfies both advanced programmers, who can spend a lot of time and effort to achieve high performance, and novice programmers, who want off-the-shelf solutions that are easy to understand and can be applied to aid in writing correct, efficient and scalable concurrent programs.

Acknowledgements

We wish to thank the contributors to the workshop: Ali-Reza Adl-Tabatabai, Yehuda Afek, Annette Bieniusa, Konstantin Busch, Luke Dalessandro, Stephan Diestelhorst, Simon Doherty, Justin Gottschlich, Rachid Guerraoui, Maurice Herlihy, Michael Hohmuth, Junwhan Kim, Eric Koskinen, Ranjeet Kumar, Mihai Letia, Mark Moir, Adam Morrison, Eliot Moss, Trek Palmer, Sathya Peri, Martin Pohlack, Binoy Ravindran, Michael L. Scott, Gokarna Sharma, Jeremy G. Siek, Michael Spear, Moran Tzafrir, Manish Vachharajani, Sean White and Bo Zhang. We are grateful to Petr Kuznetsov for comments on earlier versions of this report.

References

[1] Y. Afek, A. Morrison, and M. Tzafrir. View transactions: Transactional model with relaxed consistency checks. In *PODC '10: Proceedings of the 29th Annual ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing*, 2010.

[2] A. Bieniusa, A. Middelkoop, and P. Thiemann. Brief announcement: Actions in the twilight - concurrent irrevocable transactions and inconsistency repair. In *PODC '10: Proceedings of the 29th ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing*, pages 71–72, New York, NY, USA, 2010. ACM.

[3] L. Dalessandro, M. F. Spear, and M. L. Scott. NOrec: streamlining stm by abolishing ownership records. In *PPoPP '10: Proceedings of the 15th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, pages 67–78, 2010.

[4] D. Dice, O. Shalev, and N. Shavit. Transactional locking II. In *DISC '06: Proceedings of the 20th International Symposium on Distributed Computing*, pages 194–208, 2006.

[5] S. Doherty, L. Groves, V. Luchangco, and M. Moir. Towards formally specifying and verifying transactional memory. *Electronic Notes in Theoretical Computer Science*, 259:245 – 261, 2009. Proceedings of the 14th BCS-FACS Refinement Workshop (REFINE 2009).

[6] P. Felber, V. Gramoli, and R. Guerraoui. Elastic transactions. In *DISC '09: Proceedings of the 23rd International Symposum on Distributed Computing*, volume 5805 of *LNCS*, pages 93–107, sep 2009.

[7] J. E. Gottschlich, M. Vachharajani, and J. G. Siek. An efficient software transactional memory using commit-time invalidation. In *CGO '10: Proceedings of the 8th Annual IEEE/ACM International Symposium on Code Generation and Optimization*, pages 101–110, New York, NY, USA, 2010. ACM.

[8] V. Gramoli. What theory for transactional memory? *SIGACT News*, 40(4):79–81, 2009.

[9] V. Gramoli, R. Guerraoui, and M. Letia. The many faces of transactional software composition. Technical Report EPFL-REPORT-150654, EPFL, 2010.

[10] T. Harris, S. Marlow, S. Peyton-Jones, and M. Herlihy. Composable memory transactions. In *PPoPP '05: Proceedings of the 10th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, pages 48–60, New York, NY, USA, 2005. ACM.

[11] V. Pankratius, A.-R. Adl-Tabatabai, and F. Otto. Does transactional memory keep its promises? Results from an Empirical Study. Technical Report 2009-12 IPD, University of Karlsruhe, Germany, September 2009.

[12] S. Peri and K. Vidyasankar. Correctness of concurrent executions of closed nested transactions in transactional memory systems. In *ICDCN'11: Proceedings of the 12th International Conference on Distributed Computing and Networking*, 2011. to appear.

[13] G. Sharma and C. Busch. A competitive analysis for balanced transactional memory workloads. In *Proceedings of the 14th International Conference On Principles Of Distributed Systems*, 2010.

[14] Y. Smaragdakis, A. Kay, R. Behrends, and M. Young. Transactions with isolation and cooperation. In *OOPSLA '07: Proceedings of the 22nd Annual ACM SIGPLAN Conference on Object-Oriented Programming Systems and Applications*, pages 191–210, New York, NY, USA, 2007. ACM.