Panel on Top 10 Most Influential Parallel and Distributed Processing Concepts in the Last Millennium

Howard Jay Siegel
School of Electrical and Computer Engineering
Purdue University
West Lafayette, IN 47907-1285, USA
Email: hj@purdue.edu

The following panel has been assembled to discuss "Top 10 Most Influential Parallel and Distributed Processing Concepts in the Last Millennium."

Panel Organizer and Chair:
H. J. Siegel, Purdue University

Panelists:
Mani Chandy – Caltech
Ken Kennedy – Rice University
Tom Leighton – MIT
Jane Liu – University of Illinois
Kang Shin – University of Michigan
Marc Snir – IBM/Yorktown
Larry Snyder – University of Washington
Thomas Sterling – JPL

Panelists will be asked to present their "top 10 lists" for the most influential parallel and distributed processing concepts in the last millennium. They can present less than 10, and the ordering of the list can be arbitrary.

The panelists were chosen to represent a broad range of technical areas. The position statements included here illustrate the variety of perspectives held by the panelists.

After the panelists have given their lists at the panel, there will be an open discussion among the audience and panelists. At the end of the discussion, a ballot will be distributed for the audience to vote on the top 10 (in arbitrary order). The results of the poll will be announced at the conference on the day after the panel.

Each panelist was asked to provide a one-paragraph position statement and a one-paragraph biography for the proceedings. These are below, as submitted. The panelists were told that position statements below would be considered as first drafts, and that they could revise them as much as they wanted for the actual panel.

Mani Chandy

Position Statement:

I would like to separate two aspects of concurrency:

1. The use of concurrency for speed.
2. Systems that are inherently concurrent.

The phrase parallel computing is often associated with the former kind of concurrency, and the phrase distributed computing with the latter kind. The problem in parallel computing is: Given a computational task, select an algorithm, programming language and machine to execute the task with adequate speed. Designers of such systems may prefer to use sequential programs provided they execute with adequate speed. Distributed computing deals with collaboration among collections of people, information appliances and objects. Designers of these infrastructures cannot choose to eliminate concurrency from consideration because concurrency is part of the specification.

Most of the concepts that I consider to be important are in distributed computing because, in my view, inherent concurrency will play a much more critical role in the next millennium than discretionary concurrency will.

Also, in retrospect, most of these ideas are trivial. But then, in retrospect, the wheel is trivial too. And, just as for the wheel, the most influential ideas are so interwoven into the fabric of our lives that we've forgotten that they were once ideas.

Influential concepts:

1. IP. The protocol helped give rise to the Internet, and the Internet truly does change everything.

2. Encryption and authentication algorithms. Factoring is still (apparently) difficult. Thank God! Clever algorithms for encryption have enabled widespread distributed computing.
3. URLs. Uniform naming schemes, coupled with browsers and other tools, will change the world.
4. XML. A trivial idea, but revolutionary as more of us agree to use common document structures. Likewise, HTML is also important, and the separation of content (XML) from style (XSL) is even more so.
5. Scalable database servers have clever multithreaded algorithms, and these servers form the basis for data exchange and e-commerce that have had a profound impact. (Yes, some of that hype is real.)
6. Java. Ideas about abstract data types are at least 25 years old. The concept of a portable VM is not new. The concept of class loaders that can download classes from anywhere and instantiate objects on any machine is not a difficult concept. Ideas about containers and bean-like listening components have been around for a decade. Nevertheless these concepts, embodied in Java, are now changing information technology rapidly.
7. Optical Fiber Technologies and Wireless Technology. Very cheap communication and pervasive communication are changing the world’s cultural and political structures. There are many concepts in signal modulation and information theory that play a fundamental role in distributed computing, and it’s inappropriate to lump them all together into a single point.
8. The Interleaving Model. Reasoning about concurrent systems is difficult. But, it would have been impossible without a truly trivial and truly revolutionary idea: events that happen concurrently can be treated (in most cases) as though they happened in a total order. Thinking about events unfolding in a total order is much simpler than thinking about events happening at different times and different places. Of course, Newtonian time or real time is important. We have learned, however, to separate concerns about time by dealing with in two ways. Firstly, we ignore real time altogether and only concern ourselves with the order in which events occur. Secondly, we concern ourselves with system performance by re-introducing timing considerations. Temporal logics on the one hand, and queuing-theoretic and scheduling models on the other, demonstrate the influence of this idea.
9. Understanding Composition and State. The essence of software engineering is managing complexity by systematic composition: How we build complex systems by composing simpler components. In distributed systems we have to compose components that have state or memory. We have learned something about compositional structures of state-full components but a lot remains to be discovered. Threads, messages, listening components such as beans, and techniques for synchronization (message buffers, monitors, and barriers) are some of the ways in which we build complex systems from components. Algorithms for distributed resource management and exploiting (potentially out-of-date) global snapshots help us carry out systematic composition.
10. Adaptive Distributed Systems. The Internet is an adaptive system. The Air-Traffic Control system is adaptive. We are just beginning to develop theories of large adaptive systems. Fault-tolerance and adaptation to local congestion are just two examples of robustness through adaptation. We use relatively simple stochastic-process models or complex (but still too simple) discrete-event simulation models to help us understand adaptation, and we have come a long way. But, the best work lies ahead.

Biography:

Mani Chandy got his Ph.D. from MIT in Electrical Engineering, and his undergraduate degree from the Indian Institute of Technology, Madras. He has worked for IBM and Honeywell. He was a professor at the University of Texas at Austin for 17 years, and during that time he served as department chair. He has been at Caltech since 1987 where he is now the Simon Ramo Professor and Executive Officer for Computer Science. Mani has contributed to queuing network models of computer and communications performance and to reasoning about concurrent systems. He developed the UNITY theory with Jayadev Misra. Mani received the A.A.Michelson Award from the Computer Measurement Group for his contributions to computer performance modeling and the IEEE Koji Kobayashi Award for Computing and Communications. He is a member of the U.S. National Academy of Engineering.

Ken Kennedy

Position Statement:

In order to be included in the exalted category of most influential in the millennium, a concept should deeply influence the way we think about parallelism. For the most part, this means that the concepts should be both broad and abstract. Most of the important abstract parallel concepts that I can think of grew from analogs in every-day life. Examples include divide and conquer, cloning, load balancing, and synchronization. Even those that do not seem to arise from examples in everyday life, like scalable communication interconnects, can be seen after some thought to have strong analogs outside the world of computing. This leads us to ask the question: Are these concepts influential because they capture the familiar, or are they familiar because they are uniquely powerful concepts? When the panel has disposed of this issue, perhaps it can finally resolve the primacy of the chicken or the egg.

Biography:

Ken Kennedy is the Ann and John Doerr Professor of Computational Engineering and Director of the Center for Research on Parallel Computation (CRPC) at Rice University, an NSF Science and Technology Center that includes seven participating institutions and six affiliated sites across the country. He is a fellow of the Institute of Electrical and Electronics Engineers, the Association for Computing Machinery, and the American Association for the Advancement of Science and he has been a member of the National Academy of Engineering since 1990. From 1997 to 1999, he served as co-chair of the President's Information Technology Advisory Committee (PITAC). For his leadership in producing the PITAC report on funding of information technology research, he received the 1999 CRA Distinguished Service Award.

Prof. Kennedy has published over one hundred thirty technical articles and supervised thirty-two Ph.D. dissertations on programming support software for high-performance computer systems. Through the Center for Research on Parallel Computation, he is seeking to develop new strategies for supporting architecture-independent parallel programming, especially in science and engineering. In recognition of his contributions to software for high performance computation, he received the 1995 W. Wallace McDowell Award, the highest research award of the IEEE Computer Society. In 1999, he was named the third recipient of the ACM SIGPLAN Programming Languages Achievement Award.

Tom Leighton

Position Statement:

Here are some thoughts for some top advances in CS:

Gauss for his text and work on factoring
Incompleteness and incomputability
Linear Programming
FFT Algorithm
RSA, Merkle, and Diffie-Hellman for public-key crypto
Big O and asymptotics
Ford/Fulkerson for maxflow/mincut algorithm
NP-completeness and reductions
Depth-first search and applications
Basic data structures

Biography:

Tom Leighton is a Professor of Applied Mathematics at MIT and he is the Head of the Algorithms Group at MIT's Laboratory for Computer Science. He is also the Chief Scientist and a founder of Akamai Technologies, Inc.

Prof. Leighton is a leading expert in parallel and distributed computing. He has published over 100 research papers in the areas of parallel algorithms and architectures, communication protocols for networks, combinatorial optimization, probabilistic methods, VLSI computation and design, sequential algorithms, and graph theory. He is the author of two books, including a leading text on parallel algorithms and architectures. Prof. Leighton's research has resulted in several patents, which are actively used in industry.

Prof. Leighton received a BSE in EECS from Princeton in 1978 and a Ph.D. in Applied Mathematics from MIT in 1981. He has served on the faculty at MIT since 1982.

Jane Liu

Position Statement:

For distributed real-time applications, one of the most important breakthroughs is the principle of end-to-end resource management. Many time-critical and safety-critical distributed real-time applications adapt the pipeline architecture: to provide a service, such an application executes in turn on processors, network links, and other resources. (For example, an air-traffic control system processes sensor data on a signal processor, generates track records on a data processor, updates the tracking database, and so on.) The objective of distributed real-time scheduling and resource management is to meet end-to-end timing requirements (e.g., ensuring that a service is delivered to the user by a deadline). The principle of end-to-end scheduling and resource management is based on several recent advances including (1) ratebased algorithms for CPU and network scheduling that can provide guaranteed response times, (2) synchronization methods that can keeps worst-case response times small without sacrificing average response time, and (3) efficient, robust and accurate end-to-end schedulability analysis techniques that can be used for runtime admission control and acceptance test. Systems based on the principle can provide timing isolation to individual distributed applications and thus allow hard real-time applications to run together with soft real-time and nonrealtime applications.

Biography:

Jane W. S. Liu received her Bachelor of Science degree in Electrical Engineering in 1959 from the Cleveland State University, Ohio. She received her Master of Science and Electrical Engineers degrees in 1966 and her Doctor of Science degree in 1968 from the Massachusetts Institute of Technology. She is currently a Professor of Computer Science and of Electrical and Computer Engineering at the University of Illinois at Urbana-Champaign. Her research is in the areas of real-time systems, distributed systems and databases. In addition to publications in these areas, she has also lead several development efforts, including PERTS, a system of prototyping tools that supports the design, evaluation and validation of real-time systems, and EPIQ Open System, a runtime environment that allows real-time applications sharing a platform to be developed and validated independently. She served as the chair of the Technical Committee on Distributed Processing in 1989 and 1990. She was the program committee chair of the 1990 IEEE Real-Time Systems Symposium and the 1995 IEEE International Conference on Distributed Computing Systems. She was the editor-in-chief of IEEE Transactions on Computers from 1994 to 1998 and is an associate editor of Real-Time Systems Journal. She is an IEEE Fellow and a member of ACM.

Kang G. Shin

Position Statement:

My picks for most influential developments in parallel and distributed systems are:
- Arpanet and subsequent Internets (TCP/UDP/IP)
- Ethernet
- Web
- Client-server computing
- Distributed shared memory systems
- MPI
- PVM
- RPC
- Cluster computing
- Transaction systems

Biography:

Kang G. Shin is Professor and Director of the Real-Time Computing Laboratory, Department of Electrical Engineering and Computer Science, The University of Michigan, Ann Arbor, Michigan. He has supervised the completion of 40 Ph.D. theses, and authored/co-authored about 500 technical papers and numerous book chapters in the areas of distributed real-time computing and control, computer networking, fault-tolerant computing, and intelligent manufacturing. He has co-authored (jointly with C. M. Krishna) a textbook ‘‘Real-Time Systems,” McGraw Hill, 1997. In 1987, he received the Outstanding IEEE Transactions on Automatic Control Paper Award, and Research Excellence Award in 1989 and Outstanding Achievement Award in 1999 from The University of Michigan. His current research focuses on Quality of Service (QoS) sensitive computing and networking with emphases on timeliness and dependability. He has also been applying the basic research results to telecommunication and multimedia systems, intelligent transportation systems, embedded systems, and manufacturing applications.

Marc Snir

Position Statement:

Neural networks: the brain is a massively parallel computer, of kinds, so massively parallel computers are the way to develop a thinking machine. Neural nets also provide a graph paradigm of computation (a set of interconnected compute nodes, interacting through their connections) that is the foundation for much theory work in parallel complexity.

Computer chess (Deep Blue): a massively (well, highly) parallel computer can beat the best human chess player, so massively parallel computers are good. The most impressive feat of computation by a special-purpose parallel computer.

Digital simulations: the discretization of PDE systems and the replacement of physical experiments by simulated experiments. When a physical system, such as earth atmosphere, is digitized and simulated numerically, the parallelism is obvious: after all, the physical system evolves concurrently, with “nearest neighbors” interaction. This suggests a natural embedding of physical problems onto computational networks. Digital simulations are using the bulk of the compute power of large systems.

Packet switching. Essential, both for the Internet, and for large-scale parallel machines.

Cellular automata. The notion that regular arrays of simple automata could be universal compute systems; that they imitate life and can reproduce; that such an array can compute correctly even if components fail. This is the “Von Neumann parallel computer” – systolic arrays and much else are descendent of this work.

Biography:

Dr. Marc Snir is a senior manager at IBM T. J. Watson Research Center, where he leads research on scalable parallel systems. He and his group developed many of the technologies that led to the IBM SP product, and continue to work on future high-performance computers. Marc Snir received a Ph.D. in Mathematics, from the Hebrew University of Jerusalem, in 1979. He worked at NYU on the NYU Ultracomputer project in 1980-1982, and worked at the Hebrew University of Jerusalem from 1982 to 1986, when he joined the IBM T. J. Watson Research Center. He has published close to 100 journal and conference papers on computational complexity, parallel algorithms, parallel architectures, and parallel programming. He was a major contributor to the Message Passing Interface standards. Marc Snir is on the editorial board of Transactions on Computer Systems and Parallel Processing Letters. He is member of the IBM Academy of Technology, ACM Fellow and IEEE Fellow.

Larry Snyder

Position Statement:

Before considering the best parallelism ideas of the second millennium, consider first what was already known a thousand years ago. In fact, the two most significant ideas -- pipelining and concurrency -- have been known since the first millennium BC or earlier, since these concepts were used in grand construction projects such as the Pyramids of Cheops. Multithreading must have been invented thousands of years ago by mothers trying to get something accomplished while caring for kids. Though message passing has been a basic of human communication since prehistoric times, its application in a parallel system also dates at least to the first millennium BC, judging from the history of the Peloponnesian Wars. Warfare also motivated broadcasting, synchronization, and the use of semaphores. And of course, the eureka dates to Archimedes. So, at the dawn of the second millennium most fundamental concepts of parallelism were already well understood, making the second millennium's most significant contribution a “zero.” “One” was well known, of course, positional notation was a component of the widely used base-60 systems of ancient times, and the symbol for zero had been introduced in India and the Arabic countries. But, the synthesis that gave binary its other half, the coupling of “off” with the ubiquitous “on,” the ability to have nothing and know it, to recognize an empty task queue and initialize memory, or even to have memory at all, must be the second millennium's greatest contribution to parallelism and computation generally. “Zero,” and its partner “one,” which derived greater significance from their union, have made parallelism possible. They must be the 10 greatest concepts of the second millennium.

Biography:

Larry Snyder is professor of Computer Science and Engineering at the University of Washington, Seattle. He received his BA from the University of Iowa and his Ph.D. from Carnegie Mellon University. He is the inventor of configurable interconnect as is found in FPGAs, and the first computer based on the idea, the Configurable, Highly Parallel (CHiP) machine. Snyder is co-inventor of the Chaos Router, a randomizing, nonminimal adaptive packet routing technique. His team built Poker, the first parallel programming environment. He created the CTA, an abstract parallel machine model, and with students and colleagues created Phase Abstractions, a parallel programming model. He and his students have applied these two concepts to create ZPL parallel programming language, a fully implemented and freely available parallel programming system. ZPL programs run as fast as C with message passing, and port to all parallel and sequential platforms. His project is presently working on Advanced ZPL, a complete parallel language. Snyder is a fellow of IEEE and ACM.

Thomas Sterling

Position Statement:

Myriad concepts of parallelism and distributed action and control have driven the rapid acceleration of computing performance to the Teraflops regime accomplished by today's largest and most parallel systems. Many such parallel organizations have had significant influence including multiple computing, multiprocessing, SIMD, SPMD, data parallel, multithreaded, vector and pipelining, prefetching and caching, and dataflow and functional execution. But interestingly, the majority of these were manifest in human systems and organizations long before the advent of the electronic digital computing and applied to a much broader range of activities than computing itself. This brief discussion will highlight the impact of parallelism and distributed actions on social and commercial organizations throughout the last millennium and show their corresponding role in computing organizations as well. The speaker will also identify some concepts, which have yet to have substantial impact on commercial or high end computing, but may likely do so in the next century.

Biography:

Dr. Thomas Sterling is a Principal Scientist at the NASA Jet Propulsion Laboratory and a Faculty Associate at the California Institute of Technology. The Principal Investigator on the HTMT architecture project, he leads a team of over a dozen institutions in an in-depth exploration of an innovative approach to achieving Petaflops-scale computing by integrating advanced technologies in a dynamic adaptive latency management structure. Sterling is also engaged in the development of advanced Processor-in-Memory execution models, and the invention of Continuum Computer Architecture, a general purpose cellular architecture approach to ultra-scale computing. A graduate of MIT (Ph.D., 1984) and Hertz Fellow, Dr. Sterling has engaged in research studies in parallel computing systems architecture and software over the last two decades. He is best known for his development (with colleague Don Becker) of Beowulf-class PC cluster computing when at the NASA Goddard Space Flight Center and as a leader of the National Petaflops Initiative while at Caltech and JPL. He is the co-author of the books: “Enabling Technologies for Petaflops Computing” and “How to Build a Beowulf” and holds six patents.

Acknowledgments: H. J. Siegel, the panel organizer, thanks all of the panelists for agreeing to be on the panel and for providing the material for the symposium proceedings. H. J. Siegel also thanks Shoukat Ali and Sahra Ali for their help in preparing this proceedings paper.