Annotated Bibliography
on Global States and Times in Distributed Systems

Zhonghua Yang T. Anthony Marsland
Department of Computing Science
University of Alberta
Edmonton, Canada T6G 2H1

A distributed system consists of a collection of concurrently executing processes that do not use shared memory; processes communicate with each other solely through message-passing and coordinate among themselves to achieve a common goal. In such systems, global state descriptions are not readily available, and a common time base does not exist. However, the notion of global states and snapshots lies at the core of many problems in distributed computation, and, as Lamport pointed out in a classical paper, the concept of time is fundamental to our way of thinking. Example applications of the global states and snapshots are distributed algorithms for detecting stable properties which hold during a computation, such as deadlock detection, termination, checkpointing and recovery in distributed databases, and monitoring and debugging of a distributed program. Because of their fundamental nature, a significant body of research results about both global states and global times has been developed in the past few years.

This bibliography identifies these research results and provides brief annotations. It deals with the need to take snapshots of global states in a distributed system, and the need for global time obtained through one of the following two methods: synchronizing local clocks with reference to real time, and building logical clocks and virtual time without reference to physical or real time. Hence, the bibliography is accordingly divided into three parts under the following subtitles: *global states and snapshots, synchronized clocks, and logical clocks and virtual time.* Readers will notice that much of the bibliography is devoted to the topic of synchronized clocks, and this reflects the fact that the theme has attracted great interest among the research community.

While the bibliography is by no means exhaustive, every effort has been made to collect all important works on the topics, based on their *theoretical significance, practical applicability, and multiplicity of perspective*, with the intention of giving a representative picture of each area. We hope that this bibliography and its annotations will be of lasting benefit to interested readers.

Global states and snapshots

[Ahu90] M. Ahuja. Flush Primitives for Asynchronous Distributed Systems. *Information Processing Letters*, 34, 1990. pages 5-12.

This paper introduces three channel primitives for sending messages: *two-way-flush, forward-flush*, and *backward-flush*, collectively termed *Flush*. An *F-channel*, a non-FIFO channel that supports Flush, is defined. The use of these primitives is illustrated by

taking global snapshots in asynchronous distributed systems.

For the author's other related works, see the following two papers:

[Ahu89] M. Ahuja. Repeated Global Snapshots in Asynchronous Distributed Systems. Technical Report OSU-CISRC-8/89-TR40, The Ohio State University, August 1989.

[AKC90] Mohan Ahuja, Ajay D. Kshemkalyani, and Timothy Carlson. A Basic Unit of Computation in Distributed Systems. In *Proceedings of the 10th International Conference on Distributed Computing Systems*, pages 12–19, 1990.

[Bou87] L. Bouge. Repeated Snapshots in Distributed Systems with Synchronous Communication and their Implementation in CSP. *Theoretical Computer Science*, 49, 1987. pages 145-169.

A new version of the snapshot algorithm of Chandy and Lamport is presented which considers synchronous communications and partially ordered semantics, and allows for repeated snapshots. Its implementation in CSP is described.

[CL85] K. Mani Chandy and Leslie Lamport. Distributed Snapshots: Determining Global States of Distributed Systems. *ACM Transactions on Computer Systems*, 3(1), February 1985. pages 63-75.

This is a fundamental paper on distributed snapshots, and presents a general algorithm by which a process in a distributed system determines a global state of the system during a computation.

[CM91] R. Cooper and K. Marzullo. Consistent Detection of Global Predicates. In Barton Miller and Charles McDowell, editors, *Proceedings of the ACM/ONR Workshop on Parallel and Distributed Debugging, published in ACM SIGPLAN Notices Volume 26, Number 12, December 1991*, pages 167–174, Santa Cruz, CA, May 20-21 1991. ACM Press.

Here three algorithms for detecting global predicates in a well-defined way is presented. These algorithms do so by interpreting predicates as being *possibly* true, *definitely* true, or *currently* true, with respect to the communication that has occurred in the system.

[CT90] Carol Critchlow and Kim Taylor. The inhibition spectrum and the achievement of causal consistency. In *Proceedings of the 9th Annual ACM Symposium on Principles of Distributed Computing*, pages 31–42, Quebec city, Canada, 22-24 August 1990. Also as Technical Report TR 90-1101, Cornell University, February 1990.

A spectrum of protocol capabilities based on the type of inhibition that occur is investigated. For the first time, this paper distinguish local vs global inhibitions and proves fundamental relationships between these concepts and determining causally-consistent stattes. A thirty-six-case summary of protocols and impossibility results for the determination of causally-consistent states as a function of those characteristics is then presented.

[Cha82] E. J. H. Chang. Echo Algorithms: Depth Parallel Operations on General Graphs. *IEEE Transactions on Software Engineering*, SE-8(4), 1982. pages 391-401.

The need to obtain global information motivated the creation of algorithms, collectively called *Echo* algorithms. These algorithms are based on the technique of parallel depth-traversal of a general graph modeling the distributed system such that a minimum edge covering is obtained in an asynchronous fashion. Echo algorithms function in parallel, asynchronously, and take execution time for communications of the order of the diameter of the graph modeling the distributed systems.

[Dij85] Edsger W. Dijkstra. The Distributed Snapshot of K. M. Chandy and L. Lamport. In M. Broy, editor, *Control Flow and Data Flow: Concepts of Distributed Programming*, pages 513–517. NATO ASI Series, Vol. F14, Springer-Verlag, Berlin, 1985. Also Technical Report EWD 864a, University of Texas, Austin, Tex., 7 November, 1983.

The distributed snapshot of K. M. Chandy and L. Lamport is illustrated by taking an interleaving view of distributed computation which consists of a succession of so-called "atomic actions". A proof concerning snapshot states is also presented.

[FGL82] Michael J. Fischer, Nancy D. Griffeth, and Nancy A. Lynch. Global States of a Distributed System. *IEEE Transactions on Software Engineering*, SE-8(3), May 1982. pages 198-202.

A global state of a distributed transaction system is considered. In such a system, a global state is *consistent* if no transactions are in progress. A *global checkpoint* is a transaction which must view a globally consistent system state for correct operation. An algorithm for adding global checkpoint transactions to an arbitrary distributed transaction system is presented. The algorithm is nonintrusive in the sense that checkpoint transactions do not interfere with ordinary transactions in progress; however, the checkpoint transactions will still produce meaningful results.

[GGLS91] A. P. Goldberg, A. Gopal, A. Lowry, and R. Strom. Restoring Consistent Global States of Distributed Computations. In Barton Miller and Charles McDowell, editors, *Proceedings of the ACM/ONR Workshop on Parallel and Distributed Debugging, published in ACM SIGPLAN Notices Volume 26, Number 12, December 1991*, pages 144–154, Santa Cruz, CA, May 20-21 1991. ACM Press.

A mechanism for restoring any consistent global state of a distributed computation is presented. This capability can form the basis of support for rollback and replay of computation. The mechanism records occasional state checkpoints and logs all messages communicated between processes.

[Hel89] J. Helary. Observing Global States of Asynchronous Distributed Applications. In J.-C. Bermond and M. Raynal, editors, *Proceedings of the 3rd International Workshop on Distributed Algorithms, LNCS 392*, pages 124–135, 1989.

Notion of global states and some of their properties—consistency, being transitless—are precisely stated, and the problem, in both FIFO and non FIFO communication models, is solved in a progressive way. Solutions generalize and improve known results, both in

FIFO (relaxation of synchronization constraints) and non FIFO (absence of message storing) situations.

[HJPR87] J.-M. Helary, C. Jard, N. Plouzeau, and M. Raynal. Detection of Stable Properties in Distributed Applications. In *Proceedings of the 6th Annual ACM Symposium on Principles of Distributed Computing*, pages 125–136, August 1987.

In this paper, a general algorithm for distributed detection of stable properties in distributed applications or systems is exposed. The algorithm is generic in the sense that it deals with every stable property of a general class.

[LRV87] H. F. Li, T. Radhakrishnan, and K. Venkatesh. Global State Detection in Non-FIFO Networks. In *Proceedings of 7th Conference on Distributed Computing Systems*, pages 364–370, 1987.

Presented here is a robust and efficient algorithm which can determine the consistent global state of a distributed system, even when the communication subsystem does not guarantee ideal behavior. Its performance is also considered.

[LY87] Ten H. Lai and Tao H. Yang. On Distributed Snapshots. *Information Processing Letters*, 25(3), May 1987. pages 153-158.

An efficient snapshot algorithm that needs no control messages and does not require channels to be first-in-first-out is developed. It is also shown that several stable properties (e.g., termination, deadlock) can be detected with uncoordinated distributed snapshots. For such properties, the algorithm can be further simplified.

[Mat92a] Friedemann Mattern. Efficient Algorithms for Distributed Snapshots and Global Virtual Time Approximation. To appear in Journal of Parallel and Distributed Computing, 1993. Technical report SFB 124–11/92, Department of Computer Science, University of Saarland, Germany, Oct. 1992.

The snapshot algorithms presented in this report work without significantly affecting the underlying computation, and do not require channels to be FIFO or messages to be acknowledged. Little storage is needed. The paper proposes new and efficient Global Virtual Time approximation schemes, based on snapshot algorithms and distributed termination detection principles.

[MN91] K. Marzullo and G. Neiger. Detection of Global State Predicates. In *Proceedings of the 5th Workshop on Distributed Algorithms (WDAG-91)*, pages 254-272, Delphi, Greece, October 1991. LNCS 579.

This paper examines algorithms for detecting when a predicate holds during the execution of a distributed system. The predicates considered are expressed over the state of the system and are not assumed to have properties that facilitate detection, such as stability.

[MS91] Keith Marzullo and Laura Sabel. Using Consistent Subcuts for Detecting Stable Properties. In S. Toueg, P.G. Spirakis, and L. Kirousis, editors, *Proceedings of the 5th International Workshop on Distributed Algorithms (WDAG'91)*, pages 273–288. Springer-

Verlag, 1991. LNCS 579.

Presented is a general protocol for detecting whether a property holds in a distributed systems, where the property is a member of subclass of stable properties called the *locally stable properties.*

[Mor85] Carroll Morgan. Global and Logical Time in Distributed Algorithms. *Information Processing Letters*, 20(4), May 1985. pages 189-194.

This paper shows that the design and description of distributed algorithms can be simplified by first assuming that global time is available and can be replaced with Lamport's logical time. Doing so clearly preserves the correctness of the algorithm. The technique is illustrated by examples including Chandy and Lamport's distributed snapshot algorithm.

[Nai88] M. Naimi. Global Stability Detections in the Asynchronous Distributed Computation. In *Proceedings of IEEE Workshop on Future Trends of Distributed Computing Systems in the 1990's*, pages 87–92, Hong Kong, 14-16 September 1988.

A simple algorithm for detecting global stability in asynchronous distributed computations is presented. It is based on monitoring the outgoing channels, using a single token which acts as an observer. This algorithm does not require FIFO communication channels.

[SF86] N. Shavit and N. Francez. A New Approach to Detection of Locally Indicative Stability. In *Proceedings of the 13th International Colloquium on Automata, Language and Programming, LNCS 226*, pages 344–358, Rennes, France, 15–19 July 1986. Springer Verlag.

This paper presents a new approach to the derivation of a detection algorithm for stable properties in distributed systems in cases where local indicators may be found. A local indicator is a local predicate which, when holding, *indicates* that the current local state is a potential component of a stable global state. Thus, a stable global state can be detected by *joining* all these local indicators.

[SK86] Madalene Spezialetti and Phil Kearns. Efficient Distributed Snapshots. In *Proceedings of Sixth International Conference on Distributed Computing Systems*, pages 382–388, 1986.

This paper presents a modification to the distributed snapshot algorithm of Chandy and Lamport. This modification is based upon a general notion of exploiting the work done in the initial phase of a distributed algorithm, to make the following phases simpler or more efficient. The new algorithm offers advantages in terms of the message-passing load imposed upon the system.

[SL85] F. B. Schneider and L. Lamport. Paradigms for Distributed Programs. In M. Paul and H. T. Siegert, editors, *Distributed Systems: Methods and Tools for Specification. Chapter 8*, pages 431–480. Springer-Verlag, LNCS 190, 1985.

In this chapter of the book *Distributed Systems: Methods and Tools for Specification*, some paradigms associated with programming distributed systems are identified and described. The collection includes: Agreement and Commitment, State Machine approach, Distributed Snapshots, and Elections and Wave algorithms. Note that the distributed snapshots exposed are based on a partial ordering view of the computation.

[SL87] S. K. Sarin and N. Lynch. Discarding Obsolete Information in a Replicated Database System. *IEEE Transactions on Software Engineering*, SE-13(1), 87. pages 39-47.

A protocol is described which allows sites in a replicated database system to agree that updates older than a given timestamp are no longer needed and can be discarded. This protocol uses the "distributed snapshot" algorithm of Chandy and Lamport and represents a particular application of that algorithm.

[SMP87] B. Samadi, R. Muntz, and D. Parker. A Distributed Algorithm to Detect a Global State of a Distributed Simulation System. In *Proceedings of the IFIP WG 10.3 Working Conference on Distributed Processing*, Amsterdam, North-Holland, 5-7 October 1987.

This paper describes an algorithm for detecting a global state of a distributed simulation system. Interest in such a system is a simulation state to which all processors have advanced. The algorithm is then generalized to other distributed applications dealing with problems such as termination.

[Tay89] K. Taylor. The Role of Inhibition in Asynchronous Consistent-Cut Protocols. In J.-C. Bermond and M. Raynal, editors, *Proceedings of the 3rd International Workshop on Distributed Algorithms, LNCS 392*, pages 280–291, 1989.

This paper presents results regarding *consistent-cut* protocols. What it means for a protocol to be *non-inhibitory* is formally defined. It is shown that there are no non-inhibitory consistent-cut protocols for non-FIFO asynchronous systems. A lower bound on communication for non-inhibitory consistent-cut protocols for FIFO systems is given, and two protocols are presented.

[Ven89] S. Venkatesan. Message-Optimal Incremental Snapshots. In *IEEE 9th International Conference on Distributed Computing Systems*, pages 53–60. IEEE Computer Society Press, 1989.

Presented here is a message efficient protocol by taking incremental snapshots. It is shown that the protocol uses the minimum number of additional messages possible, and hence it is message-optimal.

[YM92] Z. Yang and T. A. Marsland. Global Snapshots for Distributed Debugging. In *Proceedings of 4th International Conference of Computers and Information (ICCI'92)*, pages 436–440, Toronto, Canada, May 28-30 1992. IEEE Computer Society Press.

This paper examines global snapshot algorithms from a distributed debugging perspective, and proposes an abstract framework based on global snapshots. In this framework, the problem of distributed breakpoints is factored into the following two parts: a simple, single sequential breakpoint set in a process, and a global snapshot algorithm to catch a consistent global state along the sequential breakpoint based on a well-established theory.

Synchronized clocks

[Arv89] K. Arvind. A New Probabilistic Algorithm for Clock Synchronization. In *Proceedings of Real Time Systems Symposium*, pages 330–339, Santa Monica, CA, 5–7 December 1989.

Presented is an averaging probabilistic clock synchronization algorithm that is based on the redundant transmission of multiple synchronization messages. The algorithm guarantees a much lower upper bound on the deviation between clocks than most existing algorithms.

[BD87] O. Babaoglu and R. Drummond. (Almost) No Cost Clock Synchronization. In *Proceedings of 7th International Symposium on Fault-Tolerant Computing*, pages 42–47. IEEE Computer Society Press, July 1987.

This paper shows how synchronized clocks can be realized in a distributed system as a byproduct of a common communication paradigm where processors periodically perform broadcasts. The approach described decouples the *precision* concern and the *accuracy* concern of clock synchronization. Given a system that guarantees only precision, this paper develops a protocol whereby high accuracy can be achieved on demand.

The following is the revised and full version of the previous paper.

[BD93] R. Drummond and O. Babaoglu. Low–Cost Clock Synchronization. To appear in *Distributed Computing.*

[BG81] G. G. Belford and E. Grapa. Setting Clocks "Back" in the Distributed Computing Systems. *Journal of Digital Systems*, 5(1-2), Spring-Summer 1981. Pages 125-135. An early version appeared in the 1st International Conference on Distributed Computing Systems. October 1-5, 1979. Huntsville, AL.

Keeping the clocks synchronized, particularly keeping them synchronized with "real" time is difficult. This paper proposes a way, in effect, to set clocks "back" when they are running too fast, without disrupting transaction order.

[BST86] Micah Beck, T. K. Srikanth, and Sam Toueg. Implementation Issues in Clock Synchronization. In B. Simons and A.Z. Spector, editors, *Proceedings of the Asilomar Workshop on Fault-Tolerant Distributed Computing, Lecture Notes in Computer Science 448*, pages 97–107, Asilomar, California, March 1986. Springer-Verlag.

Some results from an experimental implementation of a recent clock synchronization algorithm are presented. The algorithm was designed to overcome arbitrary processor failures, and to achieve optimal accuracy. That is, the accuracy of synchronized clocks (with respect to real time) is as good as that specified for the underlying hardware clocks.

[CAS86] Flaviu Cristian, Houtan Aghili, and Ray Strong. Clock Synchronization in the Presence of Omission and Performance Faults, and Processor Joins. In *Proceedings of 16th International Symposium on Fault-Tolerant Computing*, pages 218–223, Vienna, Austria, July 1986. IEEE Computer Society Press.

This paper presents a simple, yet practical protocol for synchronizing clocks in a distributed system under a realistic failure assumption. The protocol is tolerant of any number of omission failures (e.g. processor crashes, link crashes, occasional message losses), and performance failures (e.g. overloaded processors, slow links) that do not partition the communications network, and handles any number of simultaneous *processor joins.*

[CF88] R. Cole and C. Foxcroft. An experiment in clock synchronization. *The Computer Journal*, 31(6), 1988. pages 496-502.

An experiment to obtain an upper bound on the accuracy that can be expected when attempting to synchronize the system clock of one computer with the system clock of a physically remote computer is described.

[CFN91] D. Couvet, G. Florin, and S. Natkin. A Statistical Clock Synchronization Algorithm for Anisotropic Networks. In *Proceedings of 10th Symposium on Reliable Distributed Systems*, pages 42–51, Pisa, Italy, 30 September–2 October 1991.

A method is proposed to estimate the value of remote clocks in distributed systems. The method is able to deal with isotropic and anisotropic networks and includes a new way to detect performance failures on single exchanges. It uses a statistical approach to estimate the relative drift of clocks and a round trip clock reading protocol to compute the offset.

[Cri89] F. Cristian. Probabilistic Clock Synchronization. *Distributed Computing*, 3, 1989. pages 146-158. Also appear in *Proc. Ninth IEEE International Conference on Distributed Computer Systems*, June 1989, pages 288-296.

A probabilistic method is proposed for reading remote clocks in distributed systems subject to unbounded random communication delays. The method can achieve clock synchronization precisions superior to those attainable by previously published clock synchronization algorithms.

[CS90] F. Cristian and F. Shmuck. Continuous Clock Amortization Need not Affect the Precision of a Clock Synchronization Algorithm. In *9th Annual ACM Symposium on Principles of Distributed Computing*, pages 133-143. Also as Technical Report RJ7290, IBM, January 1 1990.

A common characteristic of all clock synchronization algorithms is that each processor computes periodically an *adjustment* to its logical *hardware* clock. To avoid problems with *discrete* adjustments, an *amortization* clock adjustment technique is introduced. It is shown that amortization does not worsen the precision achieved by a clock synchronization algorithm, and that it is possible to design a clock synchronization algorithm, and prove it correct independently of the way in which logical clocks are implemented.

[DHHB87] A. Duda, G. Harrus, Y. Haddad, and G. Bernard. Estimating Global Time in Distributed Systems. In *Proceedings of the 7th International Conference on Distributed*

Computing Systems, pages 299–306, Berlin, 21-25 September 1987. IEEE CS Press.

A method is presented to estimate the global time in a loosely coupled distributed system from local traces recorded using unsynchronized local clocks. The method treats pairs of message send and receive instances as realizations of two random variables having a functional linear dependence. A least-square regression analysis is used to estimate both the time offset and the time offset rate between two local clocks. Also presented is another geometric estimation method that finds a convex hull for a given set of points.

[DHM73] W. M. Daly, A. L. Hopkins, Jr., and J. F. Mckenna. A Fault-Tolerant Clocking System. In *FTCS 3, Digest of papers. The 3rd Symposium on Fault-Tolerant Computing Systems*, pages 17–21, Palo Alto, CA., 20-22 June 1973. IEEE 1973.

A clocking system is described that uses an array of identical oscillator modules to produce a number of phase-locked clock signals. Using $3f + 1$ modules, phase-locking is maintained on at least $2f + 1$ of the signals after f failures. The system can provide reliable synchronization for systems that use comparison or voting to achieve fault tolerance.

[DHS86] D. Dolev, J. Y. Halpern, and R. Strong. On the Possibility and Impossibility of Achieving Clock Synchronization. *Journal of Computer and Systems Science*, 32(2), April 1986. pages 230-250.

It is known that clock synchronization can be achieved in the presence of faulty processors as long as the nonfaulty processors are connected, provided some authentication technique is used. This paper shows that if logical clocks are restricted to running within some linear functions of real time, then clock synchronization is impossible without authentication when one-third or more of the processors are faulty. A lower bound on the accuracy is also presented.

[DHSS92] Danny Dolev, Joseph Halpern, Barbara Simons, and Ray Strong. Dynamic Fault-Tolerant Clock Synchronization. Technical Report RJ 8576 (77355), IBM Research Division, Almaden Research Center, January 16 1992. The early version titled "Fault-tolerant clock synchronization," which appeared in *Proceedings of the 3rd ACM Symposium on Principles of Distributed Computing*, pages 89–102, 1984.

This paper gives two simple efficient distributed algorithms: one for keeping clocks in a network synchronized and one for allowing new processors to join the network with their clocks synchronized. The algorithm for maintaining synchronization works for arbitrary networks and tolerates any number of processor or communication link faults, as long as the correct processors remain connected by fault-free paths. It thus represents an improvement over other clock synchronization algorithms. The algorithm for allowing new processors to join requires that more than half the processors be correct, a requirement that is provably necessary.

[EK73] C. Ellingson and R. J. Kulpinski. Dissemination of System-Time. *IEEE Transactions on Communications*, COM-23(5), May 1973. pages 605-624.

Here the problem of estimating the offset in timing of like events at geographically separated locations is considered. It forms a basis for establishing common knowledge of time and, hence, system synchronization.

[GHT88] W. Gora, U. Herzog, and S. K. Tripathi. Clock Synchronization on the Factory Floor (FMS). *IEEE Transactions on Industrial Electronics*, 35(3), August 1988. 372-380.

This paper describes the synchronization requirements on the factory floor and discusses several clock synchronization algorithms. It provides theoretical bounds on the clock synchronization, and gives some implementation results.

[GZ84] R. Gusella and S. Zatti. TEMPO—A Network Time Controller for a Distributed Berkeley UNIX System. *IEEE Distributed Processing Technical Committee Newsletter*, NoSI-2(6), June 1984. pages 7-15. Also in *Proc. Summer 1984 USENIX*, pages 78-85, Salt Lake City, UT, June 1984.

TEMPO keeps the clocks of computers in a local network synchronized with an accuracy comparable to the resolution of each individual clock. A new algorithm has been devised to perform the measurement; a protocol presented can adjust the clocks by means of a new system call that has been added to the kernel of the Berkely UNIX 4.2BSD operating system.

[HMM85] J. Halpern, N. Megiddo, and A. Munshi. Optimal precision in the presence of uncertainty. *Journal of Complexity*, 1(2), December 1985. pages 170-196.

The problem of achieving coordinated actions in a real time distributed system is considered. The paper considers how closely (in terms of real time) processors can be guaranteed to perform a particular action, in a system where message transmission is guaranteed, but where there is some uncertainty in message transmission time. An algorithm to achieve optimal precision in arbitrary networks is presented.

[HS91] J. Y. Halpern and I. Suzuki. Clock synchronization and the power of broadcasting. *Distributed Computing*, 5(2), September 1991. pages 73-82.

This paper investigates the power of a broadcast mechanism in a distributed network by considering the problem of synchronizing clocks in an error-free network, under assumption that there is no upper bound on message transmission time, but that broadcast messages are guaranteed to be received within a small constant time interval.

[Jez89] Jean-Marc Jezequel. Building a Global Time on Parallel Machines. In *Proceedings of the 3rd International Workshop on Distributed Algorithms*, pages 136–147, 1989. LNCS 392.

This paper presents a pragmatic algorithm to represent global time on any distributed system. The paper also surveys and criticizes the classical approaches based on clock synchronization techniques. A statistical method is presented as a building block to derive an original algorithm valid for any topology.

[JB84] S. C. Johnson and R. W. Butler. A Validation Methodology for Fault-Tolerant Clock Synchronization. In *Proceedings of the AIAA/IEEE 6th Digital Avionics Systems Con-*

ference, pages 225–232, Baltimore, MD, USA, 3-6, December 1984. NY, AIAA, 1985.

A validation method for the synchronization subsystem of a fault-tolerant computer system is presented. The method presented uses a formal design proof to uncover design and coding errors, and experimentation to validate the assumptions of the design proof. The experimental method is described and illustrated. The design proof of the algorithm defines the maximum skew between any two nonfaulty clocks in the system, in terms of theoretical upper bounds on certain system parameters.

[Kes84] J. L. Kessels. Two Design of a Fault-Tolerant Clocking System. *IEEE Transactions on Computers*, C-33(10), October 1984. pages 912-919.

Two designs of a fault-tolerant clocking system are described: a time-discrete design based on autonomous oscillators, and a time-continuous design based on voltage-controlled oscillators. The system is fault-tolerant in that if at most f modules (out of at least $2f+1$ in total) fail, the remaining modules continue to produce synchronous clock signals.

[KO87] H. Kopetz and W. Ochsenreiter. Clock Synchronization in Distributed Real-Time Systems. *IEEE Trans. Computers*, C-36(8), August 1987. pages 933-939.

Depending on the types and number of tolerated faults, this paper presents upper bounds on the achievable synchronization accuracy for external and internal synchronization in a distributed real time system. The concept of continuous versus instantaneous synchronization is introduced in order to generate a uniform common-time base for local, global, and external time measurements. The functions of a VLSI clock synchronization unit, which improves the synchronization accuracy and reduces the CPU load, are also described.

For the author's other related work, see also:

[Kop86] H. Kopetz. Accuracy of Time Measurement in Distributed Real Time Systems. In *Proceedings of 5th Symposium on Reliability in Distributed Software and Database Systems*, pages 35–41, Los angles, USA, 13-15, January, 1986 1986. IEEE Computer Society Press.

[Kri90] C. M. Krishna. Fault-Tolerant Synchronization Using Phase-Locked Clocks. *Microelectronics and Reliability (UK)*, 30(2), 1990. 275-287.

Over the past few years, phase-locked clocks have emerged as the leading contender for hardware-based, fault-tolerant synchronization. This paper presents a brief survey of work on this synchronization technique.

[KSB84] C. M. Krishna, K. G. Shin, and R. G. Butler. Synchronization and Fault-Masking in Redundant Real-Time Systems. In *The 14th Int'l Symposium on FTCS*, pages 152–157, Kissimmee, FL, 20–22 June 1984.

The overhead associated with synchronization in software is estimated. A theorem is proved that allows the indefinite expansion of phase-locked clocks that maintain synchrony with negligible overhead. The impact of the fault and locations overhead on reliability is assessed.

[KSB85] C. M. Krishna, K. G. Shin, and R. G. Butler. Ensuring Fault Tolerance of Phase-Locked Clocks. *IEEE Transactions on Computers*, C34(8), August 1985. pages 752-756.

This paper shows how to construct phase-locked clocks that operate correctly in the face of a given number of malicious failures.

[Lis91] Barbara Liskov. Practical Uses of Synchronized Clocks in Distributed Systems. In *Proceedings of the Tenth Annual ACM Symposium on Principles of Distributed Computing*, pages 1–9, August 1991.

Although uses of synchronized clocks in distributed algorithms have received relatively little attention, this paper discusses a number of distributed algorithms that make use of synchronized clocks and analyzes how clocks are used in these algorithms.

[LL84] J. Lundelius and N. Lynch. An Upper and Lower Bound for Clock Synchronization. *Information and Control*, 62(2-3), August-September 1984. pages 190-204.

The problem of synchronizing clocks of processes in a fully connected network is considered. It is proved that, even if the clocks all run at the same rate as real time and there are no failures, an uncertainty of ε in the message delivery time makes it impossible to synchronize the clocks of n processes any more closely than $\varepsilon(1 - 1/n)$. A simple algorithm is given that achieves this bound.

[LL88] J. Lundelius and N. Lynch. A New Fault-Tolerant Algorithm for Clock Synchronization. *Information and Computation*, 77(1), April 1988. pages 1-36. Also in *Proceedings of the 3rd ACM Symposium on Principles of Distributed Computing*, pages 75-88, August, 1984.

A new fault-tolerant algorithm for solving a variant of Lamport's clock synchronization problem is described. The algorithm solves the problem of maintaining closely synchronized local times, assuming that processes' local times are closely synchronized initially. The algorithm is able to tolerate the failure of just under a third of the participating processes. Reintegration of a repaired process can be accomplished through a slight modification to the basic algorithm. A similar style of the algorithm can also be used to achieve synchronization initially.

[LMS85] L. Lamport and P. M. Melliar-Smith. Synchronizing Clocks in the Presence of Faults. *Journal of the ACM*, 32(1), January 1985. Pages 52-78.

Algorithms are described for maintaining clock synchrony in a distributed multiprocess system where each process has its own clock. These algorithms work in the presence of arbitrary clock or process failures, including "two-faced clocks" that present different values to different processes. Two of the algorithms require that fewer than one-third of the processes be faulty. A third algorithm works if fewer than half the processes are faulty, but requires digital signatures.

For the author's other related works, see the following two papers:

[Lam87] Leslie Lamport. Synchronizing Time Servers. Technical Report 18, Digital System Research Center, CA. June 1 1987.

[LMS86] L. Lamport and P. M. Melliar-Smith. Byzantine Clock Synchronization. *Operating System Review*, 20(3), July 1986. pages 10-16. Also in *Proceedings of the 3rd ACM Symposium on Principles of Distributed Computing*, 1983.

[LSW91] B. Liskov, L. Shrira, and J. Wroclawski. Efficient At-Most-Once Messages Based on Synchronized Clocks. *ACM Transactions on Computer Systems*, 9(2), May 1991. pages 125-142.

This paper describes a new at-most-once message passing protocol, *synchronized clock message protocol (SCMP)*, that is based on synchronized clocks. The method described depends on the bound for performance but not for correctness. It is the first message-passing protocol to make use of a synchronized clock to provide at-most-once delivery.

[LZM89] M. Lu, D. Zhang, and T. Murita. A Design Approach for Self-Diagnosis of Fault-Tolerant Clock Synchronization. *IEEE Transaction on Computers*, 38(9), September 1989. 1337-1341.

A general design approach for self-diagnosis of faulty clocking modules is presented. It is based on a statistical testing method. With the support of a necessary repair technique, integration of the self-diagnosis algorithm into a mask-based FTCS algorithm helps achieve an ideal level of clock availability. A self-stability evaluation method is also discussed.

[Mil91] D. L. Mills. Internet Time Synchronization: The Network Time Protocol. *IEEE Transactions on Communications*, 39(10), October 1991. pages 1482-1493.

This is one of the important papers about clock synchronization. It describe a well-known Internet Standard Recommended Protocol—the network time protocol (NTP). The paper also discusses the architecture, protocol and algorithms, which were developed over several years of implementation refinement. The performance data of the protocol are also presented.

For the complete description of the NTP protocol and related work, see the following two documents:

[Mil89] D. L. Mills. Network Time Protocol (Version 2) Specification and Implementation. DARPA internal report RFC 1119, DARPA, September 1989.

[Mil90] D. L. Mills. On the Accuracy and Stability of Clocks Synchronization by the Network Time Protocol in the Internet System. *Computer Communication Review*, 20(1), January 1990. pages 65-75.

[MO85] Keith Marzullo and Susan Owicki. Maintaining the Time in a Distributed System. *ACM Operating System Review*, 19(3), July 1985. pages 44-54. Also in the *Proc. 2nd ACM Symposium on Principles of Distributed Computing*, pages 295-305, Canada, August, 1983.

This paper describes algorithms for obtaining a clock value from a set of clocks, some of which may be faulty, and for providing an accurate time and date.

[MS85] S. R. Mahaney and F. B. Schneider. Inexact Agreement: Accuracy, Precision, and Graceful Degradation. In *Proceedings of the 4th ACM Symposium on the Principles of Distributed Computing*, pages 237–249, Minaki, Canada, August 1985.

Two protocols to achieve *Inexact Agreement* are presented. They give better precision, but possibly worse accuracy, than previous work, and exhibit graceful degradation — reasonable and predictable behavior — when as many as 2/3 of the processors are faulty. It is the first work to address this issue, and shows how these protocols can be used to synchronize clocks in a distributed system.

[Ofe89] Yoram Ofek. Generating a Fault Tolerant Global Clock in High Speed Distributed System. In *IEEE 9th International Conference on Distributed Computing Systems*, pages 218–226. IEEE Computer Society Press, 1989. A revised version of this paper will appear in *IEEE Transactions on Communications*, 1992.

This work describes a technique for constructing a fault tolerant global clock in a point-to-point distributed system with an arbitrary topology constituting a wide area network. It is assumed that the network is constructed of optical links with very high transmission rates. Therefore, the approach taken in this work is to generate a global clock from the ensemble of local clocks, and not to directly synchronize them. The synchronization algorithm tolerates any physical failure.

[OS91] Alan Olson and Kang G. Shin. Probabilistic clock synchronization in large distributed systems. In *Proceedings of 1991 IEEE 11th International Conference on Distributed Computing Systems*, pages 290–297, Arlington, Texas, May 20-24 1991.

This paper proposes a synchronization algorithm which uses one of two probabilistic techniques to estimate remote clock values, and uses an interactive convergence algorithm on the resulting estimates to adjust the local clock. The algorithm does not require master/slave clocks and yet requires fewer messages. As a result it is suitable for use in large distributed systems.

[Ric88] N. W. Rickert. Non Byzantine Clock Synchronization—a Programming Experiment. *ACM Operating Systems Review*, 22(1), January 1988. pages 73-78.

Here the clock synchronization problem is used as a basis for programming experiments to investigate environmental adaptation programming. Several algorithms based on information about the clocks of the immediate neighbors for adjusting each node's clock are experimented with.

[RKS90] P. Ramanathan, D. D. Kandlur, and K. G. Shin. Hardware-Asisted Software Clock Synchronization for Homogeneous Distributed Systems. *IEEE Transactions on Computers*, 39(4), April 1990. pages 514-524.

A synchronization scheme that strikes a balance between the hardware requirement and the clock skews is attainable. The proposed scheme is a software algorithm that uses minimal additional hardware to achieve reasonably tight synchronization. Unlike other software solutions, the guaranteed worst case skews can be made insensitive to the maximum variation of message transit delay in the system.

[RS88] P. Ramanathan and K. G. Shin. Checkpointing and Rollback Recovery Using Common Time Base. In *Proceedings of 7th IEEE Symposium on Reliable Distributed Systems*, pages 13–21, October 1988.

An approach to checkpointing and rollback recovery in a distributed computing system using a common time base is presented. First, a common time base is established in the system using a hardware clock synchronization algorithm. This common time base is coupled with a pseudo-recovery block approach to develop a checkpointing algorithm.

[RSB90] P. Ramanathan, Kang G. Shin, and R. W. Butler. Fault–Tolerant Clock Synchronization in Distributed Systems. *IEEE Computer*, 23(10), October 1990. 33-42.

This article compares and contrasts existing fault-tolerant clock synchronization algorithms. The worst-case clock skews guaranteed by representative algorithms are compared, along with other important aspects such as time, message, and cost overhead imposed by the algorithms. Special emphasis is given to more recent developments such as hardware-assisted software synchronization, probabilistic clock synchronization, and algorithms for synchronizing large, partially connected distributed systems.

[Sch87] F. B. Schneider. Understanding Protocols for Byzantine Clock Synchronization. Technical Report 87-859, Department of Computer Science, Cornell University, August 1987.

All published fault-tolerant clock synchronization protocols are shown to result from refining a single paradigm. This allows the different clock synchronization protocols to be compared and permits presentation of a single correcting analysis that holds for all. The paradigm is based on a reliable time source that periodically causes events: detection of such an event causes a processor to reset its clock.

For the author's other related work, see the following paper:

[Sch86] F. B. Schneider. A Paradigm for Reliable Clock Synchronization. In *Proc. Advanced Seminar on Real Time Local Area Networks*, Bandol, France, April 1986, pages 85-104. Also as Technical Report 86-735, Department of Computer Science, Cornell University, February 1986.

[Smi81] T. B. Smith. Fault-Tolerant Clocking System. In *FTCS 11, Digest of papers. The 11th Symposium on Fault-Tolerant Computing Systems*, pages 262–264. IEEE Computer Society Press, 1981.

A fault-tolerant clocking system was developed that provides the necessary identical and phase aligned clocking signals to each of the replicated processing elements.

[SR87] K. G. Shin and P. Ramanathan. Clock Synchronization of a Large Multiprocessor System in the Presence of Malicious Faults. *IEEE Transactions on Computers*, C-36(1), January 1987. pages 2-12.

In this paper, a new method is proposed that i) requires little time overhead by using phase-locked clock synchronization, ii) needs a clock network very similar to the processor network, and iii) uses only 20-30 percent of the total number of interconnections required by a fully connected network for almost no loss in the synchronizing capabilities. Both ii) and iii) are made possible by grouping the various clocks in the system into many different clusters and then treating them as single clock units as long as the network is connected. An example hardware implementation is presented.

For the authors' other related works, see the following two papers:

[SR85] K. G. Shin and R. Ramanathan. Synchronization of a Large Clock Network in the presence of Malicious Faults. In *Proceedings of Real Time System Symposium*, pages 13–24, San Diego, CA, USA, 3-6, December 1985. IEEE Computer Society Press.

[SR88] K. G. Shin and P. Ramanathan. Transmission Delays in Hardware Clock Synchronization. *IEEE Transactions on Computers*, C-37(11), November 1988. pages 1465-1467.

[SS88] K. Sugihara and I. Suzuki. Nearly Optimal Clock Synchronization under Unbounded Message Transmission Time. In *Proceedings of 1988 International Conference on Parallel Processing*, pages 14–17, 1988.

This paper presents the first known solution for clock synchronization under unbounded message transmission time. It assumes a fully connected network of $n \geq 3$ processors in which a process can send messages to a set of other processes simultaneously at time t, and these messages are guaranteed to be delivered in the time interval $[t+\delta, t+\delta+\epsilon]$ for some δ and ϵ, where ϵ is a constant but δ can vary and no upper bound on δ is known. It is shown that, under this assumption, the clocks of the n processes cannot be synchronized any more closely than $(1+\frac{1}{n(n-2)})\epsilon$, even if the clocks run at the rate of real time. A simple algorithm that synchronizes the clocks to within $(1+\frac{1}{n})\epsilon$ is presented.

[ST87] T. K. Srikanth and Sam Toueg. Optimal Clock Synchronization. *Journal of the ACM*, 34(3), July 1987. pages 626-645.

This paper presents a simple, efficient, and unified solution to the problems of synchronizing, initializing, and integrating clocks for systems with the following failure types: crash, omission, and arbitrary failures with and without message authentication. This is the first known solution that achieves optimal accuracy. The solution is also optimal with respect to the number of faulty processes that can be tolerated to achieve this accuracy.

[SWL90] B. Simons, J. Welch, and N. Lynch. An Overview of Clock Synchronization. In B. Simons and A. Spector, editors, *Fault-Tolerant Distributed Computing, Lecture Notes in Computer Science 448*, pages 84–96. Springer-Verlag, Berlin Heidellberg, 1990. Also Research Report RJ 6505, IBM Almaden Research Center, 1988.

This paper provides an overview on some of the theoretical results involving clock synchronization. Descriptions of several algorithms to synchronize clocks in partially synchronous, unreliable models are given. Some problems closely related to the clock synchronization problem are discussed, some open problems are mentioned in closing.

[TFKW89] P. Thambidurai, A. M. Finn, R. M. Kieckhafer, and C. J. Walter. Clock Synchronization in MAFT. In *FTCS 19, Digest of papers. The 19th Symposium on Fault-Tolerant Computing*, pages 142–149, Chicago, Illinois, 21-23 June 1989. IEEE Computer Society Press.

This paper discusses the "steady-state" clock synchronization algorithm of MAFT, an extremely reliable system for real time applications. The synchronization algorithm has been implemented in hardware, and a system prototype has been constructed. The algorithm employs an Interactive Convergence approach. The problem of detecting clock faults is also addressed.

[VM88] N. Vasanthavada and P. N. Marinos. Synchronization of Fault-Tolerant Clocks in the Presence of Malicious Failures. *IEEE Transactions on Computers*, C-37(4), April 1988. pages 440-448.

This paper addresses the problem of achieving global clock synchronization in fault-tolerant clocks by preventing "multiple cliques" in the presence of malicious clock failures. A new solution to the problem, referred to as *the averaging rule*, is developed and its use is analytically justified using the notion of clock partitions and generalized clock partitions. The efficacy of the averaging rule was established not only by analysis but also by means of simulations and experimentation with hardware clock implementations.

The following three papers are the authors' other related works:

[VMM86a] N. Vasanthavada, Peter N. Marinois, and Gerald S. Mersten. Design and Performance Evaluation of Mutual Synchronized Fault-Tolerant Clock Systems. In *FTCS 16, Digest of papers. The 16th Symposium on Fault-Tolerant Computing*, pages 206–211. IEEE Computer Society Press, 1986.

[VMM86b] N. Vasanthavada, P. N. Marinos, and G. S. Mersten. A Study of Fault-Tolerant Clock Systems in the Presence of Malicious Multiple Module Faults. In *Proceedings of IEEE International Conference on Computer Design (ICCD'86)*, pages 516–521, Port Chester, NY, USA, 6-9 October, 1986 1986. IEEE Computer Society Press.

[VTM89] N. Vasanthavada, P. Thambidurai, and P. N. Marinois. Design of fault-tolerant clocks with realistic failure assumption. In *FTCS 19, Digest of papers. The 19th Symposium on Fault-Tolerant Computing*, pages 128–133, Chicago, Illinois, 21-23 June 1989. IEEE Computer Society Press.

[Wag88] K. D. Wagner. Clock System Design. *IEEE Design and Test of Computers*, 5(5), October 1988. 9-27.

This tutorial provides a framework for understanding system timing and then describes how the clock system executes the time specifications. It examines clock generations and the construction of clock-distribution networks, which are integral to any clock system. Examples from contemporary high-speed systems highlight several common methods of clock generation, distribution, and tuning.

Logical clocks and virtual time

[ABDM91] Uri Abraham, Shai Ben-David, and Shlomo Moran. On the Limitation of the Global Time Assumption in Distributed Systems. In S. Toueg, P. G. Spirakis, and L. Kirousis, editors, *5th International Workshop on Distributed Algorithms (WDAG'91)*, pages 1–8, Delphi, Greece, October 1991. Springer-Verlag. LNCS 579.

An ongoing debate among theoreticians of distributed systems concerns the global time issue. The basic question seems to be to what extent does a model with global time reflect the 'real' behavior of a distributed system. This paper demonstrates, for the first time, a protocol that is guaranteed to perform well as long as it is run in a system that enjoys the existence of global time, yet it may fail in some other conceivable circumstances.

[ACGS91] Mohan Ahuja, Timothy Carlson, Ashwani Gahlot, and Deborah Shands. Timestamping Events for Inferring "Affects" Relation and Potential Causality. In *Proceedings of the IEEE 15th Annual International Computer Software and Applications Conference (COMPSAC)*, pages 606–611, 1991.

A relation "Affects" between every pair of events based on a relation "Logically Affects" for each process between each pair of events on the process is defined. The mechanism for timestamping events such that "Affects" relation, and so potential concurrency, between events can be inferred from their timestamps is presented.

The following report is the author's related work:

[ACG92] M. Ahuja, T. Carlson, and A. Gahlot. Passive-space and Time View of Computing. Technical Report OSU-CISRC-1/92-TR-1, Computer and Information Science Research Center, January 7 1992.

[AR85] K. R. Apt and J.-L. Richier. Real Time Clocks versus Virtual Clocks. In M. Broy, editor, *Control Flow and Data Flow: Concepts of Distributed Programming*, pages 513–517. NATO ASI Series, Vol. F14, Springer-Verlag, Berlin, 1985.

Symmetric distributed termination algorithms are systematically developed by using Lamport's logical clocks in place of perfectly synchronized clocks. Solutions are first presented in an abstract setting of Dijkstra, Feijen and Van Gasteren and then gradually transformed into solutions to the distributed termination problem of Francez.

[CB91] Bernardette Charron-Bost. Concerning the Size of Logical Clocks in Distributed Systems. *Information Processing Letters*, 39(7), July 1991. pages 11-16.

It was already known that it is sufficient to use n-tuples as timestamps of events for a system distributed over n processes, if causal independence is to be characterized. This paper proves that vectors of this length n are necessary to characterize causality, by constructing an appropriate distributed computation, and smaller clocks do not work if only the number of processes is known.

[Fid88] C. J. Fidge. Timestamps in Message-Passing Systems that Preserve Partial Ordering. In *Proceedings of 11th Australian Computer Science Conference*, pages 56–66, February 1988.

This paper presents algorithms for timestamping events in both synchronous and asynchronous message-passing programs that allows for access to the partial ordering inherent in a distributed system. The vector-time is, for the first time, proposed as the timestamping mechanism that preserves the partial ordering.

[Fid91] Colin Fidge. Logical Time in Distributed Computing Systems. *IEEE Computer*, 24(8), August 1991. 28-33.

Unlike conventional sequential programs, the computations performed by distributed computing systems do not yield a linear sequence of events. This article defines the partial ordering of events, describes its generalized and practical implementations in terms of partially ordered logical clocks, and summarizes some current applications of the new technology.

[Jef85] David R. Jefferson. Virtual Time. *ACM Transaction on Programming Languages and Systems*, 7(3), July 1985. Pages 404-425.

Virtual time is a new paradigm for organizing and synchronizing distributed systems. It provides a flexible abstraction of real time in much the same way that virtual memory provides an abstraction of real memory. In this paper, a *Time Warp mechanism* for implementing virtual time is described. The mechanism is a synchronization protocol distinguished by its reliance on lookhead-rollback, and its implementation of rollback via antimessages.

[Lam78] Leslie Lamport. Time, Clocks, and the Ordering of Events in a Distributed System. *Communications of the ACM*, 21(7), July 1978. pages 558-565.

This is a clasical paper on the topic. The concept of one event happening before another in a distributed system is examined, and is shown to define a partial ordering of the events. A distributed algorithm is given for synchronizing a system of logical clocks which can be used to totally order the events. The use of the total ordering is illustrated with a method for solving synchronization problems. The algorithm is then specialized for synchronizing physical clocks, and a bound is derived on how far out of synchrony the clocks can become.

[MSV91] S. Meldal, S. Sankar, and J. Vera. Exploiting locality in maintaining potential causality. Technical Report CSL-TR-91-446, Stanford University, April 1991. Also in *Proceedings of the 10th Annual ACM Symposium on Principles of Distributed Computing.*

This paper examines the algorithm proposed by Fidge and Mattern to determine the temporal relationships, and argues that by cutting down on its generality, and assuming that communication links between processes *is static and known ahead of time*, and also that one is interested only in determining the temporal ordering between messages arriving *at the same process*, space requirement can be significantly reduced.

[Mat89] Friedemann Mattern. Virtual Time and Global States of Distributed Systems. In M. Cosnard and P. Quinton, editors, *Proceedings of International Workshop on Parallel and Distributed Algorithms (Chateau de Bonas, France, October 1988)*, pages 215–226, Amsterdam, 1989. Elsevier Science Publishers B. V.

This paper provides a good survey about the notion of time. It is argued that a linearly ordered structure of time is not (always) adequate for distributed systems, and proposes a generalized non-standard model of time which consists of vectors of clocks. These clock-vectors are partially ordered and form a lattice. By using timestamps and a simple clock update mechanism the structure of causality is represented in an isomorphic way. The paper also presents a new algorithm to compute a consistent global snapshot of a distributed system where messages may be received out of order.

[Mat92b] Friedemann Mattern. On the Relativistic Structure of Logical Time in Distributed Systems. Technical report, Department of Computer Science, University of Saarland, Germany, 1992.

This report defines time to be a partially ordered set of vectors forming a lattice structure, and examines the notion of logical time which represents the causality structure of events in an isomorphic way. With this definition, the relations "later" and "at the same time" get a new and generalized interpretation. The motivation of the concept of "vector time" is presented, its properties are elaborated. Its analogy to Minkowski's relativistic space-time is also discussed.

[NT87] G. Neiger and S. Toueg. Substituting for Real Time and Common Knowledge in Distributed Systems. In *Proceedings of 6th ACM Symposium on Principles of Distributed Computing*, pages 281–293, 1987.

Both time and knowledge in reliable distributed systems with asynchronous communication are investigated. The logical clocks and broadcast primitives proposed in this paper are tools that considerably simplify the design of distributed algorithms: one can now design and prove them correct with the assumption that real time clocks are available and common knowledge is achievable.

[Ray87] Michel Raynal. A Distributed Algorithm to Prevent Mutual Drift Between n Logical Clocks. *Information Processing Letters*, 24, February 1987. pages 199-202.

An *a priori* technique to prevent mutual drift between n logical clocks is examined. This paper generalizes the distributed algorithm proposed by Carvalho and Roucairol from 2 to n processes, and proves the correctness of the Carvalho and Roucairol algorithm. The behavior of the generalized algorithm in various environments is studied.

[SK92] Mukesh Singhal and Ajay Kshemkalyani. An Efficient Implementation of Vector Clocks. *Information Processing Letters*, 31(1), August 1992.

In this paper, an efficient technique to maintain Mattern-Fidge's vector clocks is presented. The technique is based on the following observation: only those elements of vector time that have changed since the last communication between two processes need to be transmitted.

[SM91] R. Schwarz and F. Mattern. Detecting Causal Relationships in Distributed Computations: In Search of the Holy Grail. Technical Report SFB 124-15/92, (Department of Computer Science, University of Kaiserslautern and University of Saarland), December 1992.

The notion of causality and its relation to logical time is introduced; some fundamental results concerned are presented. Recent work on the detection of causal relationships in distributed computing is surveyed. The relative merits and limitations of the different approaches are discussed, and their general feasibility is analyzed.