Laxmikant V. Kalé

Office: Department of Computer Science
1304 W. Springfield Avenue
Urbana, IL 61801Fax: (217) 333-3501
Phone: (217) 244-0094
Email: kale@cs.uiuc.edu

Education

Ph.D., Computer Science, December 1985, SUNY at Stony Brook, NY 11794
M.E., Computer Science, July 1979, Indian Institute of Science, Bangalore, India.
B. Tech. Elect. Engr., May 1977, Banaras Hindu University, Varanasi, India

Academic Positions Held

Professor, Computer Science: August 2000—present
Associate Professor, Computer Science: August 1991—August 2000
Assistant Professor, Computer Science: August 1985—August 1991
Joint/Affiliate Faculty member, Beckman Institute, EcE and M&IE Depts.

Professional Societies and Activities (past 5 years)

Member: ACM, IEEE, IEEE Computer Society
Vice Chair: ICPP (Intl. Conf. on Parallel Programming) 2003
Publicity Chair and Program Committee member, JavaGrande 2002
Program Committee member, IPDPS 2001.
General Chair, IPPS/SPDPA Workshop on Runtime Systems for Parallel Programming, 1999, 2000.
Program Committee member, 7'th and 8'th Int. Conf. on
High Performance Computing (HiPC'2000,2001)
Program Committee member, Intl. Conference on Supercomputing, 1998.
Reviewer for numerous computer science conferences and journals.

Awards

Gordon Bell award (Special category), 2002 (SC2002)
Finalist, Gordon Bell award (Special category), 2000 (SC2000)

Research Interests

Scalable Parallel Programming
Re-use of parallel software
Computational Science and Engineering
Irregular Parallel Problems
Dynamic Load Balancing
Parallel Application Frameworks

Significant Projects:

Charm++: Object-based Parallel Programming System, based on processor-virtualization, with intelligent Runtime support
Adaptive MPI: MPI implementation based on processor-virtualization
Faucets: Efficient resource allocation across the computational grid, and within clusters
Parallel Adaptive Frameworks for CSE including FEM, structured multiblock and AMR
BioPhysics: Parallel Molecular Dynamics and collaborative molecular modeling.
Extensive collaborative Projects with Scientists and Engineers across a variety of fields (including Quantum Mechanical Modeling, Molecular Dynamics, Rocket Simulation, Computational cosmology, Material Structures, and Space-time meshing).
Software Distribution: Software tools produced in all the above projects distributed via internet.

Graduate Students

Past Master's students (14)
Past Ph.D. students (7)
Current Master's students (10)
Current Ph.D. students (7)

Laxmikant V. Kale
Research Accomplishments in Perspective

My research has centered upon parallel computing, and more specifically on improving (computer) performance and (human) productivity in parallel programming. My choice of specific research topics and strategies is guided by the following (admittedly subjective) axioms.

- The research must be Computer Science centered yet application oriented[64]. Centering one's research on a particular application will typically not lead to development of enabling technologies, applicable across a broad variety of applications. At the same time, without application orientation, one is in danger of developing tools and techniques that aren't useful to real applications. Using application benchmarks to demonstrate CS techniques isn't enough. So, one should develop core "enabling technologies" while focusing on full-fledged "real" applications, via intensely collaborative interdisciplinary research.

- As the systems area of computer science is an engineering discipline, it is necessary to embody the techniques developed as part of our research into reusable software, and to make such software available to other researchers/practitioners.

Consequently, my research has focused on developing the *middle layers* of enabling technologies, with portability and data driven execution as the underlying themes, with dynamic and irregular applications and environments as the context, and with performance, scalability, and ease of programming as the objectives. The following paragraphs summarize components of my past and ongoing research.
Portability: In 1986-87, when the *chare kernel* [107, 112, 113, 114] (predecessor to *Charm*) was being designed with the objective of providing portability, it was deemed an unrealistic objective, especially portability across shared and distributed memory machines. Such portability is now passe.
Data driven objects: Data-driven execution ideas were at the basis of data-flow machines proposed in the '70s. Software-based data driven execution was used in several research projects, including Rediflow (parallel functional languages), our own research [115] on parallel prolog and Agha's elaboration of Hewitt's Actor model. Chare kernel was one of the first pragmatic (e.g. C based) data-driven (active) object based system; [107, 112, 113, 114] it was available on most parallel machines of the day including workstation networks. [79] Other parallel object systems in that timeframe include ABCL, ORCA, PC++, and UC++. One of the first parallel implementations of Actors (HAL, Houck and Agha, ICPP '92) was developed on top of Charm. The performance and modularity benefits of data driven execution were systematically studied and demonstrated by us [116, 65, 61, 68]. We've continued to elaborate and develop the Charm++ parallel object model, which is at the foundation of much of our research. Beyond data driven objects themselves, Charm++ concepts include object groups, information sharing abstractions,[71] object-placement based load balancing [117] and prioritized scheduling[66]. A more recent addition is object arrays [55, 42] . Charm++ has been used extensively for developing applications by us and others. Especially notable is a series of results in parallel VLSI CAD applications obtained by Bannerjee and Ramkumar using Charm.
Run-time systems and multi-paradigm interoperability: It is difficult to persuade broad developer community to switch to a new paradigm(such as Charm++). Further, different paradigms may be appropriate for different components of a single application. We developed the Converse runtime framework [57, 47] to address these issues. Converse provides interoperability between multiple data-driven and multithreaded paradigms as well as with traditional message passing

paradigms. Converse is a component based architecture, with components encapsulating commonly needed run-time functionalities. As a result, to develop a portable run-time library for a new paradigm on top of Converse becomes extremely easy. We implemented more than ten such (interoperable) libraries, representing several parallel programming languages and paradigms [47, 56].

Performance analysis and visualization tools: A parallel object language such as Charm++ provides a wealth of information to the runtime system about the behavior of the application program. Exploiting this, we developed performance visualization and automatic performance analysis techniques that can identify performance hurdles in an application [76, 50]. Going a step further, we developed closed-loop techniques [53, 54] that analyze the performance data from one run to generate a "hints" database that is utilized by subsequent runs (or iterations) to improve performance.

Parallel molecular dynamics: A large fraction of my research effort during the past few years was devoted to a collaborative project [118, 119, 9] aimed at developing a scalable and efficient parallel molecular dynamics program. The resultant program, NAMD [7], is a production quality program used to produce over a dozen published results in biophysics. It is written using Charm++, and uses Converse [44] to interoperate with libraries written in other languages. We believe it is the fastest parallel molecular dynamics program (speedup of 180 on 220 processors, not surpassed by any other production quality program). Although it has required significant effort on our part to support the nitty-gritty of a real application, it has also been a wellspring of ideas for Computer Science research.

Migrating objects and threads: To deal with unpredictable dynamic behavior of many applications, it is necessary to adaptively move an appropriate fraction of work away from overloaded processors. Charm++ objects provide a potential mechanism for this purpose. We've developed a framework that supports object migration, as well as thread migration.

Dynamic and irregular applications in science and engineering: We are developing a broad "multi-domain partitioning" approach for such applications, based on objects and the migration framework. Strategies for detecting application-induced imbalances, and adaptively reacting to them to restore efficiency are being developed [111]. Consistent with our philosophy, these are being tested on applications in biophysics [46], rocket simulation, and solidification of metals, in collaborative projects. We expect the resultant framework to be a broadly applicable "enabling technology".

Cluster computing: From early versions of Charm [114], we have targeted workstation clusters for parallel computing. Technological advances in recent years have given a new fillip to this area [43]. Based on the object migration framework, we have developed strategies [42] for dealing with extraneous load imbalances generated in a time shared cluster.

State-space search/non-numeric computations: One of the early successes of our parallel object methodology was in the area of state-space search. Based on the ideas of fine-grain prioritization, and scalable load balancing, we've obtained strong results [70] in state space search for any one solution (which was a very difficult problem), iterative deepening, branch and bound [66], and so on.

Web based interactive parallel programs: We have developed a client-server interface that allows parallel programs to interact with outside world. Building on this, we have developed a web interface [109, 120] that allows one to attach to running parallel programs from a browser anywhere, monitor its performance, debug it, and interact with it.

The citations numbers above refer to the publication list in the resume. Further information about these topics can be obtained from http://charm.cs.uiuc.edu

Publications by (overlapping) categories

Parallel Objects and Processor Virtualization: [113] [10] [3] [7] [112] [117] [121] [122] [114] [79] [76] [123] [116] [65] [71] [64][61] [60] [55] [53] [54] [50] [68] [51] [49] [42]

Intelligent Runtime Systems: [113] [44] [6] [7] [114] [79] [57] [56] [52] [47] [45]

Parallel Programming Tools: [76] [50] [109] [120]

Cluster Computing: [114] [42] [43]
Molecular Dynamics: [118] [119] [9] [44] [6] [7] [124] [47] [45] [46] [43]

Science and Engineering Applications

Parallel Prolog: [125] [126] [127] [128] [129] [130] [131] [132] [133] [134] [135] [136] [137] [122] [138] [139] [140] [78]

Non-numeric (Symbolic) Computations: [125] [141] [142] [143] [115] [144] [145] [146] [66]

Publications

[1] Milind Bhandarkar, Gila Budescu, William F. Humphrey, Jesus A. Izaguirre, Sergei Izrailev, Laxmikant V. Kale, Dorina Kosztin, Ferenc Molnar, James C. Phillips, and Klaus Schulten. Biocore: A collaboratory for structural biology. *In Agostino G. Bruzzone, Adelinde Uchrmacher, and Ernest H. Page, editors, Proceedings of the SCS International Conference on Web-Based Modeling and Simulation*, pages 242–251, 1999.

[2] Tarmar Schlick, Robert Skeel, Axel Brünger, Laxmikant Kalé, John A.Board Jr, Jan Hermans, and Klaus Schulten. Algorithmic chanllenges in computational molecular biophysics. *Journal of Computational Physics*, 151:9–48, 1999.

[3] L. V. Kalé. Programming Languages for CSE: the state of the art. *IEEE Computational Science and Engineering*, 5(2):18–26, April-June 1998.

[4] J. Yelon and L. V. Kalé. Static networks: A powerful and elegant extension to concurrent object-oriented languages. In *Lecture Notes in Computer Science*, volume 1505, pages 143–150. Springer Verlag, 1998.

[5] Tamar Schlick, Robert D. Skeel, Axel T. Brünger, Klaus Schulten, Laxmikant V. Kale, Jan Hermans, and Jr. John A. Board. Computational biophysics today. *Journal of Computational Physics*, 1998. Submitted.

[6] James C. Phillips, Robert Brunner, Aritomo Shinozaki, Milind Bhandarkar, Neal Krawetz, Laxmikant Kalé, Robert D. Skeel, and Klaus Schulten. Avoiding algorithmic obfuscation in a message-driven parallel MD code. In P. Deuflhard, J. Hermans, B. Leimkuhler, A. Mark, S. Reich, and R. D. Skeel, editors, *Computational Molecular Dynamics: Challenges, Methods, Ideas*, volume 4 of *Lecture Notes in Computational Science and Engineering*, pages 472–482. Springer-Verlag, November 1998.

[7] Laxmikant Kalé, Robert Skeel, Milind Bhandarkar, Robert Brunner, Attila Gursoy, Neal Krawetz, James Phillips, Aritomo Shinozaki, Krishnan Varadarajan, and Klaus Schulten. NAMD2: Greater scalability for parallel molecular dynamics. *Journal of Computational Physics*, 151:283–312, 1999.

[8] L. V. Kale and Sanjeev Krishnan. Charm++: Parallel Programming with Message-Driven Objects. In Gregory V. Wilson and Paul Lu, editors, *Parallel Programming using C++*, pages 175–213. MIT Press, 1996.

[9] Mark Nelson, William Humphrey, Attila Gursoy, Andrew Dalke, Laxmikant Kale, Robert D. Skeel, and Klaus Schulten. NAMD—a parallel, object-oriented molecular dynamics program. *Intl. J. Supercomput. Applics. High Performance Computing*, 10(4):251–268, Winter 1996.

[10] Joshua Yelon and L. V. Kale. Agents: An undistorted representation of problem structure. In *Lecture Notes in Computer Science*, volume 1033, pages 551–565. Springer-Verlag, August 1995.

[11] L. V. Kale, B. H. Richards, and T. D. Allen. Efficient parallel graph coloring with prioritization. In *Lecture Notes in Computer Science*, volume 1068, pages 190–208. Springer-Verlag, August 1995.

[12] L. V. Kalé, B. Ramkumar, A. B. Sinha, and V. A. Saletore. The CHARM Parallel Programming Language and System: Part II – The Runtime system. *IEEE Transactions on Parallel and Distributed Systems*, 1994.

[13] L. V. Kalé, B. Ramkumar, A. B. Sinha, and A. Gursoy. The CHARM Parallel Programming Language and System: Part I – Description of Language Features. *IEEE Transactions on Parallel and Distributed Systems*, 1994.

[14] M.Nelson, W. Humprey A. Gursoy, A¿ Dalke, L.V. Kalè, R Skeel, K. Schulten, and R. Kuffrin. MDScope-A visual COmputing Environment for Structural Biology. *Computer Physics COmmunications*, 91:111–134, October 1995.

[15] John A. Board Jr., Laxmikant V. Kale, Klaus Schulten, Robert D. Skeel, , and Tamar Schlick. Modeling biomolecules: Large sclaes, longer durations. *IEEE Computational Science & Engineering*, 1:19–30, Winter 1994.

[16] David Sehr, L.V. Kale, and David A. Padua. Loop transformation for prolog programs. In *Lecture Notes in Computer Science*, pages 374–389. Springer Verlag, 1993.

[17] L.V.Kalè and B. Ramkumar. *The Reduce-OR-Process Model for Parallel Logic Programming on Nonshared Memory Machines.* John Wiley, June 1992. Editors: M. Wise and Peter Kacssuk.

[18] B.Ramkumar and L.V. Kalè. A join algorithm for combining AND parallel solutions in AND/OR parallel systems. *International Journal of Parallel Processing*, 1992.

[19] B. Ramkumar and L.V. Kale. Machine independent AND and OR parallel execution of logic programs: Part I and II. *IEEE Transactions on Parallel and Distributed Systems*, 5(2), Feb. 1991.

[20] L.V. Kalé. *Journal of Logic Programming*, 11(1):55–84, July 1991.

[21] B.Ramkumar and L.V. Kalé. Machine independent AND and OR parallel execution of logic programs Part II - compiled execution. *IEEE Transactions on Parallel and Distributed Systems*, 5(2):181–192, February 1994.

[22] B. Ramkumar and L.V. Kalé. Machine independent AND and OR parallel execution of logic programs PartI - the binding environment. *IEEE Transactions on Parallel and Distributed System*, 5(2):170–180, February 1994.

[23] L.V. Kalé and V. Saletore. Parallel state-space search for a first solution with consistent linear speedups. *Internaltional Journal of Parallel Programming*, 19(4):251–293, 1990.

[24] W.W. Shu and L.V. Kalé. Chare Kernel - a runtime support system for parallel computations. *Journal of Parallel and Distributed Computing*, 11:198–211, 1990.

[25] L.V. Kalè. An almost perfect heuristic or the N-queens problem. *Information Processing Letters*, 34(4):173–178, April 1990.

[26] L.V. Kalè, D.A. Padua, and D.C. Sehr. OR parallel execution of Prolog programs with side effects. *The Journal of Supercomputing*, 2(2):209–223, October 1988.

[27] Laxmikant V. Kalé, Sameer Kumar, Gengbin Zheng, and Chee Wai Lee. Scaling molecular dynamics to 3000 processors with projections: A performance analysis case study. In *Terascale Performance Analysis Workshop (ICCS'03)*, Melbourne, Australia, June 2003.

[28] Laxmikant V. Kalé. The virtualization model of parallel programming : Runtime optimizations and the state of art. In *LACSI 2002*, Albuquerque, October 2002.

[29] James C. Phillips, Gengbin Zheng, Sameer Kumar, and Laxmikant V. Kalé. Namd: Biomolecular simulation on thousands of processors. In *Proceedings of SC 2002*, Baltimore, MD, September 2002.

[30] Orion Sky Lawlor and L. V. Kalé. A voxel-based parallel collision detection algorithm. In *Proceedings of the International Conference in Supercomputing*, pages 285–293. ACM Press, June 2002.

[31] James Phillips, Gengbin Zheng, and Laxmikant V. Kalé. Namd: Biomolecular simulation on thousands of processors. In *Workshop: Scaling to New Heights*, Pittsburgh, PA, May 2002.

[32] Gengbin Zheng, Arun Kumar Singla, Joshua Mostkoff Unger, and Laxmikant V. Kalé. A parallel-object programming model for petaflops machines and blue gene/cyclops. In *Workshop on Massively Parallel Processing (IPDPS'02)*, Fort Lauderdale, FL, April 2002.

[33] Laxmikant V. Kalé, Sameer Kumar, and Jayant DeSouza. A malleable-job system for timeshared parallel machines. In *2nd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGrid 2002)*, May 2002.

[34] Neelam Saboo, Arun Kumar Singla, Joshua Mostkoff Unger, and L. V. Kalé. Emulating petaflops machines and blue gene. In *Workshop on Massively Parallel Processing (IPDPS'01)*, San Francisco, CA, April 2001.

[35] Milind Bhandarkar and L. V. Kale. An Interface Model for Parallel Components. In *Proceedings of the Workshop on Languages and Compilers for Parallel Computing (LCPC), Cumberland Falls, KY*, August 2001.

[36] O. Lawlor and L. V. Kalé. Supporting dynamic parallel object arrays. In *Proceedings of ACM 2001 Java Grande/ISCOPE Conference*, pages 21–29, Stanford, CA, Jun 2001.

[37] Robert K. Brunner, James C. Phillips, and Laxmikant V. Kale. Scalable Molecular Dynamics for Large Biomolecular Systems. In *Proceedings of Supercomputing (SC) 2000, Dallas, TX, November 2000. Nominated for Gordon Bell Award.*, November 2000.

[38] L. V. Kale, Milind Bhandarkar, and Robert Brunner. Run-time Support for Adaptive Load Balancing. In J. Rolim, editor, *Lecture Notes in Computer Science, Proceedings of 4th Workshop on Runtime Systems for Parallel Programming (RTSPP) Cancun - Mexico*, volume 1800, pages 1152–1159, March 2000.

[39] Eric deStruler, Jay Hoeflinger, L. V. Kale, and Milind Bhandarkar. A New Approach to Software Integration Frameworks for Multi-physics Simulation Codes. In *Proceedings of IFIP TC2/WG2.5 Working Conference on Architecture of Scientific Software, Ottawa, Canada*, pages 87–104, October 2000.

[40] Milind Bhandarkar, L. V. Kale, Eric de Sturler, and Jay Hoeflinger. Object-Based Adaptive Load Balancing for MPI Programs. In *Proceedings of the International Conference on Computational Science, San Francisco, CA, LNCS 2074*, pages 108–117, May 2001.

[41] Milind Bhandarkar and L. V. Kalé. A Parallel Framework for Explicit FEM. In M. Valero, V. K. Prasanna, and S. Vajpeyam, editors, *Proceedings of the International Conference on High Performance Computing (HiPC 2000), Lecture Notes in Computer Science*, volume 1970, pages 385–395. Springer Verlag, December 2000.

[42] Robert K. Brunner and Laxmikant V. Kalé. Adapting to load on workstation clusters. In *The Seventh Symposium on the Frontiers of Massively Parallel Computation*, pages 106–112. IEEE Computer Society Press, February 1999.

[43] Laxmikant Kale, Robert Brunner, James Phillips, and Krishnan Varadarajan. Application performance of a linux cluster using converse. In *Proceedings of 3rd Workshop on Runtime Systems for Parallel Programming, Rolim et al (Eds)*, Lecture Notes in Computer Science 1586, pages 483–495. IPPS/SPDP, Springer, 1999.

[44] L. V. Kalé, M. Bhandarkar, R. Brunner, N. Krawetz, J. Phillips, and A. Shinozaki. Namd: A case study in multilingual parallel programming. In Zhiyuan Li, Pen-Chung Yew, Siddharta Chatterjee, Chua-Huang Huang, P. Sadayappan, and David Sehr, editors, *Languages and Compilers for Parallel Computing*, number 1366 in Lecture Notes in Computer Science, pages 367–381. Springer-Verlag, 1998.

[45] Robert Brunner, Laxmikant Kalé, and James Phillips. Flexibility and interoperability in a parallel molecular dynamics code. In *Object Oriented Methods for Inter-operable Scientific and Engineering Computing*, pages 80–89. SIAM, October 1998.

[46] L. V. Kalé, Milind Bhandarkar, and Robert Brunner. Load balancing in parallel molecular dynamics. In *Fifth International Symposium on Solving Irregularly Structured Problems in Parallel*, volume 1457 of *Lecture Notes in Computer Science*, pages 251–??, 1998.

[47] Robert Brunner L. V. Kale, Milind Bhandarkar and Joshua Yelon. Multiparadigm, Multilingual Interoperability: Experience with Converse. In *Proceedings of 2nd Workshop on Runtime Systems for Parallel Programming (RTSPP) Orlando, Florida - USA*, Lecture Notes in Computer Science, March 1998.

[48] L. V. Kalé, Milind Bhandarkar, Robert Brunner, N. Krawetz, J. Phillips, and A. Shinozaki. Namd: A case study in multilingual parallel programming. In *Proc. 10th International Workshop on Languages and Compilers for Parallel Computing*, Minneapolis, Minnesota, August 1997.

[49] L. V. Kalé, Milind Bhandarkar, and Terry Wilmarth. Design and implementation of parallel java with a global object space. In *Proc. Conf. on Parallel and Distributed Processing Technology and Applications*, pages 235–244, Las Vegas, Nevada, July 1997.

[50] Amitabh Sinha and L. V. Kale. Towards Automatic Peformance Analysis. In *Proceedings of International Conference on Parallel Processing*, volume III, pages 53–60, August 1996.

[51] L. V. Kale and Milind Bhandarkar. Structured Dagger: A Coordination Language for Message-Driven Programming. In *Proceedings of Second International Euro-Par Conference*, volume 1123-1124 of *Lecture Notes in Computer Science*, pages 646–653, September 1996.

[52] L. V. Kale and Joshua Yelon. Threads for Interoperable Parallel Programming. In *Proc. 9th Conference on Languages and Compilers for Parallel Computers*, San Jose, California, August 1996.

[53] Sanjeev Krishnan and L. V. Kale. Automating Parallel Runtime Optimizations Using Post-Mortem Analysis. In *Proc. 10th ACM International Conference on Supercomputing*, pages 221–228, Philadelphia, PA, May 1996.

[54] Sanjeev Krishnan and L. V. Kale. Automating Runtime Optimizations for Load Balancing in Irregular Problems. In *Proc. Conf. on Parallel and Distributed Processing Technology and Applications*, San Jose, California, August 1996.

[55] Sanjeev Krishnan and L. V. Kale. A parallel array abstraction for data-driven objects. In *Proceedings of Parallel Object-Oriented Methods and Applications Conference*, Santa Fe, NM, February 1996.

[56] Milind Bhandarkar and L. V. Kale. MICE: A Prototype MPI Implementation in Converse Environment. In *Proceedings of the second MPI Developers Conference*, pages 26–31, South Bend, Indiana, July 1996.

[57] L. V. Kale, Milind Bhandarkar, Narain Jagathesan, Sanjeev Krishnan, and Joshua Yelon. Converse: An Interoperable Framework for Parallel Programming. In *Proceedings of the 10th International Parallel Processing Symposium*, pages 212–217, April 1996.

[58] L. V. Kale, Milind Bhandarkar, Narain Jagathesan, Sanjeev Krishnan, and Joshua Yelon. Converse: An Interoperable Framework for Parallel Programming. In *Proceedings of the 10th International Parallel Processing Symposium*, pages 212–217, April 1996.

[59] L. V. Kale, Milind Bhandarkar, Narain Jagathesan, Sanjeev Krishnan, and Joshua Yelon. Converse: An Interoperable Framework for Parallel Programming. In *Proceedings of the 10th International Parallel Processing Symposium*, pages 212–217, April 1996.

[60] Sanjeev Krishnan and L. V. Kale. A parallel adaptive fast multipole algorithm for n-body problems. In *Proceedings of the International Conference on Parallel Processing*, pages III 46 – III 50, August 1995.

[61] L. V. Kale and Attila Gursoy. Modularity, reuse and efficiency with message-driven libraries. In *Proc. 27th Conference on Parallel Processing for Scientific Computing*, pages 738–743, February 1995.

[62] E. Kornkven and L. V. Kale. Efficient implementation of high performance fortran via adaptive scheduling – an overview. In *Proceedings of the International Workshop on Parallel Processing*, Bangalore, India, December 1994.

[63] E. Kornkven and L. V. Kale. Efficient implementation of high performance fortran via adaptive scheduling – an overview. In *Proceedings of the International Workshop on Parallel Processing*, Bangalore, India, December 1994.

[64] L. V. Kale. Application oriented and computer science centered HPCC research. In *Proceedings of Developing a CS Agenda for High-Performance Computing*, pages 98–105, March 1994. Position Paper.

[65] A. Gursoy and L.V. Kalé. Dagger: Combining the Benefits of Synchronous and Asynchronous Communication Styles. In H. G. Siegel, editor, *Proceedings of the 8th International Parallel Processing Symposium*, pages 590–596, Cancun, Mexico, April 1994.

[66] A. Sinha and L.V. Kalé. A load balancing strategy for prioritized execution of tasks. In *Seventh International Parallel Processing Symposium*, pages 230–237, Newport Beach, CA., April 1993.

[67] L.V. Kale and Sanjeev Krishnan. Medium grained execution in concurrent object-oriented systems. In *Workshop on Efficient Implementation of Concurrent Object Oriented Languages, at OOPSLA 1993*, September 1993.

[68] Attila Gursoy and L. V. Kale. Simulating Message-Driven Programs. In *Proceedings of International Conference on Parallel Processing*, volume III, pages 223–230, August 1996.

[69] L. V. Kale and Attila Gursoy. Performance benefits of message driven execution. In *Intel Supercomputer User's Group*, St. Louis, October 1993.

[70] L.V. Kale, B. Ramkumar, V. Saletore, and A. B. Sinha. Prioritization in parallel symbolic computing. In T. Ito and R. Halstead, editors, *Lecture Notes in Computer Science*, volume 748, pages 12–41. Springer-Verlag, 1993.

[71] A. Sinha and L.V. Kalé. Information Sharing Mechanisms in Parallel Programs. In H.J. Siegel, editor, *Proceedings of the 8th International Parallel Processing Symposium*, pages 461–468, Cancun, Mexico, April 1994.

[72] L.V. Kalé and S. Krishnan. CHARM++: A Portable Concurrent Object Oriented System Based on C++. In A. Paepcke, editor, *Proceedings of OOPSLA'93*, pages 91–108. ACM Press, September 1993.

[73] L. V. Kale and Sanjeev Krishnan. A comparison based parallel sorting algorithm. In *Proceedings of the 22nd International Conference on Parallel Processing*, pages 196–200, St. Charles, IL, August 1993.

[74] Amitabh Sinha and L.V. Kale. A load balancing strategy for prioritized execution of tasks. In *Workshop on Dynamic Object Placement and Load Balancing, in co-operation with ECOOP's 92*, Utrecht, The Netherlands, April 1992.

[75] A. Gursoy, L.V. Kale, and S.P. Vanka. Unsteady fluid flow calculations using a machine independent parallel programming environment. In R. B. Pelz, A. Ecer, and J. Hauser, editors, *Parallel Computational Fluid Dynamics '92*, pages 175–185. North-Holland, 1993.

[76] L.V. Kalé and Amitabh Sinha. Projections: A preliminary performance tool for charm. In *Parallel Systems Fair, International Parallel Processing Symposium*, pages 108–114, Newport Beach, CA, April 1993.

[77] L.V. Kale. Parallel problem solving. In Vipin Kumar, P. S. Gopolakrishnan, and L. N. Kanal, editors, *Parallel Algorithms for Machine Intelligence and Vision*, pages 146–181. Springer-Verlag, 1989.

[78] D. Sehr and L.V. Kale. Estimating the inherent parallelism in Prolog programs. In *Proceedings of the 1992 International Conference on Fifth Generation Computer Systems*, Tokyo, Japan, June 1992.

[79] W. Fenton, B. Ramkumar, V.A. Saletore, A.B. Sinha, and L.V. Kale. Supporting machine independent programming on diverse parallel architectures. In *Proceedings of the International Conference on Parallel Processing*, pages 193–201, St. Charles, IL, August 1991.

[80] David C. Sehr, Laxmikant V. Kalé, and David A. Padua. Fortran-style transformations for functional programs (Extended abstract). In ICPP91 [147], pages 282–283.

[81] A. Gursoy and L.V. Kale. High-level support for divide-and-conquer parallelism. In *Proceedings of Supercomputing '91*, pages 283–292, November 1991.

[82] L.V. Kalé and B. Ramkumar. Implementing a parallel Prolog interpreter on multiprocessors. In *5th International Parallel Processing Symposium*, pages 543–548, Anaheim, CA, April 1991.

[83] L.V. Kalé and Vikram Saletore. Efficient parallel execution of IDA* on shared and distributed memory multiprocessors. In *Proceedings of the Sixth Distributed Memory Computing Conference (DMCC6)*, Portland, Oregon, April 1991.

[84] L. V. Kalé and B. Ramkumar. Joining AND-parallel solutions in AND/OR parallel systems. In *Proceedings of the North American Conference on Logic Programming*, pages 614–631, October 1990.

[85] L.V. Kale. The Chare Kernel parallel programming language and system. In *Proceedings of the International Conference on Parallel Processing*, volume II, pages 17–25, August 1990.

[86] V. Saletore and L.V. Kale. Consistent linear speedups for a first solution in parallel state-space search. In *Proceedings of the AAAI*, pages 227–233, August 1990.

[87] B. Ramkumar and L. V. Kalé. A Chare Kernel implementation of a parallel Prolog compiler. In *Proceedings of the Second Conference on Principles and Practice of Parallel Programming*, pages 99–108, Seattle, WA, March 1990.

[88] B. Ramkumar and L. V. Kalé. An abstract machine for the reduce or process model for parallel Prolog. In *Proceedings of Knowledge Based Computer Systems Conference*, pages 267–276, Bombay, India, December 1989.

[89] W. W. Shu and L. V. Kalé. A dynamic load balancing strategy for the Chare Kernel system. In Supercomputing89 [148], pages 389–398.

[90] B. Ramkumar and L. V. Kalé. Compiled execution of the Reduce-Or process model on multiprocessors. In *The North American Conference on Logic Programming*, pages 313–331, Oct 1989.

[91] V. Saletore and L. V. Kalé. Obtaining first solutions fast in parallel problem solving. In *The North American Conference on Logic Programming*, pages 390–408, Oct 1989.

[92] L. V. Kalé and W. Shu. The Chare Kernel base language: Preliminary performance results. In ICPP89 [149], pages 118–121.

[93] V. P. Pethe, C. P. Rippey, and L. V. Kalé. A specialized expert system for judicial decision support. In *Proceedings of the Second International Conference on AI and Law (ICAIL-89)*, pages 190–194, June 1989.

[94] L. V. Kalé. A brief perspective on parallel programming. In *IEEE Tencon Regional Conference (Invited)*, pages 1085–1088, Bombay, India, November 1989.

[95] L. V. Kalé, B. Ramkumar, and W. Shu. Parallel Prolog on Intel's iPSC/2. In *Proceedings of the Fourth Conference on Hypercube Concurrent Computers and Applications*, March 1989.

[96] L. V. Kalé. The mesh superceded? In *Proceedings of ACM Computer Science Conference*, pages 180–187, Feb 1989. Also available as TR#1362, dated July 1987.

[97] L. V. Kalé. A tree representation for parallel problem solving. In *Proceedings of AAAI*, pages 677–681, August 1988.

[98] L. V. Kalé, B. Ramkumar, and W. Shu. A memory independent binding environment for AND and OR parallel execution of logic programs. In *The Joint International Conference/Symposium on Logic Programming*, pages 1223–1240, Seattle, WA, 1988.

[99] L. V. Kalé. Comparing the performance of two dynamic load distribution methods. In *Proceedings of the 1988 International Conference on Parallel Processing*, pages 8–11, St. Charles, IL, August 1988.

[100] M. Gooley, L. V. Kalé, D. Padua, D. Sehr, B. Ramkumar, U. S. Reddy, W. Shu, and B. Wah. Prolog research at University of Illinois. In *Compcon*, pages 68–73, San Francisco, Feb 1988.

[101] L. V. Kalé and B. Ramkumar. D-trees: A class of dense regular interconnection topologies. In *Proceedings of the Frontiers of Massively Parallel Computation*, pages 207–210, Oct 1988.

[102] L. V. Kalé. Parallel execution of logic programs: the REDUCE-OR process model. In *Proceedings of Fourth International Conference on Logic Programming*, pages 616–632, May 1987.

[103] L. V. Kalé. 'Completeness' and 'Full parallelism' of parallel logic programming schemes. In *Proceedings of 1987 Symposium on Logic Programming*, pages 125–133, San Francisco, CA, 1987.

[104] L.V. Kalé. Optimal communication neighborhoods. In *Proceedings of the 1986 International Conference on Parallel Processing*, pages 823–826, St. Charles, Illinois, August 1986.

[105] L. V. Kalé. Lattice-mesh: a multi-bus topology. In *Proceedings of the 1985 International Conference on Parallel Processing*, pages 700–703, St. Charles, Illinois, August 1985.

[106] L. V. Kalé and D. S. Warren. A class of architectures for Prolog machine. In *Proceedings of the Conference on Logic Programming*, pages 171–182, Uppsala, Sweden, July 1984.

[107] L. V. Kalé and W. Shu. The Chare Kernel language for parallel programming: A perspective. Technical Report UIUCDCS-R-88-1451, Department of Computer Science, University of Illinois, August 1988.

[108] Amitabh B. Sinha, L. V. Kalé, and B. Ramkumar. A dynamic and adaptive quiescence detection algorithm. Technical Report 93-11, Parallel Programming Laboratory, Department of Computer Science , University of Illinois, Urbana-Champaign, 1993.

[109] Parthasarathy Ramachandran and L. V. Kalé. Web-based interaction and monitoring for parallel programs. Technical Report 99-05, Parallel Programming Laboratory, Department of Computer Science, University of Illinois at Urbana-Champaign, 1999.

[110] Robert K. Brunner Shobana Radhakrishnan and Laxmikant V. Kalé. Branch and bound based load balancing for parallel applications. Technical Report 99-06, Parallel Programming Laboratory, Department of Computer Science, University of Illinois at Urbana-Champaign, May 1999. Submitted for publication.

[111] Robert K. Brunner and Laxmikant V. Kalé. Handling application-induced load imbalance using parallel objects. Technical Report 99-03, Parallel Programming Laboratory, Department of Computer Science, University of Illinois at Urbana-Champaign, May 1999. Submitted for publication.

[112] L. V. Kalé and W. Shu. The Chare Kernel base language: Preliminary performance results. In ICPP89 [149], pages 118–121.

[113] W. W. Shu and L. V. Kalé. Chare Kernel - a runtime support system for parallel computations. *Journal of Parallel and Distributed Computing*, 11:198–211, 1990.

[114] L.V. Kalé. The Chare Kernel parallel programming language and system. In *Proceedings of the International Conference on Parallel Processing*, volume II, pages 17–25, August 1990.

[115] L. V. Kalé and D. S. Warren. A class of architectures for Prolog machine. In *Proceedings of the Conference on Logic Programming*, pages 171–182, Uppsala, Sweden, July 1984.

[116] A. Gursoy and L.V. Kale. Tolerating latency with dagger. In *Proceedings of the Eigth International Symposium on Computer and Information Sciences*, Istanbul, Turkey, November 1993.

[117] W. W. Shu and L. V. Kalé. A dynamic load balancing strategy for the Chare Kernel system. In Supercomputing89 [148], pages 389–398.

[118] John A. Board Jr., Laxmikant V. Kale, Klaus Schulten, Robert D. Skeel, , and Tamar Schlick. Modeling biomolecules: Larger scales, longer durations. *IEEE Computational Science & Engineering*, 1:19–30, Winter 1994.

[119] M. Nelson, W. Humprey A. Gursoy, A. Dalke, L.V. Kalé, R. Skeel, K. Schulten, and R. Kuffrin. MDScope-A visual Computing Environment for Structural Biology. *Computer Physics Communications*, 91:111–134, October 1995.

[120] Parthasarathy Ramachandran and L. V. Kalé. Mulitlingual debugging support for data-driven and thread-based parallel languages. Technical Report 99-04, Parallel Programming Laboratory, Department of Computer Science, University of Illinois at Urbana-Champaign, August 1999. To appear in the Proc. of 12th International Workshop on Languages and Compilers for Parallel Computing (LCPC '99).

[121] L. V. Kalé. A brief perspective on parallel programming. In *IEEE Tencon Regional Conference (Invited)*, pages 1085–1088, Bombay, India, November 1989.

[122] B. Ramkumar and L. V. Kalé. A Chare Kernel implementation of a parallel Prolog compiler. In *Proceedings of the Second Conference on Principles and Practice of Parallel Programming*, pages 99–108, Seattle, WA, March 1990.

[123] L.V. Kalé and Sanjeev Krishnan. Charm++ : A portable concurrent object oriented system based on C++. In *Proceedings of the Conference on Object Oriented Programmi ng Systems, Languages and Applications*, September 1993.

[124] Tamar Schlick, Robert Skeel, Axel Brünger, Laxmikant Kalé, John A. Board Jr., Jan Hermans, and Klaus Schulten. Algorithmic challenges in computational molecular biophysics. *Journal of Computational Physics*, 151:9–48, 1999.

[125] L. V. Kalé, D. A. Padua, and D. C. Sehr. OR parallel execution of Prolog programs with side effects. *The Journal of Supercomputing*, 2(2):209–223, October 1988.

[126] L.V. Kalé. The REDUCE OR process model for parallel execution of logic programs. *Journal of Logic Programming*, 11(1):55–84, July 1991.

[127] B. Ramkumar and L.V. Kalé. A join algorithm for combining AND parallel solutions in AND/OR parallel systems. *International Journal of Parallel Processing*, 1992.

[128] B. Ramkumar and L.V. Kalé. Machine independent AND and OR parallel execution of logic programs: Part I - the binding environment. *IEEE Transactions on Parallel and Distributed Systems*, 5(2):170–180, February 1994.

[129] B. Ramkumar and L.V. Kalé. Machine independent AND and OR parallel execution of logic programs: Part II - compiled execution. *IEEE Transactions on Parallel and Distributed Systems*, 5(2):181–192, February 1994.

[130] L. V. Kalé. 'Completeness' and 'Full parallelism' of parallel logic programming schemes. In *Proceedings of 1987 Symposium on Logic Programming*, pages 125–133, San Francisco, CA, 1987.

[131] L. V. Kalé. Parallel execution of logic programs: the REDUCE-OR process model. In *Proceedings of Fourth International Conference on Logic Programming*, pages 616–632, May 1987.

[132] L. V. Kalé, B. Ramkumar, and W. Shu. A memory independent binding environment for AND and OR parallel execution of logic programs. In *The Joint International Conference/Symposium on Logic Programming*, pages 1223–1240, Seattle, WA, 1988.

[133] L. V. Kalé. A tree representation for parallel problem solving. In *Proceedings of AAAI*, pages 677–681, August 1988.

[134] M. Gooley, L. V. Kalé, D. Padua, D. Sehr, B. Ramkumar, U. S. Reddy, W. Shu, and B. Wah. Prolog research at University of Illinois. In *Compcon*, pages 68–73, San Francisco, Feb 1988.

[135] B. Ramkumar and L. V. Kalé. Compiled execution of the Reduce-Or process model on multiprocessors. In *The North American Conference on Logic Programming*, pages 313–331, Oct 1989.

[136] B. Ramkumar and L. V. Kalé. An abstract machine for the reduce or process model for parallel Prolog. In *Proceedings of Knowledge Based Computer Systems Conference*, pages 267–276, Bombay, India, December 1989.

[137] L. V. Kalé, B. Ramkumar, and W. Shu. Parallel Prolog on Intel's iPSC/2. In *Proceedings of the Fourth Conference on Hypercube Concurrent Computers and Applications*, March 1989.

[138] L. V. Kalé and B. Ramkumar. Joining AND-parallel solutions in AND/OR parallel systems. In *Proceedings of the North American Conference on Logic Programming*, pages 614–631, October 1990.

[139] L.V. Kalé and B. Ramkumar. Implementing a parallel Prolog interpreter on multiprocessors. In *5th International Parallel Processing Symposium*, pages 543–548, Anaheim, CA, April 1991.

[140] David C. Sehr, Laxmikant V. Kalé, and David A. Padua. Fortran-style transformations for functional programs (Extended abstract). In ICPP91 [147], pages 282–283.

[141] L. V. Kalé. An almost perfect heuristic for the N-queens problem. *Information Processing Letters*, 34(4):173–178, April 1990.

[142] L. V. Kalé and V. Saletore. Parallel state-space search for a first solution with consistent linear speedups. *International Journal of Parallel Programming*, 19(4):251–293, 1990.

[143] D. S. Warren, M. Ahamad, S. K. Debray, and L. V. Kalé. Executing distributed prolog programs on a broadcast network. In *Proceedings of the Logic Programming Symposium*, pages 12–21, Atlantic City, February 1984.

[144] V. Saletore and L. V. Kalé. Obtaining first solutions fast in parallel problem solving. In *The North American Conference on Logic Programming*, pages 390–408, Oct 1989.

[145] V. Saletore and L.V. Kalé. Consistent linear speedups for a first solution in parallel state-space search. In *Proceedings of the AAAI*, pages 227–233, August 1990.

[146] L.V. Kalé and Vikram Saletore. Efficient parallel execution of IDA* on shared and distributed memory multiprocessors. In *Proceedings of the Sixth Distributed Memory Computing Conference (DMCC6)*, Portland, Oregon, April 1991.

[147] *Proceedings of the 1991 International Conference on Parallel Processing*, St. Charles, IL, August 1991.

[148] *Proceedings of Supercomputing '89*, November 1989.

[149] *Proceedings of the 1989 International Conference on Parallel Processing*, St. Charles, IL, August 1989.

Software developed and distributed via the web

- Charm++ : parallel object language, as a C++ library
- Converse: Portable parallel runtime framework:
- NAMD: Parallel Molecular Dynamics program (available from : http://ks.uiuc.edu).
- Master-slave library, with a sophisticated load balancer.
- mdPerl: parallel Perl based on Converse
- Tempo: message passing with user level threads.
- Generic branch-and-bound : A C++ class library that supports template based implementation of the parallel branch and bound algorithm, on top of Charm++ with prioritized load balancing.
- A "standard library" of parallel components, built on top of Converse, and Charm++; In progress.

  All the above software, except NAMD, is distributed from http://charm.cs.uiuc.edu.