ACM SIGACT News Distributed Computing Column 6

Sergio Rajsbaum*

January 25, 2002

Abstract

The Distributed Computing Column covers the theory of systems that are composed of a number of interacting computing elements. These include problems of communication and networking, databases, distributed shared memory, multiprocessor architectures, operating systems, verification, internet, and the web.

This issue consists of two parts:

- a survey of DISC'01 by Panagiota Fatourou, and
- a survey of Self-Stabilization at WSS'01 and DISC'01 by Ted Herman.

Many thanks to Gioula and Ted for contributing to this issue.

Request for Collaborations: Please send me any suggestions for material I should be including in this column, including news and communications, open problems, and authors willing to write a guest column or to review an event related to theory of distributed computing.

Report on DISC 2001

Panagiota Fatourou
Department of Computer Science
University of Ioannina
45110 Ioannina, Greece
faturu@cs.uoi.gr

The 15th International Symposium on DIStributed Computing (DISC 2001) was held October 3-5 in Lisbon, Portugal. There were 70 submissions, 8 of which were submitted to both the regular and the brief announcement tracks. The program of the conference included 23 out of the 70 submitted papers at a 32% acceptance rate. The symposium proceedings include only accepted regular papers, and have been published by Springer in its "Lecture Notes in Computer Science" series, LNCS #2180. The 5 accepted brief announcements have been published in a technical report (DI-FCUL, TR-01-7) by University of Lisbon.

DISC'01 was held in conjuction with the 5th Workshop on Self-Stabilizing Systems (WSS 2001). Although WSS had its own program committee and its submitted papers were reviewed separately,

*Compaq Cambridge Research Laboratory, One Cambridge Center, Cambridge, MA 02142-1612. Sergio.Rajsbaum@compaq.com, rajsbaum@math.unam.mx. On leave from Instituto de Matemáticas, UNAM.

DISC-WSS joint submissions were allowed, and 19 out of the 70 submitted papers of DISC were jointly submitted to WSS; therefore, a large number of submissions were on the topic of self-stabilization. The rest of submitted papers addressed a wide variety of subjects, including fault-tolerance of distributed computing, networking and distributed computing issues on the internet and the web, distributed data structures, distributed databases, distributed operating systems, mobile computing, multiprocessor algorithms, consistency conditions, specification, semantics and verification of distributed systems, distributed algorithms and complexity. The distribution across topics of submitted and accepted papers was fairly uniform.

In addition to the regular and brief announcement sessions, the conference featured two invited talks by Gerard Le Lann (INRIA, France) and David Peleg (Weizmann Institute, Israel), and a rump session which took place at the end of the first day, immediately after the business meeting.

This year there was an election of a steering committee member during the business meeting. The DISC steering committee consists of seven members, the chair and the vice chair (elected for two years), a troika of the past, present and next program committee chairs (each serves for three years), and two additional members (elected for two years). Every even year the chair of the steering committe steps out and the vice chair becomes chair, while a new vice chair is elected. Every odd year the additional members step out and there is an election of two new additional members. This year, it was decided, after election, that the policy of electing steering committee members should slightly change: If an additional member has to be elected on an even year, s/he is elected for three and not for two years. Alex Shvartsman (University of Connecticut, USA) was elected this year a member of the steering committee for the next two years.

Other topics discussed during the business meeting include the possibility of having a set of the accepted papers published as invited papers to a journal, as well as the possibility of changing DISC dates to be six months apart from PODC. DISC 2002 will be held October 28-30 in Toulouse, France.

A formal web site for DISC has been constructed, and it is available (from this year on) at the URL: `http://www.disc-conference.org/`, containing links to DISC conferences of past years, online versions of the papers accepted to recent DISCs and other information about the conference. The DISC web page contains a link to the DISC 2001 home page, which includes links to the online versions of the regular papers and brief announcements, slides of the invited speakers' presentations, as well as other interesting information about DISC'01.

As in previous years, DISC'01 had around 80 participants. The local arrangements were excellent and everything worked out smoothly. Lisbon is a very nice city with a lot of sight-seeings and beautiful colors.

Best Student Paper Award, Invited Talks

This year's Best Student Paper Award was given to Yong-Jik Kim (University of North Carolina, USA) for his paper "A Time Complexity Bound for Adaptive Mutual Exclusion", coauthored with James Anderson (University of North Carolina, USA). The paper studies the time complexity of adaptive mutual exclusion algorithms; *adaptive* are the algorithms whose time complexity is a function of the number k of actually contending processes and not the total number N of processes in the system. The *time complexity* is measured in the paper as the maximum number of remote memory references generated by any process in order to enter and execute its critical section once and then become inactive again.

The paper presents a time complexity lower bound of $\Omega(k)$, if N is sufficiently large relative to k. More specifically, the authors prove that for any k, and for any mutual-exclusion algorithm based on atomic reads, writes or comparison primitives, and executed in a system with sufficiently many processes, there exists a computation involving $\Theta(k)$ processes in which some process performs $\Omega(k)$ remote memory references for entering and executing its critical section. This lower bound is interesting because it precludes the design of any deterministic $o(k)$ adaptive, mutual-exclusion algorithm. In particular, it gives a negative answer to the (previously open) question whether there exists an adaptive mutual-exclusion algorithm with time complexity $O(\log k)$. Finally, the authors address the issue of using randomization to overcome this lower bound.

The two invited talks of DISC'01 were very interesting and motivating, and they stressed the need for more work on the respective areas.

The first invited talk was entitled "Is Asynchronous Real-Time an Oxymoron?" by Gérard Le Lann of INRIA, France. Le Lann started by explaining the steps through which a real problem $R\Pi$ can be solved as a computer science problem Π. He pointed out a set of proof obligations that must be respected by any design solution Δ. Obviously, Π should be a correct translation of $R\Pi$, and Δ should solve Π. This means that there should be presented proofs that each of the properties of Π (safety, liveness, timeliness, dependability, etc.) holds, under the problem's and the design solution's assumptions with higher probability than the property's coverage; the *property's coverage* is a lower bound on the probability that the property is not violated at run time. The speaker next stressed why choosing the appropriate design assumptions is not an easy task; infact, the coverage of the design assumptions must be higher than the coverage of the set of the problem properties. Then, Le Lann described some of the properties of Π (safety properties) that must be always true at run time despite the fact that some other properties (e.g., timeliness properties) may be violated.

At the second part of his talk, the speaker briefly discussed the three major models of computation: synchronous, asynchronous, and partially synchronous, and he brought our attention to the fact that the asynchronous model can be augmented with time-free semantics, still maintaining its asynchrony property. (For instance, an asynchronous model augmented with Chandra and Toueg's weak failure detectors is still asynchronous.) He next pointed out the difficulty in solving real-time problems still maintaining the proof obligations mentioned above. He mainly insisted on two requirements for the design of real-time problem solutions: (1) don't assume too much, and (2) don't expect too much. Unfortunately, the real-time problems that meet these requirements, called *elementatry problems*, are not always interesting. Since most interesting problems cannot be expressed as elementary problems, solving a real-time problem is a challenging task, and any claimed solution must be carefully crutinized.

Le Lann devoted the third part of his talk to the importance of instantiating the model. He argued that a good model should respect several proof obligations, which he described in his talk. Essentially, he characterized such a model for real-time computing, as one which is immune to any adversary, while simultaneously is not far from an elementary model (that is, one with elementary assumptions like, e.g., the asynchronous model). He said: "Synchronous Distributed may be an Oxymoron", criticizing the synchronous model for the obvious reason that it is impossible to implement it efficiently in most types of distributed systems. The speaker then presented, criticized and compared designs, problems and solutions under both the synchronous and the asynchronous approach. He insisted again that, sooner or later, asynchrony must be taken under consideration for obtaining a valid design, and he described a set of guiding principles for designing a "real" solution for a real-time problem under an appropriately augmented asynchronous model.

The talk ended by stressing that a lot of work on real-time computing is still to be done, and by presenting a set of concluding remarks, the most major of which was that "Asynchronous Real-Time is not an Oxymoron", answering the corresponding question of the presentation's title.

The second invited talk was "Compact and Localized Labeling Schemes for Graphs" by David Peleg of Weizmann Institute, Israel. The major question considered in the talk was: "Given a graph G with n nodes, are there efficient and compact ways to associate a (preferably short) label with any node such that labels allow inferring of information locally, without using additional memory?" Peleg argued for the importance of designing labeling schemes for several different types of information, and pointed out that labeling mechanisms provide methods for storing and extracting such information in a distributed fashion.

The speaker presented a comprehensive set of research results on labeling schemes for several different information functions. He started with the simple case of adjacency labeling; given the labels of two nodes, decide with no other information whether the nodes are adjacent. Peleg described solutions for trees, as well as extensions to other graph types, for example, bounded arboricity graphs and intersection graphs, and discussed negative results for "large" graph families, e.g., bipartite or chordal graphs.

The second part of the talk focused on distance labeling schemes, whose goal is to provide labeling algorithms for assigning short labels encoding distances between any two nodes, and decoding algorithms for computing such distances given only the nodes' labels. Peleg brought to our attention the fact that this problem can be trivially solved for highly regular graph classes, for example, rings, meshes, tori or hypercubes, but as he pointed out it becomes more challenging for other types of graphs, while it cannot be solved efficiently (using logarithmic only length labels) for very large graph classes. He gave an overview of research results on distance labeling schemes for several types of graphs, and he presented and analyzed in detail an elegant distance labeling scheme based on recursive partitioning. Peleg next introduced approximate distance labeling in which the decoding algorithm can only estimate the distance between two nodes given just their labels. He presented and analyzed a $\sqrt{\log n}$-approximate distance labeling scheme for any weighted graph G, based on tree covers of the graph, which uses labels of length $O(\log diam(G) \log^2 n)$. Several other upper and lower bounds for approximate distance labeling were also discussed.

The speaker described a set of interesting applications of approximate labeling schemes. He first explained how approximate distance labeling can be applied to achieve best service connection setup in a circuit-switched communication network with quiescence time and memory requirements within a polylog factor of optimum. As a second example, he examined how memory-free routing can be implemented efficiently using approximate labeling schemes; the goal of such a system is to route messages between two network nodes relying solely on their labels.

The last part of the talk was devoted to an extensive set of research results on labeling schemes for other information functions, including ancestors, separation levels, least common ancestors (LCA), center and Steiner trees, flow and connectivity.

Concluding the talk, the speaker discussed future applications of labeling schemes, including the design of bounded broadcast protocols and topology update mechanisms, the optimization of multicast schedules, group communication, certain information representation problems on the Web, and computational geometry.

Brief Survey of Regular Papers

We conclude this review with a brief survey of the technical papers presented at the conference.

The theme of mutual exclusion is studied in one more paper. Joung presents a novel quorum system that supports group mutual exclusion, a generalization of mutual exclusion that allows processes of the same group to simultaneously share a resource but requires mutual exclusion among processes of different groups. The proposed quorum system is geometrically evident and it has a nice structure, which makes it is easy to construct. When used with Maekawa's algorithm (ACM TOCS, 1985) it allows up to $\sqrt{\frac{2n}{m(m-1)}}$ processes to have simultaneous access to a resource, where n is the number of processes and m is the number of groups in the system.

An interesting (and usually hard) problem in distributed computing is to characterize the relations that can be computed under specific models of computation. The paper of Boldi and Vigna gives an elegant characterization of problems that can be solved on classes of anonymous networks under a wide range of communication models (synchronous, asynchronous, and interleaved). The presented characterization theorems are based on the notion of *graph fibration*, which is a weakening of the notion of graph coverings. The proof technique employed in the paper turns into a protocol computing the corresponding relation or an impossibility proof if the class of networks under examination and the relation to be computed are finite.

Fault-tolerance is one of the most major topics in distributed computing research. A number of the accepted papers address fault-tolerance issues. Jiménez-Peris, Patinõ-Martinez, Alonso and Arevalo present a non-blocking distributed commit protocol which encompasses a set of new features in order to get combine the low latency of the two-phase commit algorithm with the non-blocking behavior of the three-phase commit algorithm. The new protocol uses optimism to achieve low latency and the non-blocking property, and employs a replicated set of sites as "committ servers" to achieve scalability. Finally, it requires fewer forced writes to the log for better efficiency.

The design of wait-free algorithms, as a mechanism for computing in the face of variable process speeds and failures, has been a very active area of research. In particular, many efforts were invested in constructing long-lived, adaptive, wait-free shared objects. An adaptive, long-lived wait-free $O(k^2)$-renaming algorithm for the read/write shared memory model is presented by Inoue, Umetani, Masuzawa and Fujiwara. The algorithm achieves better space and time complexity than the previous best algorithm of Attiya and Fouren presented in DISC'00 (in its memory bounded version, provided as a manuscript by Afek, Attiya, Fouren, Stupp and Touitou). The authors have described weaker and more efficient mechanisms that can replace some parts of the previous algorithm, and they are still sufficiently strong to provide the properties needed by the algorithm.

The k-set agreement problem has been studied extensively in the literature of distributed computing. A well-known lower bound for the k-set consensus problem is that it requires $\lfloor f/k \rfloor + 1$ rounds in a synchronous, message-passing model with f crash failures. All known proofs of this result are based on complicated connectivity reasoning of algebraic topology, which usually mix semantic, combinatorial and topological arguments. In fact, analyzing connectivity in the synchronous model is a difficult task because the connectivity of the protocol's complex changes from round to round. The paper of Herlihy, Rajsbaum and Tuttle presents a new simpler and shorter proof for this lower bound of k-set consensus. The major idea in the paper is to organize the global states (simplexes) of the protocol into a partial order by introducing one more round to the protocol complex, which makes it easier to prove connectivity arguments in the synchronous model.

Georgiou, Russell and Shvartsman study the synchronous, iterative Do-All problem, where N tasks are to be performed in a system of p-failure prone processors. The efficiency of Do-All algorithms is measured in terms of work complexity, the total number of processing steps taken by the processors. The paper refines previous upper and lower bounds, capturing the dependence of work not only on n and P, but also on f, the number of processor crashes. A new approach for analyzing iterative Do-All algorithms is introduced, whose major feature is that the cost for tolerating failures while performing work and the cost for load-balancing is measured separately. The utility and generality of this approach is illustrated by improving the analysis of two known Do-All algorithms.

Several distributed file systems employ replication of data to increase fault-tolerance and enhance availability. Two of the accepted papers are motivated by this issue. The paper of Douceur and Wattenhofer studies the performance of three strategies for placing file replica in a distributed system with the goal to maximize the availability of the files. The authors concentrate on hill-climbing algorithms that perform a sequence of swap operations, in each of which the machine locations of two file replicas are exchanged; such swapping is done only if it causes the absolute difference of the availabilities of the files to increase. The authors employ competitive analysis to theoretically evaluate their strategies, and they perform simulations driven by actual measurements to verify the good behavior of their algorithms in practice. In related reearch, Malkhi, Pavlov and Sella study the problem of diffusing updates to replicas in the presence of Byzantine faults, in a system where digital signatures cannot be used. The authors prove that moving from the conservative approach, in which a replica sends an update to another replica only if it is sure of the update's veracity, to a more "liberal" approach improves the complexity of algorithms. A liberal algorithm with better complexity than a previous lower bound on the complexity of any conservative algorithm is presented, and a matching lower bound is proved.

The paper of Aguilera, Delporte-Gallet, Fauconnier and Toueg provides a bridge between the research areas of failure detectors and self-stabilization. The authors introduce the concept of a stable leader election algorithm, which ensures that once a leader is elected it remains the leader for as long as it does not fail. They study this problem in a complete message passing network with process crashes and message losses. In essence, a failure detector Ω plays the role of a leader elector: the process currently trusted to be up by another process p is considered to be the current leader of p, and Ω ensures that eventually all processes have the same leader. Under this setting, a number of stable, leader election algorithms are presented, which are communication-efficient and robust. The presented algorithms are mentioned to be self-stabilizing.

The issue of self-stabilization is the subject of a number of other accepted papers. The paper of Higham and Liang contributes the first self-stabilizing algorithms for constructing a minimum spanning tree in a message-passing network which can change dynamically. The algorithms presented in the paper differ from the Gallager-Humblet-Spira approach. The first starts with any spanning tree and converts it into a minimum spanning tree by replacing, if necessary, maximum edges in "near circles" with smaller edges. The second is a general self-stabilizing minimum spanning tree algorithm, which, although it uses some ideas of the first algorithm, is developed from scratch.

Fich and Johnen have presented a self-stabilizing leader-election algorithm on a uni-directional prime-size ring of identical processors, which has optimal space complexity. Their result reduces the gap between the best space complexity upper and lower bounds to a constant, resolving a question by Burns and Pachl that has remained open for several years. The new algorithm employs ideas of the Burns and Pachl's algorithm, but it also uses the technique of time sharing the space to achieve better space complexity: two pieces of information are carefully stored alternately in one variable instead of using two different variables.

The paper of Duflot, Fribourg and Picaronny, applies Markov chain theory to the analysis of randomized, self-stabilizing algorithms. More specifically, Markov chain theory is employed (1) to prove the convergence of randomized self-stabilizing algorithms under both memoryless and adversarial schedulers, and (2) to study the expected convergence time under memoryless schedulers. The area of specification, semantics and verification of distributed systems is further represented in DISC'01 by the paper of Mittal and Garg, which, however, is not specifically related to self-stabilization. The paper generalizes the notion of "slices", an abstraction to facilitate the analysis of distributed computations. The authors prove that for any distributed computation and any global predicate concerning the computation there exists a *slice*, that is, a subcomputation containing those consistent cuts of the original computation that satisfy the predicate. The paper also provides an optimal algorithm to compute slices for special classes of predicates. Applications presented in the paper illustrate that slicing significantly facilitates the global property evaluation of distributed systems.

The field of distributed data structures continues to be a major topic in distributed computing research and it is the subject of four accepted papers. Two of these papers study self-stabilizing algorithms for constructing distributed data structures connecting the field to self-stabilization. Distributed queuing is a fundamental problem arising in a variety of distributed applications. Herlihy and Tirthapura's paper investigates how to self-stabilize the Arrow protocol (presented by Demmer and Herlihy in DISC'98), which is a protocol for implementing distributed queues based on path reversal on a network spanning tree. They propose a new self-stabilizing protocol, which executes on top of any self-stabilizing spanning tree algorithm, and, surprisingly, achieves self-stabilizing distributed queueing in constant time, establishing that self-stabilizing distributed queueing is no more difficult than self-stabilizing spanning-tree maintenance. A self-stabilizing system of replicated search trees is described by Herman and Masuzawa. The paper presents a new self-stabilizing broadcast algorithm which is used to guarantee the coherence of the replica. This algorithm is combined with previous work of the authors to support self-stabilizing replicated search trees with local find operations and global, linearizable insert and delete operations.

The paper of Harris presents a new non-blocking, linearizable implementation of linked lists, based on the compare-and-swap operation. The new feature of the algorithm is to perform deletions in two stages, the first of which logically deletes the node by marking a pointer without changing it, while the second physically deletes the node by removing a marked node from the list. Compare-and-swap is used once in each stage to achieve synchronization. The proposed implementation improves on the best previous algorithms not only because it is simpler, but also because it achieves better performance.

In recent years, a lot of research has been done on counting networks, which are highly-distributed and loosely-coordinated data structures implementing *fetch&increment* counters. Fatourou and Herlihy introduce *adding networks*, distributed data structures that support the concurrent, lock-free, low-contention implementation of the more general *fetch&add* counters; counting networks are just an instance of adding networks that support only *fetch&increment* operations. The authors prove a lower bound stating that any lock-free, low-contention adding network, even those that support addition by only two distinct values, must have high latency, even in the absence of concurrency. This lower bound is shown to be tight. The authors present the first concurrent, lock-free, low-contention network data structure that supports arbitrary *fetch&add* operations.

The area of networking continues to be active, and was nicely represented at DISC'01. Peleg and Pincas study the virtual path layout problem, where a collection of virtual paths must be established over a network. An efficient design of a system of virtual paths guarantees that there exists an upper bound on the number of virtual paths traversing each edge of the network graph, while additionally the (average or maximum) hop count measure (that is, the number of edges along the virtual path) must be small. The authors establish a connection between the average and the maximum hop-count measure for a family of graphs, called *non-condensingly contractable graphs*; paths, cycles and trees are contained in this family. They also present lower and upper bounds on the optimal hop count for various classes of graphs. An algorithm for finding the minimum average hop-count over all directed trees is presented; the algorithm employs dynamic programming and works for the one-to-all communication model.

The paper of Barrière, Fraigniaud, Kranakis and Krizanc investigates the problem of routing in long range contact graphs. In addition to the underlying topology, such graphs contain extra links to connect nodes to some of their "long distance neighbors"; these extra links, called *long* *range links*, represent the knowledge that a node a priori has about its long distance neighbors, and are set up randomly. Long range contact graphs provide a simple model for the "small world" phenomenon which is related to the web and to peer-to-peer networks. The paper presents a comprehensive study of the performance of greedy routing in rings which are augmented with long range links based on the harmonic distribution. According to previous results, efficient routing can be achieved in a grid augmented with long range links which have been set up using the harmonic distribution; on the contrary, the routing performance of a ring augmented with long range contacts based on other distributions is bad. The paper builds upon these results to get the best of both worlds.

An interesting but largely (theoretically) unexplored direction is communication in ad-hoc mobile networks. Chatzigiannakis, Nikoletseas and Spirakis study this problem assuming total absence of any fixed network infrastructure. A straightforward solution to the problem is having the protocol notifying every user that the sender meets hoping that some of them will eventually meet the receiver. The major goal of the paper is to derive more efficient techniques that avoid flooding the network. In previous work, the authors presented an approach based on imposing constraints on the movement of all hosts in the network. This paper significantly improves those results by proposing the much more flexible idea of using only a small subset of the deployed hosts for network operations, while the vast majority of the hosts are moving arbitrarily according to application needs.

Autonomous mobile agents are widespreadly used in many network applications including the web. One of their major tasks is searching, i.e., locating information, resources or generally "items" in the network. The paper of Dobrev, Flochini, Prencipe and Santoro addresses a related problem, where mobile agents are searching a ring network for a highly harmful item, called *black hole*, in the network. The black hole is a stationary process and destroys all agents visiting its node, without leaving any evidence; for example, it could be a local program that severely damages visiting agents. The agents are assumed to work asynchronously, to have limited computing capabilities and bounded storage. The authors give tight upper and lower bounds on the number of agents sufficient to locate the black hole and the time complexity for a wide spectrum of settings in different variations of the model. Searching is also considered in Hoepman's paper, although in a different direction. In the mutual search problem, k agents distributed over the network are required to learn each other's location. Previously published work on randomised mutual search considered only the case of two agents. Hoepman's paper generalizes randomized mutual search to the case of many agents, by presenting upper and lower bounds for a wide class of randomized mutual search algorithms (including adaptive such algorithms) for $k > 2$ agents.

In conclusion, DISC'01 regular papers were of high quality, represented a wide range of topics in distributed computing research, and addressed a number of open problems still to be considered by the distributed computing community.

Acknowledgments

Many thanks to Jennifer Welch for her nice feedback on earlier versions of this report. Thanks also to André Schiper and Michel Raynal for having a look at the report.

Self-Stabilization at WSS'01 and DISC'01

Ted Herman
Department of Computer Science
University of Iowa
Iowa City IA 52240
herman@cs.uiowa.edu

The Fifth International Workshop on Self-Stabilizing Systems (WSS 2001) was held this year in conjunction with the Fifteenth International Conference on Distributed Computing (DISC 2001) during the first week of October, in Lisbon, Portugal. WSS 2001 had more submissions than any previous WSS—27 papers, of which 14 appear in the proceedings, which are published as Springer LNCS number 2194. An invited presentation, four short presentations, and an open problems session complemented the accepted paper presentations at WSS 2001. Remarkably, the proceedings of DISC 2001 include four papers directly concerned with self-stabilization, another paper that introduces new self-stabilizing algorithms, and a sixth paper that analyzes self-stabilizing algorithms. The combination of these two events provides excellent material for a report on the current trends in the area.

The model of mobile agents was the subject of the workshop's invited presentation, *Cooperating Mobile Agents and Stabilization*, by Sukumar Ghosh. In a previous paper presented at COCOON 2000, Ghosh introduced the idea of mobile agents for stabilization, which is a convenient and attractive technique for adding stabilization to a non-stabilizing system. In essence, a mobile agent acts as a travelling repair entity, with its own state and program. A home process uses timeout to trigger agent creation; the agent's task is to diagnose the system configuration and change local process states to enforce convergence to a legitimate configuration. Agents are invisible to normal system operation, and thus do not assist normal system function. One attraction of this model is that it explicitly factors out stabilization effort from other system tasks, so that we can separate concerns and measure their costs separately. In the invited talk, Professor Ghosh considered the question of how to use multiple agents to speed up stabilization, presenting two illustrative examples—one of spanning tree repair and the other of maximum finding in a ring. In the case of the spanning tree, using multiple agents reduces both time and message complexities. For the latter example, it turns out that multiple agents add to message complexity but do not speed up convergence.

The theme of agents in stabilization is found in two other workshop papers. A paper by Beauquier, Hérault and Schiller emphasizes the point that the technique of mobile agents can more easily be tuned (frequency of generation, priority of migration) than traditional stabilization techniques. Moreover, some impossibility results from the literature of stabilization can be bypassed using agents that are inserted by entities external to the system. The paper presents alorithms for agent traversal, leader election, naming, local mutual exclusion (dining philosopher) and token circulation. The token circulation is based on a binary state token-passing protocol that is not stabilizing, leaving the task of convergence to an agent. This reduces the normal system state complexity below what is required for traditional self-stabilization. Using agents can impose less overhead than traditional stabilization techniques because the agent is not required for legitimate configurations. The issue of agent traversal is the subject of the workshop paper by Herman and Masuzawa. They introduce the problem of n agent traversal, which requires that n agents (one per node) visit all nodes, subject to the constraint that no two agents are present simultaneously at any node — thus agents must swap locations to move about the system. The paper gives lower and upper time bounds for synchronous and asynchronous models for tree topologies. Related to this agent work is a workshop paper by Sébastien Tixeuil on network traversal. His algorithm achieves space complexity of d states per node, where d is the outdegree of the node, for Eulerian traversal of general graphs in a message-passing model (further considering cut-through message routing). Establishing an Eulerian traversal provides a virtual ring on any connected topology and thus provides a layer that adapts ring-based algorithms to general graphs. Franck Petit's contribution to the workshop demonstrates that the time complexity of depth-first search traversal, without IDs, in a network with an initiator, is $O(n)$, surpassing the best previous bound of $O(n \cdot Diam)$.

A number of papers in recent years have proposed weakened forms of self-stabilization, and this was also a theme of this year's workshop. Some long-standing problems are the difficulty of designing self-stabilizing systems and the resource costs of self-stabilization. Therefore, it is natural to look for weaker, but still useful, types of "stabilization" to solve these problems. Mohamed Gouda, who presented *The Theory of Weak Stabilization* at the workshop, has for some years advocated weakening the definition of stabilization. His paper investigates the replacement of "for all executions" to "there exists an execution" in the definition: a system is *weakly stabilizing* if, starting from any initial configuration, there is some possible execution leading to a legitimate configuration (and the closure requirement specifies stability thereafter). Weak stabilization is shown to have many nice composition properties, and it has an interesting advantage over traditional stabilization. In practice, implementations of stabilizing algorithms can introduce delays (e.g., extra buffering, non-atomic variable assignment, etc.) which break the property of self-stabilization. Weak stabilization does not suffer from this drawback. In other papers from distributed computing literature, a more typical weakening of the definition of self-stabilization is to assume a restriction on the set of possible initial configurations, rather than allowing entirely arbitrary ones. Two workshop papers do this, the paper by Tixeuil already mentioned, and a paper of Herman and Pirwani. The Tixeuil paper considers the case where channels states are limited, but node states are arbitrary, calling such algorithms *node stabilizing*; the Herman-Pirwani paper considers data structures in arbitrary initial states, but assumes that the internal variables of operations on the data structures are correct. In an entirely different direction, a paper by Dolev and Herman defines weak stabilization to permit arbitrary initial configurations, but does not insist on deterministic closure. A system has the weak stabilization property if, with probability 1, every execution reaches a legitimate state, but from a legitimate state there is a finite probability of a transition to an illegitimate state. This type of weak stabilization is, for instance, seen in some analysis of control systems. The main topic of the Dolev-Herman paper is the use of weaker than atomic register communication in the context of self-stabilization. Several variants of Dijkstra's first self-stabilizing ring algorithm are presented for safe, regular, and probabilistically correct registers.

The probabilistically correct registers used in the Dolev-Herman papers are, for synchronous executions, analyzed by Markov chains to prove convergence. A paper of Duflot, Fribourg and Picaronny, presented at DISC, formalizes the analysis of self-stabilizing algorithms using Markov chains, and then derives expected time bounds for stabilization. A workshop paper by Beauquier, Gradinariu, and Johnen formally shows that randomized scheduling algorithms can be used to support the assumption of fairness in composing self-stabilizing programs. The scheduler is combined with an arbitrary program by an operation called *cross-over composition.* Proofs in their paper combine adversarial schedulers with randomized scheduling algorithms into game trees that facilitate correctness proofs of probabilistically stabilizing algorithms.

Scheduling continues to be a core topic in self-stabilization research. A lucky schedule, for example, transforms Gouda's weak stabilization into self-stabilization. The cross-over composition of Beauquier et. al. is entirely devoted to the task of scheduling. An interesting technical use of scheduling is found in the DISC paper by Fich and Johnen. Their result settles a long open question about the space complexity of leader election in a uniform, unidirectional ring of prime size: the gap between upper and lower bounds on space complexity is reduced to a constant. The algorithm relies on an *alternating schedule*, which specifies that between every two steps of each process, there is exactly one step of each of its neighbors. A constant-space implementation of an alternating scheduler was published in PODC 1999, and it is used here to reduce the space requirement for sending information around the ring. Essentially, a process can alternate phases of computation in which information is sent around the ring; processes that propagate this information time-share buffer space for the different phases, thanks to the alternating schedule.

Other workshop and DISC papers treat topics such as network protocols, mutual exclusion, and tree construction. Cobb and Gouda present a self-stabilizing routing protocol for directed networks. Previous stabilizing distance-vector routing protocols assume that links are undirected (i.e., channels are bidirectional) whereas this protocol works, to the extent that the network is strongly connected, in the directed case. A paper by Gouda, Huang and Arora, analyzes security issues for the familiar Internet PING program by including an adversary as part of the state transition diagram for the program, and then asking questions about convergence and stability. Mutual exclusion in a tree network is generalized to ℓ tokens in a paper by Hadid and Villain in such a way that space complexity is independent of ℓ except at the root and convergence time is proportional to tree height. This is also the first self-stabilizing ℓ-exclusion algorithm in the message-passing model. Herlihy and Tirthapura illustrate, in their DISC paper, that some tasks have quite scalable stabilization overhead: their result self-stabilizes the Arrow protocol (introduced in DISC'98) for distributed queuing, proving that it has constant convergence time for any network that has a spanning tree; thus total stabilization time is no more than a constant over the stabilization time for spanning tree construction. Higham and Liang's DISC contribution is the first self-stabilizing algorithm for MST in the message-passing model. Their algorithm uses a novel idea for tree construction, described as having edge processors compete for chosen status, rather than following the familiar GHS method of merging fragments.

Two workshop papers, one by Dolev and Schiller, and another due to Kulkarni and Ravikant, contribute technical improvements for self-stabilizing group communication. The Dolev-Schiller paper defines the problem of self-stabilizing group membership maintenance in an asynchronous system and describes an algorithm that is communication adaptive: the communication overhead for a legitimate configuration significantly improves over the communication complexity during convergence from an arbitrary initial configuration. The Kulkarni-Ravikant paper examines the publish-subscribe paradigm, where they show how time bounds on message delay and clock drift can be used for a bounded stabilizing solution to the core problem of logical timestamps. This enables the construction of causal deterministic merge and delivery with space overhead growing only logarithmically in the number of processes.

The workshop paper by Gärtner and Pleisch connects stabilizing behavior to the task of predicate detection, implemented using failure detectors. Weak forms of failure detectors such as $\Diamond\mathcal{P}$ (eventually perfect) and $\Box\Diamond\mathcal{P}$ (infinitely often accurate) have semantics similar to the convergence of self-stabilization. Gärtner and Pleisch investigate various cases of failure detectors to see when they make possible (or are insufficient) to support several semantics of predicate detection. A connection between self-stabilization and failure detection also appears in the DISC paper by Aguilera, Delporte-Gallet, Fauconnier and Toueg, entitled *Stable Leader Election*. The paper presents algorithms for iterative leader election (in essence, failure detectors work as leader electors, if we consider the good processors to be candidate leaders); these protocols converge to electing the same leader forever. It is mentioned in passing that their algorithms also happen to be self-stabilizing. Their protocols are round-based algorithms, which is typical of failure detectors. Another DISC paper, by Herman and Masuzawa, also uses round-based algorithms. The basic tool developed in that paper is a stabilizing atomic broadcast, which is used to synchronize operations on replicated trees. Operations can be applied at arbitrary configurations, including cases of inconsistent replicas, however operations are guaranteed to have trustworthy (though degraded) response during the period of convergence.

The program committee of WSS 2001 was impressed by the quantity and quality of the submitted papers. The many papers of DISC and WSS exhibit new research areas within self-stabilization and connect well to other established topics of distributed computing.