The Distributed Computing Column

by

Mario Mavronicolas

Department of Computer Science, University of Cyprus
75 Kallipoleos St., CY-1678 Nicosia, Cyprus
mavronic@cs.ucy.ac.cy

A Report from DISC 2005, the 19th International Symposium on Distributed Computing

María J. Blesa[1] and Chryssis Georgiou[2]

[1] *ALBCOM research group, Universitat Politècnica de Catalunya*
Ω-213 Campus Nord, E-08034 Barcelona, Spain
mjblesa@lsi.upc.edu

[2] *Department of Computer Science, University of Cyprus*
75 Kallipoleos Str., P.O. Box 20537, CY-1678 Nicosia, Cyprus
chryssis@cs.ucy.ac.cy

Abstract

This is a review on the 19th International Symposium on Distributed Computing, which took place in Kraków, Poland, on September 26–29, 2005. The proceedings of DISC 2005 are published by Springer, as volume 3724 of the Lecture Notes in Computer Science (LNCS) series. The conference website can be found at www.mimuw.edu.pl/~disc2005.

1 Introduction

DISC 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.

The symposium was established in 1985 as a biannual International Workshop on Distributed Algorithms on Graphs (WDAG). The scope was soon extended to cover all aspects of distributed algorithms as WDAG came to stand for International Workshop on Distributed AlGorithms, and in 1989 it became an annual symposium. To reflect the expansion of its area of interest, the name was changed to DISC (International Symposium on DIStributed Computing) in 1998. The name change also reflects the opening of the symposium to all aspects of distributed computing. The aim of DISC is to reflect the exciting and rapid developments in this field.

This year, DISC got 162 regular submissions and 30 brief announcements' submissions. From them, 32 regular submissions and 14 brief announcements were accepted after the PC meeting on the 1st and 2nd of July 2005, in Paris. There were 90 participants (probably a record) inscribed in DISC, and 25 additional participants who came only for the two co-located workshops, LOCALITY and DYNAMO. All the information related to the conference could be followed at `www.mimuw.edu.pl/~disc2005`.

The 19th edition of DISC took place in Kraków, Poland, on September 26–29, 2005. The conference rooms were mainly placed in the Cracovia hotel, quite close to the city center. The proceedings of the conference are published by Springer, as volume 3724 of its Lecture Notes in Computer Science series. Year 2005 is the first in which there will be a joint DISC-PODC post-conference special issue of *Distributed Computing*.

1.1 The city of Kraków

Kraków was the center of Polish cultural, artistic and academic activity over the centuries. Although Kraków lost its political importance in the beginning of the 17th century, after moving the capital of the country to Warsaw, it remains a place famous for its historical monuments and vibrant artistic life.

Kraków has been settled since the Stone Age at least. In 1038 Kraków became the capital of Poland and Polish monarchs took up their residence in its Wawel Royal Castle. The Old Town historical district in Kraków's heart is actually the medieval city established in 1257 by Prince Boleslav V. The first university in the country was established in Kraków in 1364 by king Casimirus the Great. The Kraków Academy exists up to this day under the name of Jagiellonian University.

In its long history Kraków underwent many ups and downs. The proud capital city of a mighty kingdom for centuries, it was turned into a sleepy borderland

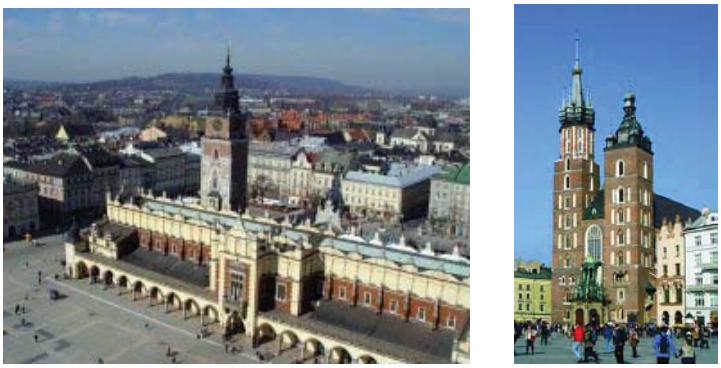


Figure 1: The Cloth Hall (left) and St.Mary's Church(right).

town of the Austrian empire in the 19th century. Then it became a vital center of Polish national awakening at the turn of the 20th century and the cradle of Poland's rebirth, only to be reduced to backwater under communism. Now Kraków is nearly a million city ripe for restoration to European status. And the beautiful Old Town area remains its vibrant hub with numerous landmarks, museums, art galleries, music venues, theaters, university colleges, etc. on top of myriad boutiques, cafes, and restaurants. UNESCO entered the whole of Kraków's Old Town in the list of the world cultural heritage.

The **Wawel** (*Wzgórze wawelskie*) is the name of a hill situated on the left bank of the Vistula river, that places on his top the Royal Castle and the Cathedral. During the middle ages, the history of the Wawel was deeply intertwined with the history of the Polish lands and Polish royal dynasties. As the Polish-Lithuanian Commonwealth formed and grew, the Wawel became the seat of one of Europe's most important states, until the 17th century when Warsaw became the capital. During the period of the partitions, the Wawel became a symbol of the lost nation.

The Main Market Square (*Rynek Główny*) is the natural center of Kraków since the Great Royal Charter in 1257. The centrally located Cloth Hall (*Sukiennice*) has survived to this day; the building was originally a commercial establishment for trading in cloths, and for over a century has been the main seat (and later one of the branches) of the National Museum. Other buildings standing in the heart of the Main Market Square include the diminutive Church of St. Adalbert (*Wojciech* or *Voitek*), and a solitary tower remnant of the Town Hall demolished in the 19th century. In the north-eastern corner of the square stands one of Kraków's landmarks: St. Mary's Church, frequently referred to as a basilica, with its two slender, spired towers. Inside of it, one can find the monumental High Altar of St. Mary's, a marvel that attracts thousands of tourists every day.

Figure 2: Pierre Fraignaud, DISC 2005 PC chair (left). Steering Committee chairs from left to right: Shmuel Zaks, Alex Shvartsman, Michel Raynal, Andre Schiper, Sam Toueg (right)

2 DISC 2005: The 19th Edition

The opening and closing of DISC 2005 was conducted by Alex Shvartsman (University of Connecticut). Pierre Fraigniaud (CNRS and University of Paris Sud) was the **Program chair** of this 19th edition.

The **Steering Committee** was composed by Alex Shvartsman (University of Connecticut), Chair, Paul Vitanyi (CWI and University of Amsterdam), Vice-chair, Hagit Attiya (Technion), Roger Wattenhofer (ETH Zurich), Faith Fich (University of Toronto), Shlomi Dolev (Ben-Gurion University), Pierre Fraigniaud (CNRS, University of Paris Sud), and Rachid Guerraoui (EPFL).

During DISC 2005, Faith Fich (DISC 2003 PC chair) was replaced by Shlomi Dolev (DISC 2006 PC chair). Also, Hagit Attiya whose 2-year term ended at DISC 2005, was re-elected for another 2-year term as member of the Steering Committee. DISC 2005 was the first DISC conference where all five Steering Committee chairs (past and current) were present: Sam Toueg (1996–1998), Shmuel Zaks (1998–2000), Andre Schiper (2000–2002), Michel Raynal (2002–2004) and the current chair Alex Shvartsman (2004–2006).

The **Program Committee** was composed by Lenore Cowen (Tufts University), Panagiota Fatourou (University of Ioannina), Hugues Fauconnier (University of Paris VII), Pierre Fraigniaud (CNRS University of Paris Sud), Chair, Roy Friedman (Technion), Yuh-Jzer Joung (National Taiwan University), Dariusz Kowalski (Warsaw University), Victor Luchangco (Sun Microsystems Laboratories), Maged Michael (IBM T.J. Watson Research Center), David Peleg (Weizmann Institute), Greg Plaxton (University of Texas at Austin), Sergio Rajsbaum (National Autonomous University of Mexico), Sylvia Ratnasamy (Intel Research Laboratory), Nicola Santoro (Carleton University), Sebastiano Vigna (University

of Milano), Jennifer Welch (Texas A&M University).

The **Organizing institutions** were the Warsaw University, Jagiellonian University. The **Organizing Chair** was Dariusz Kowalski (University of Liverpool). The **Organizing team** was formed by Krzysztof Diks and Adam Iwanicki (Warsaw University), Kazimierz Grygiel and Krzysztof Szafran (Foundation for Information Technology Development), Marek Zaionc, and 8 other people from the Jagiellonian University in Kraków.

2.1 Conference program and invited talks

DISC 2005 got 162 regular submissions and 30 brief announcements' submissions. From them, 32 regular submissions and 14 brief announcements were accepted for presentation. Each regular contribution was given 25 minutes for presentation, while each brief announcement was given 6 minutes. Additionally, the program included 2 invited 1-hour talks, which we outline in the following:

Michael Mitzenmacher, Harvard University, USA. *Digital Fountains and Their Application to Informed Content Delivery over Adaptive Overlay Networks.* This invited talk was about informed content delivery over adaptive overlay networks. In the first part of the talk, the speaker introduced and explained the *digital fountains* paradigm. The idea surrounding this paradigm is to stop thinking of data as an ordered stream of bytes, as the standard TCP paradigm "forces us" to do, and instead view data as water from a fountain: you place a cup under the fountain to fill it with water; you do not care which drops of water you get or in which order the drops get into your cup! Since the digital fountain paradigm alleviates the need of ordered data, many applications can benefit from its use (e.g., Multicast and point-to-point data transition, Parallel download, One-to-Many TCP etc). However, a natural question arises: Can we efficiently implement digital fountains? A positive answer was given; digital fountains can be constructed using erasure codes (Raptor codes seem to be the most appropriate to use). In the second part of the talk the speaker described how digital fountains can be used for content delivery over overlay networks and presented a collection of useful algorithmic tools for efficient estimation, summarization, and approximate reconciliation of sets of symbols between pairs of collaborating peers that keep message complexity and computation to a minimum.

Amir Herzberg, Bar Ilan University, Israel. *Securing the Net: Challenges, Failures and Directions.*
The second invited talk was concerned with the problems arising from the insecurity of the Internet. The speaker identified four major reasons that make Internet vulnerable to malicious attacks: (i) the fact that Internet is global and open to

Figure 3: Michael Mitzenmacher and Amir Herzberg, the invited speakers

everybody (including the attackers), (ii) computers are unprotected and not properly managed (insecure platforms are of wide use and many users are naive and not properly trained against attackers), (iii) there is a plethora of untrusted clients and peers, and (iv) there is a plethora of threats. According to the speaker, the most acute threats are email spam, malware (virus, trojans, worms, spyware etc), denial of service, con-sites (fake/spoofed sites, scam/fraud sites), and intrusion. Intrusion seems to be the most dangerous thread, since an intrusion to a system can open the way for all other threats. The speaker then presented a nice all-around overview of issues surrounding these threads and outlined solutions and directions for future applied and analytical research. He also presented TrustBar, a secure user interface add-on to browsers that offers protection for web users, from spoofing/phishing attacks. More on TrustBar can be found at `AmirHerzberg.com/TrustBar`.

2.2 Awards

The best student paper award went to *"Space and Step Complexity Efficient Adaptive Collect"*, by Yaron De Levie and Yehuda Afek, and *"General Compact Labeling Schemes for Dynamic Trees"*, by Amos Korman. Yaron De Levie and Amos Korman shared the award, since Yehuda Afek is not a student.

2.3 Workshops

Two workshops were co-located with DISC 2005: LOCALITY 2005 (Locality Preserving Distributed Computing Methods), and DYNAMO 2005 (Dynamic Commu-

nication Networks: Foundations and Algorithms). From the 115 participants that the 19th edition of DISC obtained, 25 of them came only for the workshops.

The LOCALITY 2005 workshop was chaired by Cyril Gavoille (University of Bordeaux, France), and Dahlia Malkhi (Hebrew University of Jerusalem, Israel, and Microsoft Research, USA). The organizing chair was Dariusz Kowalski (Warsaw University, Poland). Microsoft research sponsored the workshop, including the first 25 registrants' registration fee. The complete information of the LOCALITY 2005 workshop can be found at `www.cs.huji.ac.il/~locality05/`.

DYNAMO 2005 is the workshop of the COST Action 295 *Dynamic Communication Networks: Foundations and Algorithms*, which was chaired by Roger Wattenhofer (ETH Zurich, Switzerland), consisted in a series of invited talks and shared some sessions with DISC. More information can be tracked at the website `www.mimuw.edu.pl/~disc2005/index.php?page=dynamo`.

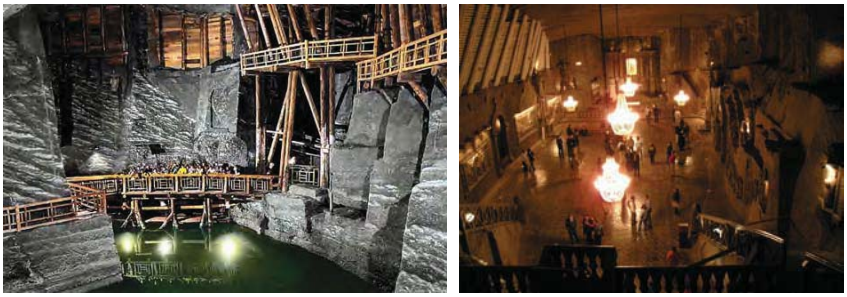
2.4 Social events

The **Reception banquet** took place in the *Collegium Maius*, the oldest college among the Polish universities. The 630-year-old Jagiellonian University moved in to this building in 1400. The Collegium Maius was rebuilt by the end of the 15th century as a splendid late-Gothic edifice around a vast courtyard with surrounding arcades and a well of 1517 in the center. Professors lived and worked upstairs, while lecturing downstairs. Copernicus in the 1490s, and Pope John Paul II, are among the most illustrious of Krakow university's graduates.

The **Conference trip** brought us to the UNESCO World Heritage old Wieliczka Salt Mine. Nine centuries of mining in Wieliczka produced over 250 kilometers of passages as well as 2,040 caverns of varied size. The mine has nine working floors that arrive down to 327 meters below ground. The old miners were deeply religious and held services before starting work. They made chapels underground which were richly decorated with wooden carvings. The largest example is the chapel of St. Kinga, which was started in 1896 and is 101 meters underground, 10 meters high, 15 meters wide and 54 meters long. All the fittings and statuary are carved from salt; even the chandeliers are made from rock salt crystals.

The **Conference dinner** took place in the Wierzynek restaurant, which is located in two ancient buildings at the Main Market Square. The history of the Wierzynek Restaurant dates back to the 14th century. Legend has it that in the year 1364 Polish king Casimir the Great invited monarchs from all over Europe in order to settle an argument that brought the continent at the verge of war. The king, however, had to have a pretext to get all the monarchs together; the forthcoming wedding of his granddaughter with the Emperor Charles of Luxembourg proved ideal for this purpose. The king asked Mikołaj Wierzynek, an affluent merchant,

(a) Courtyard of the Collegium Maius (left), and DISC 2005 reception, banquet and visit in Collegium Maius (right).



(b) In Wieliczka salt-mines: St. Kinga chapel (right).

(c) Conference dinner at the Wierzynek restaurant.

Figure 4: Social events: reception banquet, conference trip and conference dinner

to supervise all the festivities that were to take place on the occasion. It was only after Wierzynek invited the king and his guests to his house for a sumptuous feast (lasting 21 days and nights) that a consensus was reached.

3 Overview on the Scientific Contents

We present an overview of the regular papers presented in DISC 2005. We grouped the papers in research areas in an attempt to emphasize the scientific contributions of the presented papers for specific topics of interest.

3.1 Fundamental problems: cooperation and synchronization

One could consider, that most of the fundamental problems arising in distributed computing are, in their basics, related to cooperation and synchronization of processes. Among others, some of those important problems are *consensus*, *broadcasting*, *collect*, *time-stamping* and *self-stabilization*.

Consensus
In many distributed applications, all the processes that cooperate to achieve a common goal have to share a common view of the state of the system. To build such a common view, processes have to execute an agreement protocol during which each process proposes its own partial view and gets a final value which must be the same for every one. Among all the agreement problems, the consensus problem is the simplest paradigm. The consensus problem is stated as follows. Each process of a set of n processes proposes a value, and each non-faulty process has to decide a value (termination) in such a way that a decided value is a proposed value (validity) and the non-faulty processes decide the same value (agreement). Unfortunately, it has been shown that the consensus problem has no deterministic solution in a purely asynchronous distributed system. To design correct agreement protocols, one needs either to weaken the problem or strengthen the underlying system by adding synchrony assumptions.

In *"Ω Meets Paxos: Leader Election and Stability Without Eventual Timely Links"*, by D. Malkhi (Microsoft Research Silicon Valley, USA, and Hebrew University of Jerusalem, Israel), F. Oprea (Carnegie Mellon University, USA) and L. Zhou (Microsoft Research Silicon Valley, USA) a realization of distributed leader election without having any eventual timely links is provided. Progress is guaranteed by the fact that, eventually, one process can send messages such that every message obtains f timely responses, where f is a resilience bound. A crucial point of this property is that the f responders need not be fixed, and may change from one message to another. In the (common) case where $f = 1$, this

implies that the FLP impossibility result on consensus is circumvented if one process can at any time communicate in a timely manner with one other process in the system.

S. Goldwasser, M. Sudan, and V. Vaikuntanathan (MIT CSAIL, USA), in *"Distributed Computing with Imperfect Randomness"* seek to determine whether it is possible to do with imperfect randomness all that can be done with perfect randomness, and with comparable efficiency. They give a positive answer to this question in the context of the Byzantine Agreement problem (the consensus problem in the presence of Byzantine failures).

Broadcasting

Broadcasting in a computer network refers to transmitting a packet that will be received by every device on the network. In practice, the scope of the broadcast is limited to a broadcast domain, and it is largely confined to local area network (LAN) technologies.

In *"Optimistic Generic Broadcast"*, P. Zieliński (University of Cambridge, UK) considers an asynchronous system with the Ω failure detector, and investigate the number of communication steps required by various broadcast protocols in runs in which the leader does not change. Atomic Broadcast requires 3 communication steps, while Optimistic Atomic Broadcast requires only 2 steps if all correct processes receive messages in the same order, and Generic Broadcast requires 2 steps if no messages conflict. An algorithm that subsumes both of these approaches and guarantees 2-step delivery if all conflicting messages are received in the same order, and 3-step delivery otherwise is presented.

Modern communication networks define dynamic (and often unknown) topologies whose management and implementation technology triggers new algorithmic challenges, being one of them the optimization of the power consumption. One way to reduce power consumption in some ad hoc networks is to employ a power management strategy, such that the system may only be required to operate at full functionality in the presence of a novel object or request. The functionality of a wake-up algorithm is to detect the novel object or request and to arouse the surveillance system to full functionality.

In *"Waking Up Anonymous Ad Hoc Radio Networks"*, A. Pelc (Université du Québec, Canada) considers the task of deterministically waking up an anonymous ad hoc radio network (where the nodes do not know the topology, and some might even be impossible to reach) from a single source, which is the only awake process. A deterministic wake-up algorithm for ad hoc networks is universal if it wakes up all accessible nodes. For synchronous communication, universal wake up algorithms exist, but they do not for the case of asynchronous communication.

Collect

The collect problem can be viewed as the counterpart of the gossip problem in the shared-memory model of computation: An asynchronous system consisting of n processors is assumed. Each processor has its own dedicated register into which it writes new information and it must collect the information written by all other processors. Usually this is done by having each processor reading all other registers. Although such a solution is wait-free, it is not adaptive; the step complexity of a high level operation is a function of the total number of processors in the system and not of the actual number of active processors.

An algorithm is said to be adaptive to total contention if the step complexity of a high level operation is a function only of the total number of different processors that have been active in the algorithm execution before this operation terminates. In *"Space and Step Complexity Efficient Adaptive Collect"*, Y. Afek and Y. De Levie (Tel-Aviv University, Israel) develop a space and step complexity efficient deterministic adaptive to total contention collect algorithm. The algorithm achieves optimal $O(k)$ step and $O(n)$ space complexities, n being the total number of processors and k the total contention, but restricting the processor identifiers size to $O(n)$ (if this restriction is removed, then the space complexity increases to $O(n^2)$ but the step complexity remains $O(k)$). These results improve significantly all other previously presented deterministic adaptive collect algorithms.

Time-stamping

A time-stamp is the current time of an event that is recorded by a computer. The time-stamp mechanism is used for a wide variety of synchronization purposes, such as assigning a sequence order for a multi-event transaction so that if a failure occurs the transaction can be voided. Another way that a time-stamp is used is to record time in relation to a particular starting point.

In a distributed system with n processes, time stamps of size n are necessary to accurately track potential causality between events. Plausible clocks are a family of time-stamping schemes that use smaller time stamps at the expense of some accuracy. To date, all plausible clocks have been designed to use fixed-sized time stamps, and the inaccuracy of these schemes varies from run to run. In *"Plausible Clocks with Bounded Inaccuracy"*, B.T. Moore (Ohio State University, USA) defines a new metric, *imprecision*, that formally characterizes the fidelity of a plausible clock. A new plausible clock system that guarantees an arbitrary constant bound on imprecision is also presented.

The vast majority of papers on distributed computing assume that processes are assigned unique identifiers before computation begins. But is this assumption necessary? In *"What Can Be Implemented Anonymously?"*, R. Guerraoui

(EPFL, Switzerland) and E. Ruppert (York University, Canada) consider asynchronous shared-memory systems that are anonymous. It is investigated for the first time, what can be implemented deterministically when processes can fail. Anonymous algorithms for some fundamental problems are provided, for example for time-stamping, snapshots and consensus. Interestingly, the authors also show that a shared object has an obstruction-free implementation if and only if it satisfies a simple property called *idempotence*: applying any permitted operation to the same object twice in a row has the same effect as applying it once.

Self-stabilization
As first defined by Dijkstra, an algorithm is self-stabilizing when regardless of its initial state, it is guaranteed to arrive at a legitimate state in a finite number of steps. The self-stabilization property is very desirable for distributed algorithms, especially for algorithms operating in unreliable and faulty environments. However, a self-stabilizing algorithm cannot detect by itself that stabilization has been reached. Therefore, the notion of local (deterministic) observer was introduced (by Beauquier, Pilard and Rozoy) whose role is to correctly detect stabilization without influencing the self-stabilization protocol.

In *"Observing Locally Self-stabilization in a Probabilistic Way"* J. Beauquier, L. Pilard, and B. Rozoy (Université Paris-Sud, France) introduce the notion of *probabilistic observer* which realizes the conditions for stabilization with probability 1. They show that if the network is uniform and synchronous, then some problems having a self-stabilizing solution, do not have any self-stabilizing solution that can be observed by a local and deterministic observer, but have a self-stabilizing solution that can be observed by a local and probabilistic observer.

3.2 Failure models, denial-of-service and other attacks

Network failures occur for many different reasons and in many different forms. Failure models aim at capturing the behavior of failures and predict their emergence to be able to minimize the effect. The classic failure models assume that failures are caused by the hardware components of the network. New failure models as, for example, models for failures produced from Denial of Service (DoS) Attacks need to be explored. A DoS attack is characterized by an explicit attempt by attackers to prevent legitimate users from using a certain service, e.g., attempts to flood a network, to disrupt connections between two machines, etc.

In *"Coterie Availability in Sites."*, F. Junqueira and K. Marzullo (University of California, USA), explore new failure models for multi-site systems that allow sites to fail. The use those models to derive *coteries*, which have better availability

than quorums formed by a majority of processes. Different possibilities for obtaining those models in practice are presented. The constructions proposed have substantially better availability and response time compared to majority coteries.

One of the important modern malicious environments that we find is concerned with overcoming distributed DoS attacks by realistic adversaries that can eavesdrop on messages. Two papers cover some aspects from these environments. In *"Keeping Denial-of-Service Attackers in the Dark"*, G. Badishi, I. Keidar (Technion, Israel), and A. Herzberg (Bar Ilan University, Israel) present a protocol that provides effective DoS prevention for realistic attack and deployment scenarios. This protocol works by eavesdropping adversaries, using only available and efficient packet filtering mechanisms based mainly on addresses and (non-fixed) port numbers (performing instead a 'pseudo-random port hopping').

A. Aiyer, L. Alvisi (University of Texas at Austin, USA), and R.A. Bazzi (Arizona State University, USA) consider, in *"On the Availability of Non-strict Quorum Systems"*, how to increase availability in quorums systems while at the same time tolerating a malicious scheduler and guaranteeing an upper bound on the staleness of data. The conditions under which this increase is possible turn out to depend on the ratio of the write frequency to the servers' failure frequency. For environments with a relatively large failure frequency compared to write frequency, this work proposes K-quorums that can provide higher availability than the strict quorum systems and also guarantee bounded staleness.

Another malicious situation in distributed computations is caused by *conspiracies* (a certain class of livelocks). This elementary phenomenon occurs in systems with shared variables, shared actions as well as in message-passing systems. H. Völzer (University of Lübeck, Germany) studied and characterized the phenomenon of conspiracies in *"On Conspiracies and Hyperfairness in Distributed Computing"*. This characterization uses a new notion of hyperfairness, which postulates the absence of conspiracies, and is a useful tool for understanding some impossibility results. As a main result, the author shows that a large subclass of hyperfairness can be implemented through partial synchrony and randomization.

3.3 Wait-freedom, lock-freedom and obstruction-freedom

Non-blocking algorithms are needed to allow multiple distributed threads to read and write shared data concurrently without corrupting it. Three properties are involved in their design: wait-freedom, lock-freedom and obstruction-freedom. *Lock-freedom* refers to the fact that a thread cannot lock up: every step it takes brings progress to the system. This means that no synchronization primitives such as mutex or semaphores can be involved, as a lock-holding thread can prevent global progress if it is switched out. *Wait-freedom* is the strongest property and

it refers to the fact that a thread can complete any operation in a finite number of steps, regardless of the actions of other threads. All wait-free algorithms are lock-free, but the opposite might not be true. *Obstruction-freedom* denies only deadlock and demands that any partially-completed operation can be aborted and the changes made are rolled back. All lock-free algorithms are obstruction-free.

E. Gafni (University of California, USA) and S. Rajsbaum (Universidad Nacional Autónoma de México, México) propose in *"Musical Benches"* the *musical benches problem* to model a wait-free coordination difficulty. The musical benches problem seems like just a collection of consensus problems (where by the pigeon hole principle at least one of them will have to be solved by two processes) and, thus one would be tempted to try to find a bivalence impossibility proof of the FLP style. However, the authors show that there is no such proof. This establishes a new connection between distributed computing and topology. Moreover, by considering benches other than consensus, the musical benches problem can be generalized, leading to a very interesting class of new problems.

In *"Efficient Reduction for Wait-Free Termination Detection in a Crash-Prone Distributed System"*, N. Mittal, S. Venkatesan (University of Texas at Dallas, USA), C. Freiling and L. Draque Penso (RWTH Aachen University, Germany) investigate the problem of detecting termination of a distributed computation in systems where processes can fail by crashing. Specifically, when the communication topology is fully connected, the authors describe a way to transform any termination detection algorithm $\mathcal{A}$, that has been designed for a failure-free environment, into a termination detection algorithm $\mathcal{B}$ that can tolerate process crashes and failures of up to $n - 1$ processes (wait-freedom), and that does not impose any overhead on the fault-sensitive termination detection algorithm until one or more processes crash (fault-reactivity). This transformation can be extended to arbitrary communication topologies provided process crashes do not partition the system.

Obstruction-freedom is weaker than lock-freedom and wait-freedom, and admits simpler implementations that are faster in the uncontended case. Pragmatic contention management techniques appear to be effective at facilitating progress in practice, but none guarantees progress. In *"Obstruction-Free Algorithms Can Be Practically Wait-Free"*, F.E. Fich (University of Toronto, Canada), V. Luchangco, M. Moir and N. Shavit (Sun Microsystems Laboratories, USA) present a transformation that converts any obstruction-free algorithm into one that is wait-free when analyzed in the unknown-bound semisynchronous model. For all practical purposes, obstruction-free implementations can provide progress guarantees equivalent to wait-freedom. The transformation that the authors perform preserves the advantages of any pragmatic contention manager, while guaranteeing progress.

3.4 Concurrent objects and data structures

Concurrent computing is the overlapped coordinated execution of multiple tasks on multiple processors in order to share common resources, some of which might be objects and data structures. Non-blocking algorithms are needed to avoid requiring a critical section and to allow multiple processes to access a structure without ever blocking each other. Those algorithms may involve multi-threading, support for distributed computing, message passing, and shared resources. Hence, designing non-blocking objects and data structures is a very important issue.

In *"Non-blocking Hashtables with Open Addressing"*, C. Purcell (University of Cambridge, UK) and T. Harris (Microsoft Research Ltd., UK) present the first non-blocking hashtable based on open addressing that combines good cache locality with short straight-line code. It does not need neither storage overhead for pointers and memory allocator schemes, nor periodical reorganization or replication, and nor garbage collection. The results are highly-concurrent algorithms that approach or outperform the best tested externally-chained implementations.

H. Attiya (Technion, Israel), R. Guerraoui and P. Kouznetsov (EPFL, Switzerland) study in *"Computing with Reads and Writes in the Absence of Step Contention"* implementations of concurrent objects that exploit the absence of step contention. These implementations use only reads and writes when a process is running solo and the other processes might be busy, swapped-out, failed, or simply delayed. Obstruction-free implementations (which are not required to terminate), and solo-fast implementations (which do terminate) are studied. The authors present a generic obstruction-free object implementation that has a linear contention-free step complexity and uses a linear number of read/write objects (and this is asymptotically optimal). Moreover, obstruction-free implementations are shown not to be non-blocking when the contention manager operates correctly, but remain obstruction-free when the contention manager misbehaves.

In *"Restricted Stack Implementations"*, M. David, A. Brodsky and F.E. Fich (University of Toronto, Canada) introduce a new object, BH (short for *Blurred History*), and use it to provide a characterization of the class of objects that can be implemented from commutative and overwriting objects. Although it was conjectured that Stacks and Queues shared by at least 3 processes do not belong to this class, by using a BH object two different restricted versions of Stacks do belong.

Atomicity is a usual consistency criterion for distributed services and objects. Although atomic object implementations are abundant, to provide algorithms achieving atomicity has turned out to be a challenging problem. In *"Proving Atomicity: An Assertional Approach"*, G. Chockler, N. Lynch, S. Mitra and J. Tauber (MIT CSAIL, USA) initiate the study of systematic ways of verifying distributed implementations of atomic objects. Their general approach is to

replace the existing operational reasoning about events and partial orders with assertional reasoning about invariants and simulation relations. To this end, an abstract state machine is defined that captures the atomicity property and prove correctness of the object implementations (by establishing a simulation mapping between the implementation and the specification automata). Their specification is implemented by three read/write constructions: the message-passing register emulation of Attiya, Bar-Noy and Dolev, its optimized version based on real time, and the shared memory register construction of Vitanyi and Awerbuch. Moreover, a simplified version of their specification is implemented by a general atomic object construction based on Lamport's replicated state machine algorithm.

In *"Time and Space Lower Bounds for Implementations Using k-CAS"*, H. Attiya (Technion, Israel) and D. Hendler (University of Toronto, Canada) present lower bounds on the time- and space-complexity of implementations that use the k compare-and-swap (k-CAS) synchronization primitives. They prove that the use of those primitives cannot improve neither the time- nor the space-complexity of implementations of widely-used concurrent objects, such as counter, stack, queue, and collect (in fact, they may even increase it).

C. Delporte-Gallet (ESIEE-IGM Marne-La-Vallee, France), H. Fauconnier (Université Paris VII, France) and R. Guerraoui (EPFL, Switzerland) show in *"(Almost) All Objects Are Universal in Message Passing Systems"* that all shared atomic object types that can solve consensus among $k > 1$ processes have the same weakest failure detector in a message passing system with process crash failures. In such a system, object types such as test-and-set, fetch-and-add, and queues, which are known to have weak synchronization power in a shared memory system are thus, equivalent to universal types like compare-and-swap, known to have the strongest synchronization power.

Quorum systems have become important tools for providing reliable coordination between processors in distributed systems. In *"The Dynamic And-Or Quorum System"*, U. Nadav and M. Naor (Weizmann Institute, Israel) examine the And-Or quorum system of Naor and Wool both in a static and in a dynamic and scalable environment. Specifically, in the static environment they analyze the algorithmic probe complexity of the And-Or quorum system and present two optimal algorithms: a non-adaptive algorithm with $O(\sqrt{n}\log n)$ probe complexity, and an adaptive algorithm with $O(\sqrt{n})$ probe complexity which requires at most $O(\log\log n)$ rounds (all other known adaptive algorithms require $\theta(\sqrt{n})$ rounds). Furthermore, they present and analyze the *dynamic And-Or* quorum system, a construction for a dynamic and scalable quorum system which can be viewed as a dynamic adaptation of the And-Or system. They show that the dynamic And-Or keeps the optimal load, availability and probe complexity of the And-Or systems, thus becoming an excellent candidate for implementations of dynamic quorums.

3.5 Program composition

A crucial issue in developing distributed applications is the safe composition of smaller programs into larger ones. Such compositions are not easy to obtain, as a simple composition of the programs does not automatically guarantee to maintain their correctness and their intended behavior. The notion of CCL (communication-closed layers), introduced by Elrad and Francez, captures when a program works as if it were executed in isolation in the context of a given larger program, and it is essential for obtaining safe composition.

In the presence of reliable FIFO communication, research has shown that programs can be designed that are inherently CCLs in any program context. K. Engelhardt (University of New South Wales, Australia) and Y. Moses (Technion, Israel) in *"Causing Communication Closure: Safe Program Composition with Non-FIFO Channels"* present a semantic framework for analyzing safe program composition of layers of distributed programs in different models of communication. Essentially, the authors study the impact of message reordering on the design of CCLs. Their communication model assumes that channels neither lose nor duplicate messages but message delivery is not necessarily FIFO (as opposed to prior work). The notion of *sealing* is introduced: if a program P is immediately followed by a program Q that seals P, then P will be communication-closed. The authors provide a formal characterization of sealable programs and develop efficient algorithms for testing whether a program Q seals a program P, and for constructing seals for sealable straight-line programs.

3.6 Transactional memory and replicated systems

A shared data structure is lock-free if its operations do not require mutual exclusion. In highly concurrent systems, lock-free data structures avoid common problems associated with conventional locking techniques, including priority inversion, and difficulty of avoiding deadlocks. *Transactional memory* is a multiprocessor architecture intended to make lock-free synchronization as efficient as conventional techniques based on mutual exclusion. Synchronization is achieved by light-weight in-memory *transactions*, i.e., a finite sequence of reads and writes executed *atomically* by a single thread. A transaction can either *commit* (take effect), or *abort* (have no effect). Transactions are also *serializable*.

In *software transactional memory* (STM) systems, a *contention manager* solves conflicts among transactions accessing the same memory locations. In *"Polymorphic Contention Management"* R. Guerraoui (EPFL, Switzerland), M. Herlihy (Brown University and Microsoft Research Cambridge, USA), and B. Pochon (EPFL, Switzerland) present a *polymorphic* contention manager, a structure that allows contention managers to vary not just across workloads, but

across concurrent transactions in a single workload, and even across different phases of a single transaction. A contention manager is usually evaluated by the number of transactions committed per time-unit. Based on this cost, a hierarchy of contention managers is presented, and also a general algorithm to handle conflict between contention managers from different classes.

Although transactional memory has mostly been studied in the context of multiprocessors, it has also attractive features for distributed systems. M. Herlihy and Y. Sun (Brown University, USA) in *"Distributed Transactional Memory for Metric-Space Networks"* study the implementation of transactional memory in a network of nodes where communication costs form a metric. They develop and analyze a new distributed cache-coherence protocol for tracking and moving up-to-date copies of cached objects. The protocol is evaluated in terms of its *stretch*: each time a node issues a request for a cached copy of an object, the ratio of the protocol's communication cost for that request over the optimal communication cost for that request is computed. In the context of *constant-doubling metrics*, their protocol has stretch logarithmic in the diameter of the network.

An *adaptive* STM system is considered in *"Adaptive Software Transactional Memory"* by V. J. Marathe, W. N. Scherer III, and M. L. Scott (University of Rochester, USA). A detailed analysis of the design space of modern STMs, identifying four key dimensions of STM system design, is presented. Motivated by that analysis, the authors present a new adaptive STM system that adjusts to the offered workload, allowing it to match the performance of the best known existing system on every tested workload.

Data replication in fundamental in distributed systems, and numerous replication methods have developed so far. D. Malkhi (Microsoft Research Silicon Valley, USA and Hebrew University of Jerusalem, Israel) and D. Terry (Microsoft Research Silicon Valley, USA) in *"Concise Version Vectors in WinFS"* present *predecessor vectors with exceptions* (PVEs), a novel optimistic replication technique developed for Microsoft's WinFS system, which shows a substantial reduction in storage and communication overhead associated with replica synchronization in most "normal" cases, in which communication disruptions are infrequent. Their study demonstrates a cut-off threshold in the communication fault-rate, beyond which the PVE technique become less attractive than the traditional schemes.

3.7 Graph theory and distributed computing

In mathematics and computer science, graph theory studies the properties of graphs. Structures that can be represented as graphs are ubiquitous, and numerous problems of practical interest can be represented by graphs. The development of algorithms to handle graphs is therefore of major interest in computer science,

especially in networking and distributed computing. The following papers are concerned with graph aspects used in distributed computing.

The distributed complexity of computing the maximal independent set (MIS) in a graph is a challenging problem in distributed computing of practical and theoretical importance. In *"Fast Deterministic Distributed Maximal Independent Set Computation on Growth-Bounded Graphs"* F. Kuhn, T. Moscibroda, R. Wattenhofer (ETH Zurich, Switzerland), and T. Nieberg (University of Twente, The Netherlands) develop and analyze the first *deterministic* polylogarithmic-time algorithm for the MIS problem in graphs with bounded growth. Specifically the algorithm requires time $O(\log \Delta \cdot \log^* n)$, where n is the number of nodes and Δ the maximal degree in the graph.

When taking a graph or network with a high graph diameter and adding a very small number of edges randomly, the diameter tends to drop drastically. This is known as the *small world* phenomenon. More formally speaking, a graph is said to represent a small world if the resulting oblivious diameter is polylogarithmic in the number of the involved nodes. Kleinberg has formalized and studied the above property into what is called the *small world phenomena*. In what is called the Kleinberg model, a two dimensional square mesh is augmented by the random addition of one directed outgoing arc per node. M. Flammini, L. Moscardelli, A. Navarra (University of L'Aquila, Italy), and S. Perennes (INRIA, France) in *"Asymptotically Optimal Solutions for Small World Graphs"* present the first general lower bound on the expected oblivious diameter holding for any monotone distance distribution. Furthermore, they show that the problem is intractable in the deterministic case and they develop asymptotically optimal constructions for paths, trees and Cartesian products of graphs, including d-dimensional grids for any fixed value of d.

Despite the popularity and importance of the FIFO protocol and the research devoted for its study, deciding whether a given network is stable under FIFO has remained an open question for several years. *Stability* refers to the fact that the amount of packets in the network remains always bounded. M. Blesa (Universitat Politècnica de Catalunya, Spain) in *"Deciding Stability in Packet-Switched FIFO Networks Under the Adversarial Queuing Model in Polynomial Time"* addresses the decidability and complexity of stability under the FIFO protocol) and attempts to characterize the property of stability under FIFO in terms of forbidden network (sub)topologies. The property is shown to be decidable in *polynomial* time.

The model in which a network consists of nodes with arbitrary names is called the *name-independent model*. The *stretch factor* of a routing scheme is defined as the maximum ratio over all pairs between the length of the route induced by the scheme and the length of a shortest-path between the same pair. In *"Compact*

Routing for Graph Excluding a Fixed Minor" I. Abraham (Hebrew University of Jerusalem, Israel), C. Gavoille (University of Bordeaux, France) and D. Malkhi (Hebrew University of Jerusalem, Israel and Microsoft Research Silicon Valley, USA) present a compact name-independent routing scheme for unweighted networks with n nodes excluding a fixed minor; a graph S is a *minor* of a graph G, if S is a subgraph of a graph obtained by a series of edge contractions of G. The authors show that for any fixed minor, the scheme, which can be constructed in polynomial time, has *constant* stretch factor and requires routing tables with *polylogarithmic* number of bits at each node. Furthermore, for shortest-path labeled routing schemes in planar graphs, at least $\Omega(n^{\epsilon})$ space is required to store the routing table on each node, for some constant $\epsilon > 0$.

Applications for labeling schemes concern mainly large and dynamically changing networks. A. Korman (Weizmann Institute, Israel) in *"General Compact Labeling Schemes for Dynamic Trees"* presents a new method for constructing labeling schemes for dynamic trees. As prior work, his method is based on extending existing static labeling schemes to the dynamic setting. However, his resulted dynamic schemes incur a different trade-off between the overhead factors on the label sizes and the message communication. In particular, his trade-off when compared to prior work, gives better performance for the label size on expense of communication.

4 Forthcoming DISC Conferences

The 20th edition of DISC will take place in Stockholm, Sweden. Shlomi Dolev (Ben-Gurion University, Israel) is the appointed Program Committee Chair of DISC 2006. The Organizing Chairs are Seif Haridi (Swedish Institute of Computer Science, Sweden) and Lenka Carr (Lulea Tekniska Universitet, Sweden). In 2007, the 21th edition of DISC is planned to be held in Cyprus. Chryssis Georgiou (University of Cyprus, Cyprus) is the appointed Organizing Chair.

Acknowledgments

First of all, the authors would like to thank Marios Mavronicolas, editor for the Distributed Computing Column, for his kind invitation for writing this review. The authors are also grateful to Adam Iwanicki and Shmuel Zaks for providing some of the pictures, and to Dariusz Kowalski, who did a great job as the organizing chair of DISC 2005 and provided us later with very useful information and material.

Information about Kraków were also obtained from the official websites of the city, at `www.krakow.pl` and `www.krakow-info.com`. Information about the Wierzynek restaurant was obtained from `www.wierzynek.com.pl`.