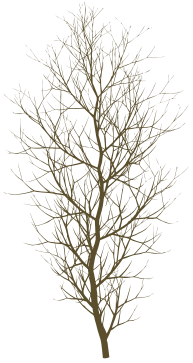
Isaac Dooley



393 Degraw Street # 1
Brooklyn, NY 11231

217-419-3738
isaac@isaacdooley.com
http://www.isaacdooley.com

Profile

Isaac Dooley was a Department of Energy High Performance Computer Science Fellow who received his Ph.D. Degree from the University of Illinois at Urbana-Champaign in Computer science and his B.S. in Mathematics and Computer Science from Birmingham-Southern College. He joined Two Sigma Investments after completing his Ph.D. in 2010.

Work Experience

2012-present	Manager. Two Sigma Investments
2010-2012	Software Engineer. Two Sigma Investments
2009-2010	Graduate Research Assistant. Parallel Programming Lab. University of Illinois
2005-2009	Department of Energy High Performance Computer Science Fellow
2007	Summer Practicum. Lawrence Livermore National Lab
2006	Summer Practicum. Sandia National Lab
2004-2005	Graduate Research Assistant. Parallel Programming Lab. University of Illinois
1998-2004	Co-founder, Sysadmin, Webmaster. ALtruis LLC.

Education

2004-2010	Ph.D.	**University of Illinois (at Urbana-Champaign)**	3.92 GPA
2004-2006	M.S.	**University of Illinois (at Urbana-Champaign)**	
2000-2004	B.S.	**Birmingham-Southern College** (BSC)	3.95 GPA
	Majors: Mathematics, Computer Science		

Standardized Test Scores

score	*out of*	
800	*800*	GRE Quantitative
6.0	*6.0*	GRE Analytical Writing
530	*800*	GRE Verbal
780	*800*	GRE Subject Test: Computer Science
800	*800*	SAT II Subject Test: Math IIC
800	*800*	SAT II Subject Test: Chemistry
780	*800*	SAT II Subject Test: Writing
800	*800*	SAT Math
630	*800*	SAT Verbal

Awards

2005-2009	Department of Energy *High Performance Computer Science Fellowship*
2004	*Faculty Outstanding Mathematics Senior Award*
2003-2004	*Vail Research Fellow*
2003	Recipient of *The Acton Award in Mathematics*
2002	1st place ACM 2002 Regional Computer Programming Competition (Division 2)
2000-2004	*Guy E. Snavely Scholarship*
1997-2000	National Math Team Champion in High School
2000	Voted Most Intellectual in High School
1999	Eagle Scout, Boy Scouts of America

Elected Positions

2003-2004	President of BSC Association for Computing Machinery chapter
2002-2004	President of BSC Intervarsity Christian Fellowship chapter
2002-2003	Vice President of BSC Association for Computing Machinery chapter
2002-2003	Vice President of BSC Kappa Mu Epsilon chapter (Math Honorary)

Computer Skills

Proficient in:

Java, C *(self taught from age 10)*, MPI, Charm++, Mac OSX, Linux

Working knowledge of *(used in the past few years)*:

C++, Fortran, Matlab, Perl, PHP, HTML, CSS, Bash, SQL, Posix Threads, Java Threads, CUDA, CVS, SVN, Git, TCP/IP, Ethernet, DNS, Eclipse, LaTeX

Familiar with:

Ruby, OpenMP, Windows, Photoshop, Dreamweaver, Apache

Peer Reviewed Journal Publications

Orion Lawlor, Sayantan Chakravorty, Terry Wilmarth, Nilesh Choudhury, Isaac Dooley, Gengbin Zheng, and Laxmikant Kale **ParFUM: A Parallel Framework for Unstructured Meshes for Scalable Dynamic Physics Applications** in *Engineering with Computers, 22:215–235, September 2006*

Isaac Dooley and Sandhya Mangala and Laxmikant Kale and Philippe Geubelle **Parallel Simulations of Dynamic Fracture Using Extrinsic Cohesive Elements** in *Journal of Scientific Computing, 39(1):144-165 April 2009*

Isaac Dooley and Laxmikant V. Kale **Detecting and Using Critical Paths at Runtime in Message Driven Parallel Programs** under review for *International Journal of Networking and Computing*

Isaac Dooley, Jonathan Lifflander, Laxmikant Kale **Performance Steering of Parallel Applications with Control Points** to be submitted to *Journal of Scientific Computing*

Peer Reviewed Conference Publications

Isaac Dooley, Chee Wai Lee, and Laxmikant V. Kale **Continuous Performance Monitoring for Large-Scale Parallel Applications** *16th Annual International Conference on High Performance Computing (HiPC 2009)*

Isaac Dooley, Chao Mei, Jonathan Lifflander, Laxmikant V. Kale **A Study of Memory-Aware Scheduling in Message Driven Parallel Programs** *17th Annual International Conference on High Performance Computing (HiPC 2010)*

Gengbin Zheng, Gagan Gupta, Eric Bohm, Isaac Dooley, and Laxmikant V. Kale **Simulating Large Scale Parallel Applications using Statistical Models for Sequential Execution Blocks** *The IEEE Sixteenth International Conference on Parallel and Distributed Systems (ICPADS 2010)*

Peer Reviewed Workshop & Symposia Publications

Orion Lawlor, Hari Govind, Isaac Dooley, Michael Breitenfeld, and Laxmikant Kale **Performance Degradation in the Presence of Subnormal Floating-Point Values** *OSIHPA Workshop at PACT05, September 2005*

Isaac Dooley and Laxmikant Kale **Quantifying the Interference Caused by Subnormal Floating-Point Values** *OSIHPA Workshop at PACT06, September 2006*

Isaac Dooley, Chao Mei, Laxmikant V. Kale **NoiseMiner: An Algorithm for Scalable Automatic Computational Noise and Software Interference Detection** in *Proceedings of HIPS Workshop at IEEE International Parallel and Distributed Processing Symposium 2008*

Aaron Becker, Isaac Dooley, Laxmikant Kale **Flexible Hardware Mapping for Finite Element Simulations on Hybrid CPU / GPU Clusters** *Symposium on Application Accelerators in HPC, 2009*

Isaac Dooley and Laxmikant V. Kale **Detecting and Using Critical Paths at Runtime in Message Driven Parallel Programs** *12th Workshop on Advances in Parallel and Distributed Computing Models (APDCM) at IPDPS 2010*

Technical Reports

Isaac Dooley and Laxmikant V. Kale **Control Points for Adaptive Parallel Performance Tuning** *PPL Technical Report 2008*

Selected Software Engineering Experience

I rewrote and re-factored large portions of a parallel performance visualization tool named *Projections*. I added multi-threaded file reading and processing, improved various graphical displays, improved its ease of use, added new performance analysis tools including one that streamed data live from a remote supercomputer, added support for compressed input file formats, fixed many bugs, and improved the code base through eliminating unused code and reducing visibility of class members. My changes allowed users to analyze significantly larger sets of trace log data than was ever possible before! Now users can easily visualize data from over 4000 processors, whereas before only 64 could be viewed at once. *Projections* consists of over 42000 lines of java code as well as trace generation code written in C++ that is included in the Charm++ runtime system. The tool is routinely used by Charm++ application developers to analyze the performance of their programs.

As part of my dissertation work, I created an online automatic tuning framework within the Charm++ runtime system. This tuning framework can dynamically reconfigure a running parallel program. The tuning framework is unique in that it measures performance characteristics of the running parallel program including information about time processors spend idle, a measure of overhead costs, memory usage, and critical path profiles. The measurements are analyzed by the tuning framework, and tunable parameters in the user program are adjusted in an attempt to fix any observed performance deficiencies. This tuning framework consists of more than 6000 lines of C++ code so far and is still under development.

I rewrote and debugged a communication optimization library named *Comlib*, which provides numerous implementations of collective communication operations to Charm++ programs. Within the library, I added numerous new *array section multicasts*. These multicasts send a message to a list of objects distributed across many processors. I implemented 7 new multicast algorithm variants (each with different spanning trees). These new multicast algorithms led to significant speedups in a new LU Matrix Factorization program that I helped develop. The resulting performance of the LU program exceeds the performance of a common reference implementation named HPL for some matrix sizes. Comlib consists of over 18700 lines of C and C++ code.

Of the developers of the Charm++ system, I am the 2nd most prolific contributor in the past 6 years, as counted by total number of commits to the Charm++ source code repository (this includes Comlib and my tuning framework, but not the Projections client).