

# Research Statement

David K. Lowenthal  
Department of Computer Science, The University of Georgia  
Athens, GA 30602-7404

My research spans the areas of parallel and distributed computing, networks, and operating systems. The common thread throughout my work is the design, implementation, and evaluation of software systems to solve fundamental problems in these three areas—with little or no involvement from users or programmers. The research described below has been funded through two grants from the National Science Foundation.

## Parallel and Distributed Computing

I have focused much of my work in parallel and distributed computing on addressing the large and growing amount of energy consumed by supercomputer centers. Such centers are housing ever-larger clusters—tens and even hundreds of thousands of nodes are becoming common. Such excess is at odds with the knowledge that power consumption is costly in several ways, including lower reliability, higher cost, and higher cooling requirements. While large-scale clusters such as IBM's BlueGene/L solve the problem by using a large number of low-power nodes, the nodes have only a single voltage level for the CPU. This is suboptimal for most applications, because they do not fully utilize the CPU—in particular, they are not CPU bound.

My work has centered on *power-scalable* clusters, which are clusters that consist of nodes that are capable of running at several different frequency/voltage pairs. Because power is proportional to the frequency multiplied by the square of the voltage, significant power can be saved by executing the CPU at lower frequencies. For memory, communication, or I/O-bound programs, the overall program slowdown due to executing at a lower frequency will be much smaller than the slowdown of the CPU itself. As parallel programs are well known to be inefficient<sup>1</sup>, there is plenty of opportunity to reduce power/energy consumption.

My first result in this area quantified the potential energy savings on power-scalable clusters on the NAS benchmark suite [FLS<sup>+</sup>05]. This paper showed that a simple, easily obtainable metric called operations per cache miss can predict energy savings and execution time increase. This work was extended, focusing on the use of energy-delay based metrics for evaluation [FLP<sup>+</sup>06]. The next paper showed that significant gain was possible if programs are partitioned into phases, with each phase executing in a potentially different frequency [FPLK05]. Additionally, I showed that load imbalance can be leveraged to reduce the frequency of nodes that arrive early at a global synchronization point [KFL05]; this work shows that up to 12% of overall system energy can be saved with no programmer assistance. I also addressed the problem of choosing the proper configuration of node/frequency combinations to minimize execution time given a cluster energy limit [SLRF06]. Results show that the choice of configuration, which is automatically selected by a low-overhead, iterative technique with a novel execution model for execution time and consumed energy, results in near-optimal execution time. Next, I designed and implemented an algorithm to detect communication phases and then perform voltage scaling during such phases [LFL06] Finally, I have abstracted the problem of saving as much energy as possible given a fixed execution time bound into a graph-theoretic problem and then solved it with a unique linear-programming formulation [RLFF06].

---

<sup>1</sup>For example, the Gordon Bell prize is given each year to the world-champion parallel program. In 2005, the winning program from LLNL was less than 80% efficient, which means that 20% of the time, the program is wasting energy.

My other projects in parallel and distributed computing have focused primarily on supporting the most common kind of parallel platform—a cluster of commodity workstations that are neither homogeneous nor dedicated to any particular task. Most academic and industrial organizations have at least one such cluster. I have developed a specialized version of MPI called Dyn-MPI [WLN03], which automatically balances the computational load on a non dedicated cluster. Results show that programs that use Dyn-MPI execute efficiently in non dedicated environments, including up to almost a three-fold improvement compared to programs that do not redistribute data and a 25% improvement over standard adaptive load balancing techniques. In addition, I developed MHETA [NLZ05], which is an execution model for heterogeneous architectures that takes into account computation, communication, and I/O. The use of MHETA results in data distributions that execute up to four times faster than naive distributions. Finally, one of the applications used to test MHETA involved a novel, out-of-core implementation of an algorithm to predict RNA secondary structure [ZL06].

Finally, I have worked on a project to improve collective communication in MPI programs through specialization [KYL03, FYL06]. The basic idea is to develop topology-aware collective communication algorithms as well as to develop infrastructure to choose the best such algorithm on the fly. Results show improvements on the NAS benchmark suite of up to a factor of almost three.

Current and future work is focused on addressing different but related high-performance, power-aware computing issues (for example, with the emerging class of dual-core machines). In addition, I am investigating locating and addressing scalability problems in high-performance parallel programs.

## Networking

Similar to parallel and distributed computing, much of my work in networking has been focused on conserving energy. Here, our challenge is to save energy in the wireless network interface card (WNIC) on mobile computers such as laptops and palmtops. This is important because the WNIC is a significant energy consumer in such computers, so battery life can be greatly extended if WNIC energy is saved.

Applications such as file downloads typically require the WNIC to remain in *idle*, *receive*, or *transmit* mode, all of which use significant amounts of energy compared to low-energy *sleep* mode. Remaining in high-energy mode is the most effective way to minimize transfer time, but in a mobile environment, users may accept a modest increase in transfer time in exchange for significant energy savings. In fact, IEEE 802.11b power saving mode (PSM) is designed specifically for this purpose; however, it can cause a large performance degradation.

I have designed and implemented a technique for TCP downloads in mobile clients that saves energy with a small time increase. The basic idea is to (1) shape traffic on the client (without *any* server modification), which forces data to arrive in bursts so that idle periods are aggregated together and then (2) predict when bursts will arrive so that the wireless network interface card can be transitioned to low-power mode during idle periods. The technique is fully compatible with TCP.

The first result [YKW<sup>+</sup>04] showed that our client-centered (CC) approach outperforms baseline TCP by up to 64% when comparing energy-delay product, with an average improvement of 19% over seven Internet sites. Note that these are real tests to real Internet servers—the servers are unaware of what the client is doing—showing the potential for widespread deployment. I additionally showed that the client can “trick” the server into providing its congestion window size, which a TCP server will otherwise not do [YKW<sup>+</sup>06]. This allowed a significant energy savings—up to 90% over all clients—when several clients share an access point. Also, I implemented a version of CC aimed at small downloads (e.g., web pages); results showed an energy savings of 21% compared to PSM [YKWL04]. Finally, I developed an active, client-directed technique that integrates the ideas of CC and PSM to handle situations where there is higher variance in the network [YLL05].

This project also produced spin-off work. First, I described techniques to infer the round-trip time of a connection on any point on a path [VLL05]. Additionally, I developed TCP-RC, a receiver-centered protocol that achieves low latency [MLWL05].

Current and future work is focused on generalizing the ideas of modifying the local end of the TCP connection to achieve wide deployment. We are developing an flexible, asymmetric, TCP-compatible protocol, called ETCP, that can support a wide variety of network applications.

## Operating Systems

In the general area of operating systems, I have focused on solving protection-based problems that arise in programming languages and compilers. Here, I developed a new way to address one of the long standing issues in programming languages and compilers—checking array bounds to ensure program correctness. Instead of either (1) using the simple solution of generating bounds checks for each array reference or (2) relying on compiler optimizations to eliminate such checks, which is known to be hard, I developed a new OS abstraction.

This abstraction is called an *index confinement region*, or ICR; essentially, an ICR is an isolated virtual memory region, of which only a small portion, corresponding to valid array data, is mapped and permissible to access. The rest of the ICR is unmapped, and any access to that portion will cause a hardware protection fault. This achieves implicit bounds checking for all array references. While ICRs can be implemented on most modern architecture/operating systems, they are primarily intended for 64-bit machines.

I described the application of ICRs to support high-performance computing in Java [BWLR04], a language that requires array references to be within bounds. Results showed that using ICRs reduces the bounds checking overhead (relative to performing no checks) for scientific Java benchmarks from an average of 63% to an average of only 9%. This work was extended to discuss how to adapt ICRs to support programs written in C [BWLR06].

Current and future work is focused on addressing other kinds of protection issues using OS mechanisms. In addition, I am developing OS-level mechanisms to support ETCP.

## References

- [BWLR04] Chris Bentley, Scott A. Watterson, David K. Lowenthal, and Barry Rountree. Implicit java array bounds checking on 64-bit architectures. In *ACM International Conference on Supercomputing*, June 2004.
- [BWLR06] Chris Bentley, Scott A. Watterson, David K. Lowenthal, and Barry Rountree. Implicit array bounds checking on 64-bit architectures. *ACM Transactions on Architecture and Code Optimization (to appear, subject to minor revisions)*, 2006.
- [FLP<sup>+</sup>06] Vincent W. Freeh, David K. Lowenthal, Feng Pan, Robert Springer, , Nandini Kappiah, Barry Rountree, and Mark E. Femal. Analyzing the energy-time tradeoff in high-performance computing applications. *IEEE Transactions on Parallel and Distributed Systems, (to appear, subject to minor revisions)*, 2006.
- [FLS<sup>+</sup>05] Vincent W. Freeh, David K. Lowenthal, Rob Springer, Feng Pan, and Nandani Kappiah. Exploring the energy-time tradeoff in MPI programs on a power-scalable cluster. In *IEEE International Parallel and Distributed Processing Symposium*, April 2005.

- [FPLK05] Vincent W. Freeh, Feng Pan, David K. Lowenthal, and Nandani Kappiah. Using multiple energy gears in MPI programs on a power-scalable cluster. In *ACM Symposium on Principles and Practices of Parallel Processing*, June 2005.
- [FYL06] Ahmad Faraj, Xin Yuan, and David K. Lowenthal. STAR-MPI: Self tuned adaptive routines for MPI collective operations. In *ACM International Conference on Supercomputing*, June 2006.
- [KFL05] Nandani Kappiah, Vincent W. Freeh, and David K. Lowenthal. Just in time dynamic voltage scaling: Exploiting inter-node slack to save energy in MPI programs. In *Supercomputing 2005*, November 2005.
- [KYL03] Amit Karwande, Xin Yuan, and David K. Lowenthal. CC-MPI: A compiled communication capable MPI prototype for ethernet switched clusters. In *ACM Symposium on Principles and Practice of Parallel Programming*, June 2003.
- [LFL06] Min Yeol Lim, Vincent W. Freeh, and David K. Lowenthal. Adaptive, transparent frequency and voltage scaling of communication phases in MPI programs. In *Supercomputing 2006 (to appear)*, November 2006.
- [MLWL05] Doug McCreary, Kang Li, Scott A. Watterson, and David K. Lowenthal. TCP-RC: a receiver-centered TCP protocol for delay-sensitive applications. In *Multimedia Computing and Networking*, January 2005.
- [NLZ05] Mario Nakazawa, David K. Lowenthal, and Wenduo Zhou. The MHETA execution model for heterogeneous clusters. In *Supercomputing 2005*, November 2005.
- [RLFF06] Barry Rountree, David K. Lowenthal, Shelby Funk, and Vincent W. Freeh. Per-task cpu frequency selection for near-optimal energy savings in MPI programs (manuscript). March 2006.
- [SLRF06] Robert C. Springer IV, David K. Lowenthal, Barry Rountree, and Vincent W. Freeh. Minimizing execution time in MPI programs on an energy-constrained, power-scalable cluster. In *ACM Symposium on Principles and Practice of Parallel Programming*, March 2006.
- [VLL05] Bryan Veal, Kang Li, and David K. Lowenthal. New methods for passive estimation of TCP round-trip times. In *Workshop on Passive and Active Measurement*, March 2005.
- [WLN03] D. Brent Weatherly, David K. Lowenthal, Mario Nakazawa, and Franklin Lowenthal. Dyn-MPI: Supporting MPI on non dedicated clusters. In *Supercomputing 2003*, November 2003.
- [YKW<sup>+</sup>04] Haijin Yan, Rupa Krishnan, Scott A. Watterson, David K. Lowenthal, Kang Li, and Larry L. Peterson. Client-centered energy and delay analysis for TCP downloads. In *IEEE International Workshop on Quality of Service*, June 2004.
- [YKW<sup>+</sup>06] Haijin Yan, Rupa Krishnan, Scott A. Watterson, David K. Lowenthal, Kang Li, and Larry L. Peterson. Client-centered, energy-efficient wireless communication on IEEE 802.11b networks. *IEEE Transactions on Mobile Computing (to appear)*, 2006.
- [YKWL04] Haijin Yan, Rupa Krishnan, Scott A. Watterson, and David K. Lowenthal. Client-centered energy savings for concurrent HTTP connections. In *ACM Workshop on Networks and Operating System Support for Digital Audio and Video*, June 2004.
- [YLL05] Haijin Yan, David K. Lowenthal, and Kang Li. Ace: An active, client-directed technique for reducing WNIC energy during web browsing. In *ACM Workshop on Networks and Operating System Support for Digital Audio and Video*, June 2005.

- [ZL06] Wenduo Zhou and David K. Lowenthal. A parallel, out-of-core algorithm for RNA secondary structure prediction. In *IEEE International Conference on Parallel Processing, (to appear)*, August 2006.