Probabilistic versus Deterministic Techniques for Shared Memory Computation

Hagit Attiya (Technion)
Maurice Herlihy (Brown University)
Philipp Woelfel (University of Calgary)

February 5–10, 2012

1 Overview of the Field

Shared memory algorithms exploit the parallelism of multiple computing units. As the number of components and the complexity of their interconnection increases, the difficulty of dealing with the resulting asynchrony and faulty behaviour of this elaborate hardware also increases. In these distributed settings proofs of correctness of proposed solutions and analysis of the complexity of the solutions are challenging. Furthermore, some fundamental algorithmic problems cannot be solved efficiently or at all. Determinism is a significant barrier that limits the power of shared memory computing. For example, a symmetric configuration of several processes may lead to deadlock. Progress will be stalled unless the symmetry can be broken, which is impossible in some deterministic settings. Sometimes, a shared memory computation can be very expensive for a small set of specific initial configurations or under an unfortunate scheduling of the various processes. In deterministic settings, there is no way to insure that these difficult cases do not arise or at least are rare. Probabilistic methods, however, have proved useful to overcome these impasses. The likelihood of problematic scenarios can be made negligible with applications of randomization. Allowing processes to use private random bits helps to break symmetry, to diminish the impact of a malicious schedule or a difficult initial configuration. There is a significant and growing collection of applications of randomization in various multi-process settings. The Ethernet exponential back-off algorithm was an early example that continues to have widespread application. Valiant's randomized routing algorithm [45] is an elegant example of a simple and effective use of randomization. The famous impossibility result [27], stating that it is not always possible to achieve consensus in a system with only one faulty process, has been overcome with increasingly more efficient randomized algorithms. As a consequence of randomized universal constructions, any shared object has a randomized implementation in an asynchronous system with faulty processes (wait free), while (provably) no such deterministic implementations exist.

Many more examples of randomized shared memory algorithms, which help overcome the difficulties associated with asynchrony and faultiness, are known. The results, however, are mostly ad hoc. A comprehensive theory of probabilistic methods for shared memory computation is missing. In general, the power of probabilistic methods and the randomized complexity of many fundamental problems is not well understood. In contrast, over the past 20 years, rich and comprehensive complexity-theoretic results for deterministic shared memory systems have emerged. One example is the infinite wait-free hierarchy that classifies the power of hardware primitives when used by deterministic algorithms. Under randomization, however, the wait-free hierarchy collapses. Similar hierarchical results are not known for systems that can make use of

private random bits. Part of the reason is a lack of general techniques to prove lower bounds for randomized algorithms. In the deterministic setting, a wide arsenal of such techniques is known. Examples include algorithmic tools such as covering, valency, scenario, or chain arguments, and mathematical tools such as information theory, Ramsey theory, and algebraic topology. In sequential computing, methods such as Yao's Min-Max principle [46] can be used to derive lower bounds for randomized algorithms from lower bounds for deterministic algorithms. In distributed computing, the situation is more complicated, and generally the usefulness of Yao's principle seems to be limited or at least is not well understood. Another example is a lack of understanding of correctness conditions for randomized algorithm design: The standard correctness condition used in deterministic settings, *linearizability* [36], is not suitable for randomized algorithms.

2 Recent Developments and Open Problems

In recent years, several new randomized algorithms and lower bounds for fundamental shared memory problems have emerged, and attempts were made to develop a theory for probabilistic methods.

2.1 Theory of Probabilistic Shared Memory Algorithms

In the standard shared memory model, each process takes steps (i.e., executes shared memory operations) as determined by its program. The order of steps by concurrent processes can be interleaved arbitrarily. The standard worst-case analysis of deterministic shared memory algorithms assumes that this interleaving occurs in the "worst" possible way. Under such assumptions, many problems don't have any or only very inefficient solutions. Probabilistic algorithms allow processes to make random decisions, and prevent (with high probability) worst-case schedules from occurring. In the analysis of randomized algorithms, adversary models describe how the interleaving of process steps can be influenced by random choices made by processes. The most optimistic assumption, where there is no such influence, is modeled by the *oblivious adversary*, and the most pessimistic reasonable assumption is modeled by the *strong adaptive adversary*. Several adversary models with intermediate strengths were defined (see [9] for an overview of some of them).

A comprehensive study of the relative strength of adversary models is still missing. Most results are ad-hoc, but some problems (most prominently the consensus problem, see Section 2.2) have been studied extensively in a wide variety of adversary models. For many problems, e.g., consensus, leader election, or mutual exclusion, no very efficient randomized algorithms are known (or are known not to exist) under the most pessimistic system assumptions (strong adaptive adversary). Therefore, more recently the oblivious adversary has become the focus of research interest. For many problems it has been shown that the oblivious (or slightly stronger) adversary model allows significantly more efficient randomized algorithms than the strong adaptive adversary. Examples are consensus [11], leader election [29], or mutual exclusion [23] (see also Section 2.2).

The standard correctness condition for deterministic shared memory algorithms is *linearizability* [36]. A linearizable implementation can replace an atomic object (which allows every operation to be executed instantaneously) in any deterministic algorithm without affecting the algorithm's worst-case behaviour. A significant amount of research effort has been spent on finding linearizable implementations of fundamental primitives, or to prove the non-existence of such implementations. It was recently observed [32] that replacing atomic objects with linearizable ones in randomized algorithms may not preserve probability distributions of executions; in other words, such a replacement may increase the strength of the underlying adversary model. While a suitable correctness condition, *strong linearizability*, was proposed and investigated for the strong adaptive adversary model [32, 33], not much is known for weaker adversary models.

Yao's Theorem [46] provides a general technique to prove lower bounds for randomized *sequential* algorithms. For example, it follows from that theorem that an expected running time lower bound for a random input (for any given probability distribution) and the fastest possible deterministic algorithm yields the same expected running time lower bound for the fastest possible randomized algorithm and the worst input. In concurrent algorithms, adversarially generated schedules play the same role as inputs in sequential algorithms. Since in non-oblivious adversary models those schedules are determined dynamically, based on previous random decisions made by processes, it is unclear how Yao's' Theorem can be applied. Recently [30], it was shown how one can apply Yao's principle in the strong adaptive adversary model, but this result is ad-hoc

for one particular problem (mutual exclusion). A more comprehensive understanding of the applicability of Yao's principle for randomized concurrent algorithms is still missing.

2.2 Randomized Algorithms for Fundamental Problems

The *consensus* problem is among the best studied problems in distributed computing. Processes *propose* different values, and a consensus protocol lets all of them agree on one of the proposed values. The consensus problem is fundamental for distributed computing—consensus objects are very powerful synchronization primitives, and the difficulty of solving consensus helps evaluating the strength of a computational model. For example, the famous Fischer-Lynch-Paterson (FLP) proof [27] applied to the shared memory model [41] states that no deterministic wait-free algorithm can solve consensus using only atomic read/write registers. It has also been known for a long time, that one can solve consensus with randomized wait-free protocols (e.g., [22]). But only in recent years, the complexity of randomized wait-free protocols for different system assumptions (e.g., various adversary models) has been determined more or less accurately [10–13, 19, 20]. One of the intriguing open question remains how fast one can solve consensus in the oblivious adversary model, i.e., under very optimistic system assumptions.

Consensus can be viewed as a form of *strong leader election*, in which all processes agree on the ID of one of the participating processes, who becomes the leader. *Weak leader election* is a simpler problem, where processes also elect one unique leader, but non-leaders don't need to learn the ID of the leader. This problem can also not be solved deterministically in systems that provide only atomic registers. An efficient randomized solution with logarithmic expected time complexity in the strong adaptive adversary model for this problem has been known since 1992 [1], but recent research efforts have several yielded refined solutions. Examples comprise an efficient *adaptive* algorithm [7], where the time complexity depends only on the number of participating processes as opposed to all processes in the system, and algorithms with almost constant time respectively small space complexity [2, 28, 29] in oblivious or weak adaptive adversary models.

Another classical problem is that of *renaming* [18], where each participating process tries to acquire a distinct name from some given small name space. This problem has deterministic solutions for a large enough name space even if the system provides only atomic registers. However, in such a system, renaming is not possible if the name space is only slightly larger than the number of participating processes (see for example [24, 37]). Moreover, most deterministic renaming algorithms are slow, even if strong synchronization primitives are available and the name space is not tight. (Only very recently a deterministic algorithm with step complexity logarithmic in the number of participating processes was obtained [26]; that algorithm uses registers and compare-and-swap objects.) Recently, several new very efficient randomized algorithms renaming algorithms were devised for different system assumptions [4, 5, 7], the fastest having only double logarithmic expected step complexity even if only atomic registers are available. A recent tight lower bound for randomized renaming algorithms [6] shows that for a polynomial name space only randomized Monte Carlo algorithms can have sub-logarithmic expected step complexity in the strong adaptive adversary model.

The most powerful synchronization primitive is a *mutual exclusion* object [25], which allows processes to serialize accesses to protected resources. Mutual exclusion objects are omnipresent in modern operating systems and concurrent algorithms, and, not surprisingly, finding efficient mutual exclusion algorithms has been of ongoing interest. A recent lower bound [21] showed that using a standard set of synchronization primitives (e.g., registers and compare-and-swap objects), deterministic algorithms cannot achieve a sub-logarithmic complexity with respect to the standard complexity remote-memory-references (RMR) measure. More precisely, the lower bound states that any deterministic algorithm incurs at least $\Omega(\log n)$ RMRs per passage through the critical section, in the worst-case. Since then researchers have tried to answer the question whether randomization can help to circumvent this barrier. While a small improvement is possible under very pessimistic system assumptions (the strong adaptive adversary model) [34, 35, 43], a recent lower bound [30] shows that more than a $\log \log n$ factor cannot be gained. However, it turned out that under optimistic system assumptions (oblivious adversary model), significantly more efficient algorithms are possible [23, 31].

In recent years, research on probabilistic methods for shared memory computing has gained significant momentum and led to several novel solutions and lower bounds for other fundamental problems. Examples are snapshot and max-registers implementations [14, 17] or task allocation protocols [3, 8].

3 Presentation Highlights and Open Questions

(a) Michael Bender and Seth Gilbert, *Mutual Exclusion in $O(\log^2 \log n)$ Time*

Michael and Seth gave a joint talk on the first randomized mutual exclusion algorithm which is exponentially faster than deterministic algorithms [23] in the oblivious adversary model. In particular, their algorithm guarantees that each passage through the critical section costs amortized $O(\log^2 \log n)$ remote memory references (RMRs) with high probability, where n is the number of processes in the system. It guarantees that every process enters the critical section (i.e., deadlock-freedom) with high probability. The algorithm achieves its efficient performance by exploiting a connection between mutual exclusion and approximate counting.

Several open problems were identified, e.g., whether similarly efficient but deterministically deadlock-free algorithms may exist, and whether the true randomized RMR complexity of mutual exclusion is constant in the oblivious adversary model.

(b) Rotem Oshman, *Mutual Exclusion with Faulty Memory*

Modern implementations of computer memory are extremely compact and power-efficient, but this comes at the cost of increased susceptibility to "soft errors", where a bit changes its value spuriously. Rotem presented her work with co-author Moscibroda [42], in which they study the effects of soft memory errors on 2-process mutual exclusion algorithms. Specifically, Rotem asks what degree of fault-tolerance can be achieved using no extra memory compared to classical algorithms such as Peterson and Dekker's algorithms; i.e., three binary registers. Rotem and her co-author suggest a probabilistic model to measure the susceptibility of a (deterministic) mutual exclusion algorithm to memory errors, and show that under this model it is possible to guarantee mutual exclusion with probability 1 when a single register is faulty and may exhibit unboundedly many faults. On the other hand, they proved that *worst-case* (i.e., deterministic) fault tolerance cannot be achieved in this setting. Rotem gave several other related results characterizing the degree of fault tolerance that can be achieved using binary registers.

An open question is what happens if registers are not binary.

(c) Dan Alistarh, *The Complexity of Renaming*

Dan presented results from his recent FOCS paper with Aspnes, Gilbert, and Guerraoui [6], studying the complexity of renaming. The paper provides an individual lower bound of $\Omega(k)$ process steps for deterministic renaming into any namespace of size sub-exponential in k, where k is the number of participants. This bound is tight: it draws an exponential separation between deterministic and randomized solutions, and implies new tight bounds for deterministic fetch-and-increment registers, queues and stacks. The proof of the bound relies on the a reduction from renaming to mutual exclusion. Dan also discussed a global lower bound of $\Omega(k \log(k/c))$ on the total step complexity of renaming into a namespace of size ck, for any $c \geq 1$. This applies to randomized algorithms against a strong adversary, and helps derive new global lower bounds for randomized approximate counter and fetch-and-increment implementations, all tight within logarithmic factors.

The talk and the following discussion raised several open questions: For example, what is the (randomized) complexity of long-lived renaming, where processes can release names that they obtained earlier? Is it possible to exploit initial names assigned to processes? Finally, are there algorithms that circumvent the lower bound by allowing errors or failures with small probability?

(d) George Giakkoupis, *On the Time and Space Complexity of Randomized Test-And-Set*

George described new results on the time and space complexity of randomized Test-And-Set (TAS) implementations from atomic read/write registers in asynchronous shared memory models with n processes [29]. He presented an adaptive TAS algorithm with an expected (individual) step complexity of $O(\log^* k)$, for contention k, against the oblivious adversary, improving a previous (non-adaptive) upper bound of $O(\log \log n)$ [2]. He also presented a modified version of the adaptive RatRace TAS algorithm [7], which improves the space complexity from $O(n^3)$ to $O(n)$, while maintaining logarithmic expected step complexity against the adaptive adversary.

An open problem is whether whether the algorithmic technique that yields the $O(\log^* k)$ upper bound can be applied to other problems, for example to consensus. Another open problem is whether the linear space upper bound is optimal.

(e) Keren Censor-Hillel, *Polylogarithmic Implementations of Restricted-Use Objects*

Implementing objects that only need to support restricted use, such as a bounded number of operations or a bounded number of values, can be strictly faster than general constructions. Keren described polylogarithmic constructions of restricted-use concurrent data structures. This is in contrast to existing linear lower bounds for the unrestricted versions of such objects by Jayanti, Tan, and Toueg [38]. Keren first described her logarithmic max register implementation obtained in collaboration with James Aspnes and Hagit Attiya [14]. Then she gave a restricted-use snapshot object that requires only $O(\log^3 n)$ steps for updates and $O(\log n)$ steps for scans, for systems with n processes. This result is based on work by the same authors together with Faith Ellen [15]. The upper bound seems counterintuitive, since it does not depend on the number of components in the snapshot. The key to this implementation is the construction of a new object consisting of a pair of max registers that supports a scan operation.

An open problem is to close the polylogarithmic gap that usually exists between upper and lower bounds for most objects. Moreover, only very few randomized lower and upper bounds are known (and they are very particular and not for general objects).

(f) Danny Hendler, *Lower Bounds on Restricted-Use Objects*

Danny's joint work with James Aspnes, Hagit Attiya and Keren Censor-Hillel [16] draws a more complete picture by defining a large class of objects for which an operation applied to the object can be "perturbed" L consecutive times, and proving lower bounds on the time and space complexity of deterministic implementations of such objects. This class includes bounded-value max registers, limited-use approximate and exact counters, and limited-use collect and compare-and-swap objects; L depends on the number of times the object can be accessed or the maximum value it can support.

For implementations that use only historyless primitives, Danny and his co-authors proved lower bounds of $\Omega(\min(\log L, n))$ on the worst-case step complexity of an operation, where n is the number of processes; they also prove lower bounds of $\Omega(\min(L, n))$ on the space complexity of these objects. When arbitrary primitives can be used, they proved that either some operation incurs $\Omega(\min(\log L, n))$ memory stalls or some operation performs $\Omega(\min(\log L, n))$ steps.

In addition to these deterministic lower bounds, the work establishes a lower bound on the expected step complexity of restricted-use randomized approximate counting in a weak oblivious adversary model.

An open question remains whether there are expected time lower bounds for randomized implementations of general L-perturpable objects.

(g) Valerie King.

Valerie presented two works in progress. The first was an examination of an often-cited but never published lower bound for resilience for a randomized Byzantine agreement problem. Valerie showed what she thought were the short-comings of method sketched in an unpublished manuscript from the 1980's by Karlin and Yao. Then she proposed a possible fix, in the case of private channels. She also asked the question of whether limiting messages passed by each processor including faulty processors would allow for an algorithm with higher resilience.

The second was a randomized message passing algorithm for asynchronous consensus which uses fewer messages than that given by emulating the shared memory solution.

(h) Eric Ruppert, *The Complexity of Non-Blocking Dictionaries*

The dictionary data structure is fundamental to computer science. It stores a set of elements and provides update operations that add and remove elements, and query operations that can find specific elements, find successors or predecessors, and find minimum and maximum elements. A dictionary implementation that can be accessed concurrently by many processes is called non-blocking if some process must eventually complete its operation. It is known that there is a non-blocking dictionary implementation that uses $O(n + c)$ amortized time per operation, where n is the number of elements in the dictionary and c is

the maximum number of processes that ever run concurrently. This is a fairly poor upper bound on the complexity of the problem. Eric discussed possible approaches for improving it.

Several open problems were identified:

- Prove a good upper bound on the amortized step complexity of operations on a non-blocking search tree.
- Prove a good upper bound on the expected depth of a search tree built by concurrent insertions.
- Prove a good upper bound on the expected amortized step complexity of a skip list.
- Extend existing non-blocking data structures to handle predecessor queries, etc.
- Use other data structures as the basis of non-blocking dictionary implementation (e.g. treaps or van Emde Boas trees).
- Create an efficient non-blocking implementation of dynamic perfect hashing.

(i) Jennifer Welch, *Shared Memory Computation Under Probabilistic Churn Models*

Distributed systems with churn, or dynamic distributed systems, allow the processes to join and leave the system at will. Jennifer described a modification of an existing algorithm to implement a shared read-write register in such a system, in which replicas of the state of the implemented object are distributed among the processes [40]. When a process joins the system, it attempts to obtain an up-to-date copy of the data from other processes. Copies of the register are updated by broadcasting information. The consistency condition provided by this algorithm for read operations that are able to return a value is shown to be in the intersection of sequential consistency and a multi-writer variant of regularity.

To model the dynamicity of the system with churn, she and her collaborators used a continuous-time birth-death process which is a special case of continuous-time Markov processes. Given the joining and leaving rates of the processes, she analyzed the likelihood there are no active processes, the mean duration of a time interval containing active processes, and the probability that an era ends while a process is attempting to join.

It was mentioned that some of the results might get improved using Wald's Identity. An open problem is whether the results extend to more realistic assumptions.

4 Scientific Progress Made and Outcome of the Meeting

We first describe several selected research results that were seeded at the workshop. Many of those resulted in new publications at the top international conferences on theoretical computer science (STOC and FOCS) respectively parallel and distributed computing (PODC, SPAA, and DISC); since we cannot list all of them, we will only discuss some selected ones. After that, we present a number of comments that were made by workshop participants, and which draw a picture of the impact the workshop had on their work and collaborations resulting from the workshop. It should be noted that workshop sparticipant Jared Saia also wrote a blog post [44], summarizing his impressions on the workshop.

4.1 Selected Research Results Seeded at the Workshop

- Michael Bender's and Seth Gilbert's talk on mutual exclusion triggered a discussion about whether randomized deadlock-freedom with a small error probability may be desirable. This led to the question whether a similarly efficient algorithm exists that achieves deterministically deadlock-free. Similarly, the algorithm demonstrated that algorithms can be significantly faster in the oblivious adversary model than in the strong adaptive adversary model, and raised the question whether mutual exclusion with constant RMR complexity is possible. The resulting discussions sparked interests of BIRS participants Giakkoupis and Woelfel and ultimately resulted in a new mutual exclusion algorithm that achieves deterministic deadlock-freedom and constant RMR complexity [31], thus answering the open questions from Bender's and Gilbert's talk.

- Discussions on the renaming problem (see also Dan Alistarh's talk) have led to new collaboration among workshop participants Dan Alistarh, James Aspnes, George Giakkoupis, and Philipp Woelfel. Research initiated at the workshop and published at PODC 2013 [5] gives renaming algorithms that circumvent the lower bounds presented by Dan Alistarh in his talk, by allowing a small error probability.

- There are close relations between the asynchronous shared memory model and message passing models. Algorithmic techniques in one model can be applicable in the other one. Ideas from work on randomized shared memory algorithms discussed at the workshop laid the foundations for a randomized, polynomial expected time algorithm to solve asynchronous Byzantine Agreement in a message passing model with a strong adversary that can control up to a constant fraction of the processors. Workshop participants Valerie King and Jared Saia published those results at STOC 2013 [39]. (See also Valerie's comment in Section 4.2).

- James Aspnes and Keren Censor-Hillel built on the work presented in Keren's talk, and used a randomized techniques to obtain *long-lived* snapshot objects and max registers with poly-logarithmic expected step complexity, whereas previous work allowed such efficiency only by limiting the number of operations on the objects [17].

- Following initial ideas discussed during the workshop, participants George Giakkoupis, Lisa Higham, and Philipp Woelfel, as well as Lisa's and Philipp's PhD student Maryam Helmi worked on understanding the relation between deterministic and randomized progress conditions. They proved that in general any deterministic obstruction-free algorithm, where each process finishes if it executes b uninterrupted steps, can be turned into a randomized wait-free one, with a step complexity that is polynomial in b and the number of processes in the system [28].

4.2 Statements from Workshop Participants

Valerie King, Feb 5, 2013: *Dear Nassif and the organizers of BIRS workshop 12w5122,*

The attached paper [39] was just accepted to STOC. It solves a longstanding problem in theoretical computer science. I just wanted to thank you and the organizers of the workshop. It was there, listening to the talks of the workshop on shared memory, in the beautiful surroundings of Banff, that we first had the idea of applying ideas from shared memory to this problem which is not about shared memory. So we have BIRS to thank. [. . .]

Lisa Higham, Feb 21, 2012 (In an e-mail to Nassif Ghoussoub and the workshop organizers):

[. . .] *In short, best workshop I have ever attended.*

First off there were talks on the latest by leaders in the research area. Without exception, they were excellent. I learned many new techniques and some surprising results. Also, a few of those beautiful insights that seem so simple and natural —- once someone shows you!!

In my own talk, I presented work from our dept (postdoc Golab, Prof Woelfel and myself). We have shown that the gold standard for correctness of implemented objects (Linearizability), which the community has relied on for 20 years, does not work when the algorithms using these implementations are randomized. Instead we need a stronger property. I followed with our recent work on possibilities and impossibilities for this stronger property. The work caused a stir. I received extremely helpful and encouraging feedback from many participants. The questions and comments have given us new research ideas and directions to last several years, as well as some immediate improvements. For concreteness, here are some directions I intend to pursue:

Rotem Oshram and Maurice Herlihy suggested that our work could be closely connected to (one of the many) notions of refinement mappings used by verification researchers. We need to nail down this connection.

I learned that the Principles of Programming Languages (POPL) community has developed some automated techniques to verify some instances of Linearizability, but have not been able to figure out when these techniques work and when they are inadequate. One of the BIRS participants, Hagit Attiya, pointed this out and suggested that their impasse may be clarified by our work. Hagit's suggestion is to team up with someone

in the POPL community to both communicate our current work and extend it. Furthermore, Hagit is helping to implement this suggested collaboration.

One conjecture that has eluded us was picked up by at least two other researchers at BIRS, making collaboration on this question a good option.

I would not have learned these connections or made these contacts without the workshop. It was exhilarating (and exhausting) in the best of ways. I am sure I am not alone in feeling an extra charge to my research as a consequence of the meeting. Because of Calgary's strong involvement, I believe the workshop also helped make the UofCalgary more prominent in this community.

Eric Ruppert, Feb 13, 2012: *Here are a few thoughts on the impact the workshop had on me.*

- *I found out a lot about what people in the area are working on currently*
- *After I gave the talk on open problems, a number of people approached me with ideas about how some of the problems could be tackled, and this could lead to future collaborations. Also, I found that one attendee was already working on related problems. Another attendee expressed interest in working on the problems too, and I hope that will lead to a collaboration.*
- *Others shared their open problems, giving me some ideas about what to work on in the future.*

Hagit Attiya, Feb 11, 2012: *Being at BIRS was a wonderful experience: The wonderful surroundings, the convenient facilities, and the warm hospitality of the staff has made the stay very comfortable. (Similar thanks go to the Banff centre staff, in the recreation facilities and the dining room.) I particularly, enjoyed the suite that I received, which was splendid!*

For me it was first and foremost an opportunity to catch up on interesting research in my area. But it was also an opportunity to create new research: With 11 of my collaborators attending the workshop, I could make progress on several papers that I am writing (in one case, a presentation of the work by one of my coauthors has helped to find new applications). I also had a chance to describe some fresh results to relevant researcher and get their feedback. Finally, I got started on one or two new research directions.

Jennifer Welch, Feb 11, 2012: *Thanks again for all your hard work for a great workshop. It was very informative and enjoyable.*

Workshop impact:

- *Opportunity to learn about the latest results in the area of shared memory distributed computing*
- *Sparked ideas for new directions for collaborative research*
- *Provided ideas for new material to cover in my distributed computing course, as well as ideas for challenging new homework and exam problems*

Jared Saia, Feb 10, 2012: *This was one of the strongest workshops that I have been to in the past 10 years. Although I initially had little background in shared memory, I had many excellent outcomes from the workshop, three of which I describe below.*

First, I had very useful feedback on my talk. My talk was on emulating a shared object in a wireless network where an adversary can jam communication between processors. I had several suggestions about ways to extend my result including determining: 1) what happens when there are multiple communication channels available; 2) what happens if one can use signal processing in the event of a jam to determine what the underlying message was in the case where many processors broadcast the sam underlying message; 3) if one can reduce the power consumption necessary to in order to maintain the state of the shared object, perhaps expending more energy only when there is a need to change state; and 4) how to design algorithms that reduce the energy consumption per processor needed to emulate the shared object, as the number of processors grows.

The second major outcome of the workshop was the beginning of collaborations on two separate papers. The first paper is related to extending the research described in my talk along direction 4) described

above. This collaboration began through discussions with Seth Gilbert, a colleague with whom I have not collaborated previously. The second paper is related to reducing communication costs for consensus in the asynchronous communication model. The main idea of this result, borrows a trick frequently used for achieving consensus with shared registers, and adapts it to the message passing model. This type of cross-pollination of ideas between shared memory and message passing models of communication is important but rare. Certainly, it would not be occurring without the impetus of the BIRS workshop. This collaboration is with workshop participants Seth Gilbert, Jim Aspnes and Valerie King.

The third major outcome was a quick but thorough education about the frontiers of shared memory research. To prepare for the workshop, I read through papers from several of the participants. However, when I arrived, I realized that I would learn about the frontier of research in shared memory, and that this was not something that one could gain from simply reading published papers. Through the workshop, I learned about major open problems, mathematical and algorithmic techniques used to achieve results, and general vision for the field. No collection of papers or even informal group meetings with colleagues at conferences could have achieved anywhere close to the same outcome.

The coordinators for the workshop were excellent, particular Phillip Woefel who went out of his way to make sure everyone had a fun and productive time. In addition to my own positive experiences, I also witnessed many small meetings among researchers discussing problems posed at the workshop after the talks. I am sure that several of these initial collaborations will blossom into interesting results. Finally, the Banff center was extremely well run with excellent hospitality, facilities, logistics, food, and a beautiful natural setting.

Valerie King, Feb 10, 2012: *The workshop gave me great feedback on these problems. A collaboration was started with Jim Aspnes who joined Seth and Jared with me to work on problem 2* [randomized message passing algorithm for asynchronous renaming]). *On problem 1* [randomized Byzantine agreement]*, I had the opportunity to see if this audience of experts could see any interpretation of the manuscript which would yield a lower bound. I was encouraged to pursue a publication with a formal proof and given the name of a student who had worked on the problem over 15 years ago, who might have some insight about what was meant (given that the authors cannot tell me themselves–I did try already to contact them about it.)*

Also, Jared and I were inspired to begin work at Banff in a new direction for us working for the first time in the shared me.

Keren Censor-Hillel, Feb 9, 2012: *Workshop impact:*

- *Getting up to date with current research interests in distributed computing in shared memory.*
- *Learning about new techniques for algorithms or for lower bound proofs.*
- *New collaborations.*

Thanks for everything—the workshop was great!

Danny Hendler, Feb 8, 2012: *Apart from being fun, this workshop contributed to me professionally. Following workshop discussions and talks, I have identified with colleagues two promising research directions which we intend to pursue. Moreover, I believe that the new algorithmic ideas and proof techniques I was exposed to in the course of this excellent workshop will benefit my research in the theory of shared-memory systems.*

References

[1] Yehuda Afek, Eli Gafni, John Tromp, and Paul M. B. Vitányi. Wait-free test-and-set. In *Proceedings of the 6th International Workshop on Distributed Algorithms (WDAG)*, pages 85–94, 1992.

[2] Dan Alistarh and James Aspnes. Sub-logarithmic test-and-set against a weak adversary. In *Proceedings of the 25th International Symposium on Distributed Computing (DISC)*, pages 97–109, 2011.

[3] Dan Alistarh, James Aspnes, Michael A. Bender, Rati Gelashvili, and Seth Gilbert. Dynamic task allocation in asynchronous shared memory. In *Proceedings of the 25th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 416–435, 2014.

[4] Dan Alistarh, James Aspnes, Keren Censor-Hillel, Seth Gilbert, and Morteza Zadimoghaddam. Optimal-time adaptive strong renaming, with applications to counting. In *Proceedings of the 30th SIGACT-SIGOPS Symposium on Principles of Distributed Computing (PODC)*, pages 239–248, 2011.

[5] Dan Alistarh, James Aspnes, George Giakkoupis, and Philipp Woelfel. Randomized loose renaming in $O(\log \log n)$ time. In *Proceedings of the 32nd SIGACT-SIGOPS Symposium on Principles of Distributed Computing (PODC)*, pages 200–209, 2013.

[6] Dan Alistarh, James Aspnes, Seth Gilbert, and Rachid Guerraoui. The complexity of renaming. In *Proceedings of the 52nd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 718–727, 2011.

[7] Dan Alistarh, Hagit Attiya, Seth Gilbert, Andrei Giurgiu, and Rachid Guerraoui. Fast randomized test-and-set and renaming. In *Proceedings of the 24th International Symposium on Distributed Computing (DISC)*, pages 94–108, 2010.

[8] Dan Alistarh, Michael A. Bender, Seth Gilbert, and Rachid Guerraoui. How to allocate tasks asynchronously. In *Proceedings of the 53rd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 331–340, 2012.

[9] James Aspnes. Randomized protocols for asynchronous consensus. *Distributed Computing*, 16:165–175, 2003.

[10] James Aspnes. Randomized consensus in expected $O(n^2)$ total work using single-writer registers. In *Proceedings of the 25th International Symposium on Distributed Computing (DISC)*, pages 363–373, 2011.

[11] James Aspnes. Faster randomized consensus with an oblivious adversary. In *Proceedings of the 31st SIGACT-SIGOPS Symposium on Principles of Distributed Computing (PODC)*, pages 1–8, 2012.

[12] James Aspnes. A modular approach to shared-memory consensus, with applications to the probabilistic-write model. *Distributed Computing*, 25(2):179–188, 2012.

[13] James Aspnes, Hagit Attiya, and Keren Censor. Randomized consensus in expected $o(n \log n)$ individual work. In *Proceedings of the 27th SIGACT-SIGOPS Symposium on Principles of Distributed Computing (PODC)*, pages 325–334, 2008.

[14] James Aspnes, Hagit Attiya, and Keren Censor-Hillel. Polylogarithmic concurrent data structures from monotone circuits. *Journal of the ACM*, 59(1):2, 2012.

[15] James Aspnes, Hagit Attiya, Keren Censor-Hillel, and Faith Ellen. Faster than optimal snapshots (for a while). In *Proceedings of the 31st SIGACT-SIGOPS Symposium on Principles of Distributed Computing (PODC)*, pages 375–384, 2012.

[16] James Aspnes, Hagit Attiya, Keren Censor-Hillel, and Danny Hendler. Lower bounds for restricted-use objects: extended abstract. In *24th ACM Symposium on Parallel Algorithms and Architectures (SPAA)*, pages 172–181, 2012.

[17] James Aspnes and Keren Censor-Hillel. Atomic snapshots in $O(\log^3 n)$ steps using randomized helping. In *Proceedings of the 27th International Symposium on Distributed Computing (DISC)*, pages 254–268, 2013.

[18] H. Attiya, A. Bar-Noy, D. Dolev, D. Peleg, and R. Reischuk. Renaming in an asynchronous environment. *Journal of the ACM*, 37(3):524–548, 1990.

[19] Hagit Attiya and Keren Censor. Tight bounds for asynchronous randomized consensus. In *Proceedings of the 39th Annual ACM Symposium on Theory of Computing (STOC)*, pages 155–164, 2007.

[20] Hagit Attiya and Keren Censor. Lower bounds for randomized consensus under a weak adversary. In *Proceedings of the 27th SIGACT-SIGOPS Symposium on Principles of Distributed Computing (PODC)*, pages 315–324, 2008.

[21] Hagit Attiya, Danny Hendler, and Philipp Woelfel. Tight RMR lower bounds for mutual exclusion and other problems. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC)*, pages 217–226, 2008.

[22] Michael Ben-Or. Another advantage of free choice: Completely asynchronous agreement protocols. In *Proceedings of the 2nd SIGACT-SIGOPS Symposium on Principles of Distributed Computing (PODC)*, pages 27–30, 1983.

[23] Michael Bender and Seth Gilbert. Mutual exclusion with $O(\log^2 \log n)$ amortized work. In *Proceedings of the 52nd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 728–737, 2011.

[24] Armando Castañeda and Sergio Rajsbaum. New combinatorial topology bounds for renaming: the lower bound. *Distributed Computing*, 22(5-6):287–301, 2010.

[25] E. W. Dijkstra. Solution of a problem in concurrent programming control. *Communications of the ACM*, 8:569, 1965.

[26] Faith Ellen and Philipp Woelfel. An optimal implementation of fetch-and-increment. In *Proceedings of the 27th International Symposium on Distributed Computing (DISC)*, pages 284–298, 2013.

[27] Michael J. Fischer, Nancy A. Lynch, and Mike Paterson. Impossibility of distributed consensus with one faulty process. *Journal of the ACM*, 32(2):374–382, 1985.

[28] George Giakkoupis, Maryam Helmi, Lisa Higham, and Philipp Woelfel. An $O(\sqrt{n})$ space bound for obstruction-free leader election. In *Proceedings of the 27th International Symposium on Distributed Computing (DISC)*, pages 46–60, 2013.

[29] George Giakkoupis and Philipp Woelfel. On the time and space complexity of randomized test-and-set. In *Proceedings of the 31st SIGACT-SIGOPS Symposium on Principles of Distributed Computing (PODC)*, pages 19–28, 2012.

[30] George Giakkoupis and Philipp Woelfel. A tight RMR lower bound for randomized mutual exclusion. In *Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC)*, pages 983–1002, 2012.

[31] George Giakkoupis and Philipp Woelfel. Randomized mutual exclusion with constant amortized RMR complexity on the DSM. Manuscript, under review, 2014.

[32] Wojciech Golab, Lisa Higham, and Philipp Woelfel. Linearizable implementations do not suffice for randomized distributed computation. In *Proceedings of the 43rd Annual ACM Symposium on Theory of Computing (STOC)*, pages 373–382, 2011.

[33] Maryam Helmi, Lisa Higham, and Philipp Woelfel. Strongly linearizable implementations: Possibilities and impossibilities. 2012. Manuscript.

[34] Danny Hendler and Philipp Woelfel. Adaptive randomized mutual exclusion in sub-logarithmic expected time. In *Proceedings of the 29th SIGACT-SIGOPS Symposium on Principles of Distributed Computing (PODC)*, pages 141–150, 2010.

[35] Danny Hendler and Philipp Woelfel. Randomized mutual exclusion with sub-logarithmic RMR-complexity. *Distributed Computing*, 24(1):3–19, 2011.

[36] M. Herlihy and J. M. Wing. Linearizability: A correctness condition for concurrent objects. *ACM Transactions on Programming Languages and Systems*, 12(3):463–492, 1990.

[37] Maurice Herlihy and Nir Shavit. The asynchronous computability theorem for t-resilient tasks. In *Proceedings of the 25th Annual ACM Symposium on Theory of Computing (STOC)*, pages 111–120, 1993.

[38] Prasad Jayanti, King Tan, and Sam Toueg. Time and space lower bounds for nonblocking implementations. *SIAM Journal on Computing*, 30(2):438–456, 2000.

[39] Valerie King and Jared Saia. Byzantine agreement in polynomial expected time. In *Proceedings of the 45th Annual ACM Symposium on Theory of Computing (STOC)*, pages 401–410, 2013.

[40] Andreas Klappenecker, Hyunyoung Lee, and Jennifer L. Welch. Dynamic regular registers in systems with churn. *Theoretical Computer Science*, 512:84–97, 2013.

[41] M. C. Loui and H. H. Abu-Amara. Memory requirements for agreement among unreliable asynchronous processes. *Advances in Computing Research*, 4:163–183, 1987.

[42] Thomas Moscibroda and Rotem Oshman. Resilience of mutual exclusion algorithms to transient memory faults. In *Proceedings of the 30th SIGACT-SIGOPS Symposium on Principles of Distributed Computing (PODC)*, pages 69–78, 2011.

[43] Abhijeet Pareek and Philipp Woelfel. RMR-efficient randomized abortable mutual exclusion. In *Proceedings of the 26th International Symposium on Distributed Computing (DISC)*, pages 267–281, 2012.

[44] Jared Saia. BIRS workshop on "probabilisitc versus deterministic techniques for shared memory computation". `http://jsaia.wordpress.com/2012/02/14/`, 2012.

[45] Leslie G. Valiant. A scheme for fast parallel communication. *SIAM Journal on Computing*, 11(2):350–361, 1982.

[46] Andrew Chi-Chih Yao. Probabilistic computations: Towards a unified measure of complexity. In *Proceedings of the 17th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 222–227, 1977.