FOREWORD

This file presents the current research interests of the faculty members of the Carnegie Mellon University Doctoral Program in Computer Science, along with those of associated faculty of other departments. Each person listed has written his or her own section of the Guide. There has been no attempt to eliminate redundancy by combining descriptions of related work. In addition to the research descriptions, indices at the end of the document list the faculty by interest keywords.

The primary purpose of the Guide is to acquaint new and prospective members of our community, especially doctoral students, with the on-going research and with the faculty involved. We also use the Guide to inform outside colleagues and visitors, and to direct those looking for persons knowledgeable about any particular topics.
UMUT ACAR
Assistant Professor, Computer Science

My main areas of interest are programming languages, parallel computing, and algorithms. In my research, I aim to raise the level of abstraction at which computer scientists reason about problems, and develop algorithms and software. To this end, I develop abstractions and design the supporting language constructs, algorithms, and software systems.

Programming languages. Programming languages aim to fill the large gap between the level at which we humans reason (as for example manifested by mathematics) and the tedious code of instructions required by the computer. They do so by offering us abstractions with which we can organize and express our thoughts and by translating our thoughts expressed as programs to code suitable for computers to execute. For reasons of efficiency, as computer scientists, we have thus far resorted to low level abstractions--abstractions closer to the level of computers than humans--for expressing computation. But today, as computer systems become architecturally more complex, it has become increasingly more difficult to design software that perform well with such low-level abstractions. The problem is exacerbated by increased demands on the capability and the quality of software, as it performs many critical tasks and handles sensitive information. I therefore develop higher-level abstractions, programming languages, and systems that enable creative thought and expression while also ensuring efficiency. My research in this area thus far focused on dynamic or incremental computation, where systems interact with dynamically changing data, and parallel computation, where multiple processors can be used to perform a task simultaneously.

Parallel computing. The turn of the 21st century may be remembered as a momentous point in the history of computing as the single-chip, multiple-processor (multicore) computer started replacing the sequential computer, the mainstay of computing until then. Unfortunately, many of today's programming languages, algorithms, and software systems are not suitable for use with parallel hardware. To take advantage of parallelism, we need new programming languages, algorithms, and software systems. Specific problems to be tackled include reliability (a.k.a., fault tolerance or resilience), scheduling for scalable efficiency, and control of communication costs between processors and memory. In order to keep the level of abstraction high without sacrificing performance, I work on these problems by using a broad methodology that combines techniques and tools from several areas including algorithms, programming languages, and systems.

Algorithms. By allowing us to reason accurately about the cost (resource usage) of computation, classic models of computation, e.g., the RAM and the PRAM models, have enabled us to design efficient algorithms for many important problems. Many algorithms, (e.g., dynamic and parallel algorithms), however, can be difficult to implement and use in practice because they require implementations to maintain complex invariants expressed in terms of the details of the machine hardware (e.g., memory layout). This theory-practice gap increases as the complexity of the hardware (e.g., non-uniform memory) and the problems that we face increase. I wish to close this gap by inventing realistic computational models that are higher-level and that simplify the design, analysis, and expression (and thus implementation) of sophisticated algorithms by eliminating hardware-specific details from algorithms. The challenge is to ensure that such algorithms can remain efficient. In the near future, I am particularly interested in models and algorithms for dynamic and parallel problems.

Understanding software. In the current state of the art, our ability to design software far exceeds our ability to understand its behavior. For example, we may spend a long time (e.g., days) to understand a small piece of code that we wrote in a fraction of the time (e.g., minutes). I am interested in developing techniques for understanding software. To this end, I am pursing the idea of enabling a conversation between the user and software, where the user queries the software and the software responds by "explaining" its work.

http://www.cs.cmu.edu/~umut
DAVID ANDERSEN
Associate Professor, Computer Science

My research focuses on networks and distributed systems. My interests lie in creating systems that meet goals such as robustness, high availability, and energy efficiency. My current research encompasses two large project areas:

*FAWN: Fast Arrays of Wimpy Nodes.* The FAWN project aims to develop computational cluster architectures, and software techniques to use them, that are drastically more energy and cost-effective than today's technologies. We do so by building clusters from systems that are comparatively slow by the standards of today's leading-edge, but that together, can provide drastically more throughput and computational capability at lower power. We then tackle the problems of actually using such systems by developing new algorithms and systems techniques for data-intensive processing on these clusters. Typical challenges we seek to overcome are using substantially less memory than prior approaches, harnessing new storage technologies such as Flash and phase-change memory, and coping with systems composed of 5x more nodes than previously used.

*XIA: The Expressive Internet Architecture.* XIA is a clean-slate approach to Internetworking, in a collaboration with numerous faculty at Carnegie Mellon and elsewhere. The goal of the project is to develop an Internet architecture for the next 100 years: one that is robust, has drastically improved security compared to todays, and, most importantly, one that can evolve easily to meet as-yet-unknown uses and challenges.

MARIA FLORINA BALCAN
Associate Professor, Computer Science, Machine Learning

My research tackles fundamental questions in Machine Learning, Algorithmic Game Theory, and Algorithms. My work develops deep new connections between these areas, using ideas and insights from each of them to solve some of their central and emerging challenges in innovative ways.

Foundations for Machine Learning Machine learning studies the design of automatic methods for extracting information from data and has become a tremendously successful discipline with a wide variety of important applications in areas such as robotics, healthcare, information retrieval, and sustainability. Its past successful evolution was heavily influenced by mathematical foundations developed for several core problems including generalizing from labeled data. However, with the variety of applications of machine learning across science, engineering, and computing in the age of Big Data, re-examining the underlying foundations of the field has become imperative. A major goal of my research is to substantially advance the field of machine learning by developing foundations and algorithms for a number of important modern learning paradigms. These include interactive learning, where the algorithm and the domain expert engage in a dialogue to facilitate more accurate learning from less data compared to the classic approach of passively observing labeled data; distributed learning, where a large dataset is distributed across multiple servers and the challenge lies in learning with limited communication; and multi-task learning, where the goal is to solve multiple related learning problems from less data by taking advantage of relationship among the learning tasks. My goal is to provide new frameworks explaining the fundamental underlying principles, as well as new powerful, principled, and practical learning algorithms designed to satisfy the new types of constraints and challenges of these modern settings (including statistical efficiency, computational efficiency, noise tolerance, limited supervision or interaction, privacy, low communication, and incentives).

Algorithmic Game Theory Traditionally, complex systems involving multiple agents each with their own interests in mind have been analyzed through purely game theoretic lenses, but technologies such as the Internet have triggered an increased growth of research concerning algorithmic aspects as well. Yet these approaches are often limited to studying static concepts. My work goes further and shows how machine learning methods can help tackle fundamental open questions regarding
information-gathering and dynamics in these settings. For example, in past work, I showed an exciting application of machine learning to automate aspects of auction design and formally address problems of market analysis for designing combinatorial pricing mechanisms with near-optimal revenue guarantees. Along different lines, my current work develops a new approach to analyzing the overall behavior of complex systems in which multiple agents with limited information are selfishly adapting their behavior over time based on past experience. My goal is to develop general techniques for influencing the behavior of natural learning dynamics towards globally good states, as well as to provide powerful tools to reason about economic agents as adaptive, learning entities.

**Analysis of Algorithms beyond the Worst Case**  Many important optimization problems are unfortunately provably hard even to approximate well on worst-case instances. However, real-world instances often satisfy certain natural regularities or stability properties. A recent direction in my work is designing algorithms for important optimization problems with strong formal guarantees under natural stability assumptions about the input instances. For example, in the context of clustering I showed that approximation stability assumptions (implicit when modeling clustering as approximately optimizing a distance-based objective, e.g., k-means) could be leveraged to overcome worst-case hardness results. I am interested to further analyze in this framework other problems of finding hidden structure in data. I additionally plan to identify other meaningful and generally applicable models of computation beyond worst-case analysis, that accurately model real-world instances and could provide a useful alternative to traditional worst-case models in a broad range of optimization problems.

**GUY BLELLOCH**  
Professor, Computer Science

My main research interest is in the interaction between algorithms and languages, mostly in the context of parallel computing, and has consisted of both theoretical and experimental work. As programming languages become higher level, implementations become more complex, and parallelism becomes pervasive, users are naturally becoming more removed from the hardware and its costs. Rather than trying to bring programmers down to the level of the machine to understand and get good performance, however, I believe that we should be trying to bring languages and cost models up to the level of the programmer. My research therefore centers around questions of how to model costs (e.g. time and space) for very-high level programming constructs (e.g. dynamic parallelism, futures, garbage collection), of how to design systems so these costs have meaning, and of how to make use of these features in effective algorithms design.

My recent work includes work on the PSCICO project with Gary Miller, Bob Harper and Peter Lee. Here we are looking at how to use very-high level programming constructs in geometric and scientific algorithms. We hope this project will give guidance to future language design, and will identify new ways of thinking about algorithm implementation. I also work on applied algorithms, parallel garbage collection, parallel scheduling, efficient parallel algorithms, and continue to work, to some extent, on the NESL programming language, a parallel language that my students and I developed in the early 90s.
My main research interests are in machine learning theory, algorithmic game theory, mechanism design, and approximation algorithms, as well as topics that combine several of these areas.

**Machine Learning Theory.** The goals of machine learning theory are to provide a mathematical understanding of the issues involved in getting programs to learn from experience. This involves designing models that capture fundamental tradeoffs, as well as producing new analyzable algorithms. One of my recent interests has been in developing an analog of the standard PAC learning model for the problem of clustering. In clustering, you have a set of data (say, documents) that you want to cluster according to some criterion (say, by topic). To aid in this task, you assume you have a pairwise similarity measure among data objects (perhaps measuring the fraction of important words that two documents have in common) that you hope is related to your criterion for clustering. The question we are interested in is: what properties of this similarity measure (in terms of how it relates to the clustering criterion) are sufficient to allow one to cluster well, and by what algorithms? One of the interesting things we have found is that if you allow the algorithm to produce a hierarchical clustering (splitting the documents into high-level categories at the top, then recursively splitting each category) and call this successful if the user's desired clustering is close to some pruning of this tree (i.e., we make it the user's responsibility to decide how specific a clustering she wants) then one can develop a fairly general theory of natural properties that are sufficient for clustering via different kinds of algorithms.

**Algorithmic game theory and mechanism design.** Many large systems involve multiple entities, interacting with each other but with their own interests in mind. The area of algorithmic game theory has developed around understanding the behavior of systems of this type: for example, analyzing the efficiency loss caused by individuals acting in their own interests rather than acting in the best interests of the overall system. Usually, this work focuses on analyzing equilibrium behavior. One of my main research interests has been in replacing the assumption that users are in a Nash equilibrium (which sometimes can even be computationally hard to find) with the much weaker assumption that individuals are adapting their behavior in ways that minimize their own regret (something for which there are many efficient algorithms). We find that results based on this assumption are often just as good as those using stronger equilibrium assumptions, plus are often resilient to the presence of Byzantine players who act in an unpredictable and even adversarial manner.

In the area of algorithmic mechanism design, I am interested in the design of auctions and other pricing mechanisms for optimizing various quantities such as revenue and efficiency. Some of these problems can be converted into hard but deceptively simple-looking graph problems.

**Approximation algorithms.** Many important problems turn out to be NP-hard, implying that it is unlikely there will be algorithms for them that are both efficient and optimal in the worst case. One approach, then, is to find algorithms that produce approximately-optimal solutions. I am interested in a variety of problems of this sort. One problem I have worked on in the past is the following "trick-or-treater's problem": given a weighted graph (a map of a neighborhood), a starting location (your house), and a length-bound L (the distance you can travel in two hours), find the path of length at most L that visits as many different locations as possible. For this problem, we have a constant-factor approximation algorithm. However, if different locations have different deadlines (instead of deadline L for everything) then the best we know is a logarithmic-factor approximation.
LENORE BLUM  
Distinguished Career Professor, Computer Science

Complexity and Real Computation. In 1989, Steve Smale, Mike Shub and I introduced a theory of 
computation and complexity over an arbitrary ring or field R. In 1997, along with Felipe Cucker, we 
published a book, Complexity and Real Computation (Springer-Verlag). From our Introduction: "The 
classical theory of computation had its origin in the work of logicians -- of Godel, Turing, ...., among 
others -- in the 1930's. The model of computation developed in the following decades, the Turing 
machine, has been extraordinarily successful in giving the foundations and framework for theoretical 
computer science.

"The point of view of this book is that the Turing model (we call it "classical") with it's dependence on 
0's and 1's, is fundamentally inadequate for giving such a foundation for modern scientific 
computation, where most of the algorithms - with origins in Newton, Euler, Gauss, et al. - are real 
number algorithms."

Our approach applies to the analysis of algorithms over continuous domains as well as the discrete. 
The classical theory is recovered if we allow the ring R to be Z_2 (the integers mod 2). But now R can 
also be the real or complex numbers, or any other field. The familiar complexity classes P, NP and 
fundamental question "does P= NP?" make sense over R and moreover, relate explicitly to 
fundamental problems in mathematics such as Hilbert's Nullstellensatz. Thus, we are particularly 
interested in research concerning the complexity of algorithms that solve systems of polynomial 
equations.

Transfer Principles for Complexity Theory. A powerful tool of the classical theory is that of 
reduction: If problem A can be shown to be reducible to problem B, then techniques for solving B can 
be used to solve A. Classically, A and B are both discrete, i.e. defined over the same domain Z_2. 
But now we have an additional powerful tool, namely that of transfer: When there was essentially only 
one model of computation (i.e. over Z_2), it didn't make sense to transfer complexity results from one 
domain to another. But now, transfer becomes a real possibility. We can ask: Suppose we can show 
P=NP over the complex numbers (using all the mathematics that is natural here). Then can we 
conclude that P=NP over another field such as the algebraic numbers or even over Z_2? (Answer: 
Yes and essentially yes.)

I am particularly interested in such transfer results and problems that appear in the interface between 
the discrete and the continuous.

MANUEL BLUM  
Bruce Nelson University Professor, Computer Science

My current research projects include:

1.1 The CAPTCHA project. CAPTCHA is an acronym for "Completely Automatic Public Turing 
Test To Tell Computers Humans Apart." It is a kind of automatic sentry, a test that is made up, 
aministered, and graded by a computer in order to permit humans but not bots to enter the 
Garden of Eden. One of the paradoxes of this project is to show how a computer can generate 
and grade a test that it (the computer itself) cannot pass! Current Captchas do this by putting a 
randomly chosen string of characters on a rubber sheet. They then deform the sheet to the point 
that OCR (Optical Character Recognition) cannot read the characters. Humans can read the 
deformed image, just as they can read the writing on a stained and twisted piece of paper. While 
the Captcha tester itself cannot read the deformed image, it can grade the test. This is because 
it remembers its initial choice of characters.

1.2 A good sound-based Captcha is needed for blind people (the blind are stopped by visual
1.3 The Ultimate OCR (Optical Character Recognition) project. The Ultimate OCR is a program that can read anything that humans can read. ***NOTE: I do not propose to create the Ultimate OCR but to get the vision community to create it for us.*** The approach to the Ultimate OCR is to build an Ultimate Captcha that can ONLY be read by a human or by a device that can read everything that humans can read. We have ideas how to build such a device, but need someone to think these ideas through with us, make improvements, and then build the Captcha. This Ultimate Captcha would serve as a challenge to the Vision Community. As with previous challenges we have set, this challenge will almost certainly be taken seriously as it would be a maximally powerful automatic test of ability to read text. A bot that could pass this test would be able to read anything that a human can read. It's code would therefore constitute an Ultimate OCR. This work is joint with Luis von Ahn and anyone whom we accept to work on it with us.

2.1 CONSCSness: CONSCS = CONceptualizing Strategizing Control System. This project is a top-down complexity theoretic approach to understanding consciousness. The intent is to set down a very small number of axioms for CONSCSness and then prove theorems about the concept. Hopefully the theorems will make sense and teach us something about how to build a CONceptualizing Strategizing Control System. This work is joint with Ryan Williams and Brendan Juba.

2.2 Proof triangles. (Definition triangles. Algorithm triangles. etc.) One concrete consequence of our study of CONSCSness is a new view of proof that extends the traditional formal concept of proof to something that we call a "proof triangle." A proof triangle has a formal mathematical proof at the base of the triangle. The apex of the triangle is a bare minimal hint. The next level might be a more extended hint. The next a proof idea. Then a proof outline. Then several levels of informal proof, leading to the final complete formal proof. None of these levels is uniquely specified. There are many different hints possible just as there are many different formal proofs. Most mathematicians consider their theorems proved when they get a relatively high intermediate level of proof. Machines currently work only at the most formal level of proof. Little wonder that while machines can check proofs, they have difficulty constructing proofs. Our goal is to create machines that can prove nontrivial theorems. This work is joint with Matt Humphrey, Brendan Juba, and Ryan Williams.

3. The Human Oriented ID project. This is a cryptographic project to develop a challenge-response authentication protocol that a human can do entirely in his head: The human must be able to authenticate himself to a computer while a powerful adversary (one with a CRAY) -- who knows the protocol, listens on the line, and records every challenge and response -- should be incapable of learning to impersonate the human. This is a very hard problem.

STEPHEN BROOKES
Professor, Computer Science

My main interests concern the mathematical semantics of programming languages. I believe that proper attention to semantic foundations can yield significant benefits in developing techniques for proving properties of programs, in program design, in language design and implementation.

I am particularly interested in developing intensional semantic models, in which one is able to reason both about the correctness and efficiency of programs. This is in contrast to most traditional semantic models, which are extensional and focus on the input-output behavior of programs while abstracting away from computation strategy. I am working mainly on the semantic foundations of parallelism. This work includes the development of axiomatic proof techniques for establishing behavioral properties of parallel systems, design rules for parallel networks that guarantee desirable behavior such as deadlock-freedom, and the design and implementation of programming languages that employ parallelism uniformly and cleanly.
A semantics for a programming language is an assignment of meanings to program terms. For a semantics to be useful it should accurately capture the computational behavior of program terms, at an appropriate level of abstraction.

I believe that major improvements in the formal treatment of program properties can be achieved by paying careful attention to semantics. If we want to reason about a particular behavioral notion (such as partial correctness) we should first define a mathematical model for programs which precisely captures this behavior without being overly complicated. Ideally, we would like a fully abstract semantics: terms should be given the same meaning precisely when the terms would induce identical behavior in all program contexts. The construction of fully abstract models is by no means an easy task, and depends in any case on the underlying notion of behavior.

For modelling and reasoning about certain types of program behavior, such as partial or total correctness, an extensional semantics is satisfactory: the meaning of a program can be chosen to be a (partial) function from initial states to final states, and all details of how the program goes about its computation can be suppressed since all we really need to keep track of is the state transformation that the program induces. However, such a semantics is no use if we want to make comparisons between programs for the same function. In an extensional semantics all sorting programs have the same meaning, whereas we might well want to design a semantics with which we can compare sorting programs with different computation strategies. This motivation leads to a desire for a theory of intensional semantics. In an intensional semantics the meaning of a program is taken to be an algorithm rather than simply a function. An algorithm can be viewed as a function together with a (mathematical representation of a) computation strategy. I have recently developed a category-theoretic approach to the modelling of algorithms, and applied these ideas to the semantics of the lambda calculus. In the resulting semantic model, there is a complete partial order on algorithms and standard operations such as composition, application, and currying are continuous; thus, one may define algorithms recursively and use the standard techniques of denotational semantics (least fixed points) to reason about recursive programs, even at this intensional level. This approach using categories is rather general, and I am exploring several other possible applications.

Semantic principles and insights should be used in the design of new programming languages, to avoid the development of cumbersome languages in which the programmer may have to labor to overcome the syntactic quirks and idiosyncrasies of the programming language in order to express his algorithm as a program. I am particularly interested in designing a language that embodies parallelism uniformly: it ought to be as easy to specify parallel expression evaluation as it is to specify parallel execution of statements, and it ought to be easy to put together the results of parallel activities. The choice of an appropriate set of primitives for such a language should be guided by proper attention to semantic foundations, and I am carrying out research on this topic.

EMMA BRUNSKILL
Assistant Professor, Computer Science

My research centers on sequential decision making under uncertainty. From web advertising to robotic grasping to treating patients, decision making under uncertainty abounds, and the potential impact of better decision policies is enormous. I am particularly interested in creating algorithms with formal performance bounds for broad questions that are motivated by societal challenges, such as education. More generally, my research interests include reinforcement learning, machine learning, planning under partial observability, human agent interaction, and using information communication technologies for international development.

http://www.cs.cmu.edu/~ebrun
RANDAL E. BRYANT
University Professor, School of Computer Science

My current research interests are split between techniques for formally verifying hardware and software, plus a new style of parallel computing we call Data-Intensive Super Computing.

In formal verification, we have developed a number of methods over the years for representing system operation, specifying desired properties of the system, and then either proving the system satisfies the properties or determining ways it can fail. Much of our success has been built on methods for representing and reasoning about Boolean functions. Most recently, we have raised the level of abstraction up to word-level models, viewing a word as either a bounded or unbounded integer. Our work combines modeling, verification tools, and ways to reason about the underlying logic.

Data-Intensive Super Computing, or "DISC" seeks to create a class of systems that will enable computing over massive data sets. Currently, systems of this type are created by Google and its competitors to support web search, but we believe the ideas can be applied to many other domains, including computational biology, medicine, and commerce. It will involve research activities in system design, programming models and languages, parallel algorithms, and application areas such as machine learning and natural language processing.

JAIME CARBONELL
Allen Newell University Professor,
Computer Science, Language Technologies
Director, Language Technologies Institute

My interests span several areas of Artificial Intelligence, Language Technologies such as Machine Translation, Machine Learning and Computational Biology. In particular, my current research is focused on areas such as text mining (extraction, categorization, novelty detection) and in new theoretical frameworks such as a unified utility-based theory bridging information retrieval, summarization, question-answering, personalized search and related tasks. My research style typically combines theory, experimentation and system building, often in collaboration with students, research staff, and/or faculty colleagues. More specifically, the following illustrate recent research areas:

Active Learning: Supervised machine learning methods require labeled training instances, but in many practical applications the difficulty or cost of such obtaining such labels presents a major barrier. Active Learning seeks to identify the fewest and most significant instances to label, conditioned on learning method, estimated data distribution, learned decision function thus far, and other factors. I am interested in ensemble-based active learning, in active learning for highly skewed class distributions (a common phenomenon), and in differential labeling cost models.

Machine Translation: I am working on learning-based methods for Machine Translation, including generalized example-based MT (learns from parallel text), Context-Based MT (which requires only monolingual training text), and rule-learning for rare languages which may lack sufficient quantities of electronic text, parallel or otherwise.

Data and Text Mining: I am working on detecting the onset of novel patterns both in text streams and structured data streams. Novelty detection goes beyond discovering outliers, to determine emergent coherent groupings of data (e.g. clusters, temporally-related patterns) that extend, change or are unrelated to historical data patterns.
My interests span three areas: Programming Systems, Hardware, and Theory. I use the techniques and insights of theoretical computer science to solve problems in programming systems and hardware design that are of practical interest. I have a number of active research projects in these areas that I would be happy to discuss with students. Below are short descriptions of two research projects that I think are particularly exciting.

**Hardware and Software Verification.** Logical errors in finite state concurrent systems like sequential circuits and communication protocols are an important problem for computer scientists. They can delay getting a new product on the market or cause the failure of some critical device that is already in use. The most widely used verification method is based on extensive simulation and can easily miss significant errors when the number possible states of the system is very large. Although there has been considerable research on the use of theorem provers, term rewriting systems and proof checkers for verification, these techniques are time consuming and often require a great deal of manual intervention. My group has developed an alternative approach called temporal logic model checking in which specifications are expressed in a propositional temporal logic and an efficient search procedure is used to determine whether or not the specifications are satisfied.

In the twenty-five years that have passed since the original algorithm was published, the size of the systems that can be verified by this means has increased dramatically. By developing special programming languages for describing transition systems, it became possible to check examples with several thousand states. This was sufficient to find subtle errors in a number of nontrivial, although relatively small, circuit and protocols designs. Use of boolean decision diagrams (BDDs) led to a major increase in the size of the examples we could handle by this technique. Representing transition relations implicitly using BDDs made it possible to verify examples that would have required $10^{20}$ states with the original version algorithm. Refinements of the BDD-based techniques have pushed the state count up over $10^{100}$ states. By combining model checking with abstraction, we have been able to check even larger examples. In one case, we were able to verify a pipelined ALU design with 64 registers, each 64 bits wide, and more than $10^{1300}$ reachable states.

**Analytica --- A Theorem Prover for Mathematica.** Analytica is an automatic theorem prover for theorems in elementary analysis. The prover runs in the Mathematica environment and is written in Mathematica language. The goal of the project is to use a powerful symbolic computation system to prove theorems that are beyond the scope of previous automatic theorem provers. The theorem prover is also able to guarantee the correctness of certain steps that are made by the symbolic computation system and therefore prevent common errors like division by an expression that could be zero.

Since we wanted to generate proofs that were as similar as possible to proofs constructed by humans, we use a variant of natural deduction to generate proofs. We have demonstrated the power of our theorem prover on a number of non-trivial examples including the basic properties of the stereographic projection and a series of three lemmas that lead to a proof of Weierstrass’s example of a continuous nowhere differentiable function. Each of the lemmas in the latter example is proved completely automatically. In a related project that uses similar techniques, we have managed to prove all of the theorems and examples in Chapter 2 of Ramanujan’s Collected Works completely automatically. We believe these examples provide convincing justification for combining powerful symbolic computation techniques with theorem provers.
KARL CRARY  
Associate Professor, Computer Science  
My research interests are in the design and implementation of advanced programming languages and in applying programming language technology to improve the development, maintenance, and performance of software systems. I am particularly interested in the application of types to the structure and implementation of programming languages and software systems.

One current focus of my research is on mechanized metatheory. The aim of mechanized metatheory is to give proof of the properties of programming languages (for example, the type safety property says that no well-typed program can crash or otherwise violate its interface) in a form that can be checked by a computer. Not only does this give us greater confidence of the correctness of our proofs, but it also makes it possible for the proof itself to become part of a software system.

For example, we can use mechanized metatheory to provide a form of certified code. By equipping a program with a proof of type safety for the language it is provided in (this may be a source or executable language), we make it possible to determine the safety of the program automatically, simply by checking the proof and type-checking the program. Thus it is feasible to check the safety of programs in a purely automated fashion.

Another current focus of my research is on the application of certified code to operating systems. Using certified code, we can replace the dynamic protection checks that are ubiquitous in systems today with purely static ones. Since dynamic checks can be unreliable and costly, this can improve the reliability and performance of software systems. Moreover, static checks have greater expressive power than dynamic checks, so they can offer a greater level of protection and security than existing means.

ROGER DANNENBERG  
Professor, Computer Science, Art \((c)\), and Music \((c)\)  
My work is focused on various aspects of computer music, a field which poses many challenges for computer science. A central problem in computer music is expressive control, that is, the detailed control of timing, gesture, nuance, and tone quality that is essential to music. This problem has many facets, resulting in a variety of research directions. The Computer Music Project has developed new languages, development tools for real-time systems, synthesis techniques, and music understanding systems. This research is more than intrinsically interesting. It can shed light on related problems in real-time systems, multimedia, human-computer interaction, and artificial intelligence. Moreover, new possibilities of control and interaction in music are changing the very nature of music composition, performance, and aesthetics.

One research example is the development of new languages for expressing temporal behavior. One of these is Nyquist, a language that provides a single abstraction mechanism for the seemingly different notions of "note," "instrument," and "musical score." Nyquist gives composers an elegant, uniform notation that spans the range from low-level digital signal processing to high-level music composition. Nyquist is not intended for interactive real-time sound generation, but concepts from Nyquist are incorporated in other systems, including one of our own named Aura.

Expressive control of musical tones is another topic of research. A violin is expressive because there are many parameters under continuous control by the player, including bow pressure, finger and bow positions, and bow velocity. These give rise to variations in the resulting sound. My colleagues and I have developed a synthesis technique, spectral interpolation, which allows us to synthesize tones with interesting variations in spectra. Spectral interpolation has been used to accurately synthesize a variety of instruments. In the future, we will use this technique to give composers and performers greater intuitive control over synthesized sound.
ANUPAM DATTA  
Associate Professor, Computer Science, Electrical and Computer Engineering

My research focuses on the scientific foundations of security and privacy. I formalize security and privacy properties, design mechanisms for enforcing these properties, and principled analysis techniques for rigorously demonstrating that proposed mechanisms achieve desired properties. Much of my work has focused on reducing reasoning about security to reasoning about *programs* with suitable abstractions of security mechanisms (e.g., cryptographic primitives, hardware-based security features) and adversary capabilities (e.g., see my work on cryptographic protocols and trustworthy systems). More recently, I have started formally investigating the role of *people* in security and privacy (e.g., see my work on privacy, audit, and accountability).

(1) Privacy, Audit and Accountability

Privacy has become a significant concern in modern society as personal information about individuals is increasingly collected, used, and shared, often using digital technologies, by a wide range of organizations. One goal of this project is to precisely articulate what privacy means in various settings, and whether and how it can be achieved. In other words, we seek to develop conceptual and technical frameworks in which privacy notions (policies) are given precise semantics, algorithms for enforcing such policies, and characterizations of classes of policies that can or cannot be enforced. In addition to general results of this form, another goal of the project is to study specific application domains that raise significant privacy concerns in modern society and to apply these results (or specialized versions thereof) to these domains. Our current focus is on the healthcare domain. We are also thinking about privacy issues on the web and in online social media.

Specifically, to mitigate privacy concerns, organizations are required to respect privacy laws in regulated sectors (e.g., HIPAA in healthcare, GLBA in financial sector) and to adhere to self-declared privacy policies in self-regulated sectors (e.g., privacy policies of companies such as Google and Facebook in Web services). We investigate the possibility of formalizing and enforcing such practical privacy policies using computational techniques. We formalize privacy policies that prescribe and proscribe *flows* of personal information as well as those that place restrictions on the *purposes* for which a governed entity may use personal information. Recognizing that traditional preventive access control and information flow control mechanisms are inadequate for enforcing such privacy policies, we develop principled audit and accountability mechanisms with provable properties that seek to encourage policy-compliant behavior by detecting policy violations, assigning blame and punishing violators. We apply these techniques to several US privacy laws and organizational privacy policies, in particular, producing the first complete logical specification and audit of all disclosure-related clauses of the HIPAA Privacy Rule.

(2) Trustworthy Systems

The security universe includes a large class of computer systems (cryptographic protocols, trusted computing systems, hypervisors, virtual machine monitors, and Web browsers, to name a few) that are designed to provide security properties in the presence of actively interfering adversaries. A unifying theme of this work is to develop theories of security that include formal models of systems, adversaries, and properties, and support rigorous analyses indicating that the system satisfies the intended security property or identifying attacks on it. Given the complexity of these systems, two central classes of techniques that we have developed to achieve scalability are (a) composition techniques that enable us to conduct security analysis of complex systems by analyzing the smaller components from which they are built; and (b) abstraction techniques that enable us to reduce the security analysis of a complex system to that of a simpler system. The techniques are provably sound, i.e. no attacks are missed by applying them. We have applied these techniques to several classes of systems: (a) trusted computing systems – proving attestation properties and discovering a composition attack on two standard protocols; (b) hypervisors -discovering attacks that violate address separation properties and proving absence of attacks on the fixed designs; (c) network protocols - proving authentication and confidentiality properties of the OpenSSL handshake implementation and rediscovering a version rollback attack on it.

(3) Cryptographic Protocols
Protocols that enable secure communication over an untrusted network constitute an important part of the current computing infrastructure. Common examples of such protocols are SSL, TLS, Kerberos, and the IPSec and IEEE 802.11i protocol suites. SSL and TLS are used by internet browsers and web servers to allow secure transactions in applications like online banking. The IPSec protocol suite provides confidentiality and integrity at the IP layer and is widely used to secure corporate VPNs. IEEE 802.11i provides data protection and integrity in wireless local area networks, while Kerberos is used for network authentication. The design and security analysis of such network protocols presents a difficult problem. In several instances, serious security vulnerabilities were uncovered in protocols many years after they were first published or deployed.

We have developed Protocol Composition Logic (PCL), a formal logic for proving security properties of such network protocols. Two central results for PCL are a set of composition theorems and a computational soundness theorem. In contrast to traditional folk wisdom in computer security, the composition theorems allow proofs of complex protocols to be built up from proofs of their constituent sub-protocols. The computational soundness theorem guarantees that, for a class of security properties and protocols, axiomatic proofs in PCL carry the same meaning as reduction-style cryptographic proofs. Tool implementation efforts are also underway. PCL and a complementary model-checking method have been successfully applied to a number of internet, wireless and mobile network security protocols developed by the IEEE and IETF Working Groups. This work identified serious security vulnerabilities in the IEEE 802.11i wireless security standard and the IETF GDOI standard. The suggested fixes have been adopted by the respective standards bodies.

Personal Web Page: http://www.andrew.cmu.edu/user/danupam/

MICHAEL ERDMANN
Professor, Computer Science, Robotics

I am interested in making robots act purposefully and successfully in a world in which most everything is uncertain. Sensors are noisy, actions are imprecise, and models are faulty. I wish to understand how these uncertainties interact and how to overcome them. My research draws on tools from geometry, mechanics, planning, probability, and topology. Most recently I have explored topological methods for planning and control. Topology allows a system to abstract connectivity properties, filtering out the imprecision caused by uncertainty. For instance, one recent novel topological result is a graph controllability theorem:

A system can reach any state in a graph with control uncertainty if and only if the graph's strategy complex is homotopic to a sphere of dimension two less than the number of states in the graph.

SCOTT FAHLMAN
Research Professor, Computer Science, Language Technologies

I am interested in artificial intelligence and its application to real-world problems. Over the years, I have worked in many areas of AI, including knowledge representation, problem solving, image processing, machine learning, massively parallel approaches to search and inference, and the development of improved learning algorithms for artificial neural networks.

I am currently working on Scone, an open-source knowledge representation system and inference engine that can be used as a component in a variety of knowledge-based systems. Scone puts particular emphasis on efficiency, scalability (up to millions of entities and statements), and ease of use. One goal of the Scone research is to develop an extensive facility for "episodic memory". This can be used both to represent sequences of actions and events and as a source of plans or recipes for a flexible, instructable problem-solver. I am also working with my students to develop a natural-language interface for Scone.

I have also worked on programming languages and software development environments that support an incremental, evolutionary style of software development. This work has been done in Common Lisp, Dylan, and Java.
CHRISTOS FALOUTSOS  
Professor, Computer Science

There are two main focus areas: graph mining and stream mining. In the first, the goal is to find patterns in large graphs, so that we can spot anomalies, communities, patterns and regularities. Graphs appear in many instances: as document-term bipartite graphs in Information retrieval, as web pages or blogs linking to each other, as customer-product recommendations, as protein-protein regulatory networks, as computer-network traffic, and many more. Our emphasis is on scalability, so that we can handle graphs with thousands and millions of nodes. Research directions include time-evolving graphs, where we have been using 'tensors' to find patterns, as well as graphs where the nodes and/or the edges have attributes.

The second research area focuses on streams, which are semi-infinite numerical time series. The setting also has numerous applications, like sensor data monitoring, motion capture data, automatic alerts in the 'self-*' PetaByte storage system, chlorine level monitoring on the drinking water, and several more. The emphasis is to develop algorithms that inspect every measurement only once, and then discard it, since we can not afford to store the huge volume of historical data.

The common threads in both areas are the power-laws and the existence of self-similarity. Real graphs have skewed, Zipf-like degree distributions, and consist of communities-within-communities. Similarly, real sensor measurements are often bursty, but still self-similar, with bursts within bursts. We use or develop tools that exactly exploit the power laws and self-similarity, to find better patterns and anomalies than standard tools would find.

keywords: Database Management Systems, Data Mining, Graphs, Social Networks, Network Security.

KAYVON FATAHALIAN  
Assistant Professor, Computer Science

My work seeks to design high-performance systems for computer graphics and visual computing. Due to the compute-intensive nature of computer graphics applications, graphics systems must operate at exceptional levels of efficiency. To enable the development of new, complex applications, these high-performance systems must also be practical to program and use. To meet both these challenges, my students design new parallel graphics architectures (GPUs), develop parallel algorithms for rendering and simulation, and develop new programming models and compiler technologies that make it easier to express and optimize graphics computations.

Graphics System Design. Over the coming decade, high-resolution outputs and resource-constrained mobile environments demand that computer systems must synthesize images one-to-two orders of magnitude more efficiently than what is possible today. Through more flexible GPU architecture designs and corresponding new programming frameworks, we aim to allow efficient expression of a wider range of computer graphics techniques, increasing overall system efficiency by allowing choice of the right algorithm for the graphics task at hand. Identifying the right set of architectural mechanisms, and developing compiler technologies to target these mechanisms is a key aspect of this work.

Systems for Analyzing Images and Video. Ubiquitous image sensing will soon provide the capability to capture an increasingly large fraction of life's events, and a key computer science challenge is developing systems to intelligently interpret, analyze, and extract value from these large volumes of visual information. We are working to establish efficient computing platforms for lifetime- and society-scale image and video analysis and mining.

http://www.cs.cmu.edu/~kayvonf
EUGENE FINK  
Senior Systems Scientist, Computer Science, Language Technologies Institute  

My research interests are in artificial intelligence, machine learning, reasoning under uncertainty, and processing of massive data sets. I am currently working on the application of artificial intelligence techniques to the analysis of uncertain situations and targeted information gathering. I am also working on the application of machine learning and distributed computing to the analysis of massive astronomical data sets and cosmological simulations.

DAVID GARLAN  
Professor, Computer Science, Institute for Software Research  

My field of interest is software engineering, and specifically the areas of software architecture, self-adaptive systems, and cyber-physical systems. The common thread that links these areas is the problem of controlling the complexity of large software systems by providing a scientific basis for software design and analysis.

**Software Architecture.** Successful design of software architecture has always been a major factor in determining the success of a software system. Until recently architectural design has been largely based on ad hoc choice, informal experience, and local expertise. The goal of my research is make this knowledge precise, codified, and available to engineers as a matter of routine engineering. My research group has developed a number of languages and tools to support architectural design, including a widely used architecture design environment called AcmeStudio. Recently we have been investigating the possibility of putting such languages and tools in the hands of “end-user architects” – people in domains such as brain imaging, intelligence analysis, and computational biology who have to compose computational elements, but lack deep technical knowledge of computing to do this effectively.

**Self-adaptive Systems.** Increasingly systems must continue to operate continuously, interacting with diverse external services not under the control of the system designer. A new paradigm is emerging to handle this challenge: a system is augmented with a control layer that takes responsibility for observing the system’s behavior, and for maintaining or improving that behavior through run-time adaptation. Currently we are particularly interested in the application of these ideas to achieve self-securing systems. Additionally we are investigating the use of techniques such as probabilistic model checking, machine learning, and planning to augment adaptive capabilities over time.

**Cyber-physical Systems.** Today many of the computing systems that we depend on involve a combination of physical and computational elements. These systems include the energy grid, air traffic control, modern automotive/highway systems, and building control and automation systems. These systems are difficult to design in part because they require expertise in many disciplines, such as control theory, physical design, software systems, and distributed systems. In our research we are exploring multi-view design methods, in which different domains of expertise can work with models appropriate to that domain, but the models can be integrated and checked for consistency through shared architectural models. Currently we are working with Toyota on advanced automotive systems involving car-to-highway, and car-to-car communication.

Research Project Web Site: www.cs.cmu.edu/~able
GARTH GIBSON
Professor, Computer Science, Electrical and Computer Engineering

In broad terms I pursue research in large scale parallelism in computer systems and its implications on operating systems and computer architecture. My particular interests focus on large scale clustering technologies, parallel and distributed file systems, storage and system area networking, and secondary memory system technologies such as magnetic disk and flash storage design and optimization. I have a strong interest in shepherding technological advances from blackboard to commercial reality and widespread use. All of my research is housed in CMU's Parallel Data Laboratory (www.pdl.cmu.edu).

My early research on redundancy in parallel storage systems, called RAID, spawned a storage industry revolution and is now a checklist requirement of a $15+ billion dollar marketplace. My research on network-attached secure disks (NASD) is shaping new storage technologies including SCSI Object Storage Devices (OSD), high-performance IP-based storage and IETF Parallel NFS filesystems standards. NASD graduate students have gone on to shape Google's file system and database software and Seagate's next generation storage devices. By founding Panasas in 1999, I have also been driving the realization and deployment of these technologies into the mainstream of high performance storage technology. For example, the world's first PetaFLOP (and, as of July 2008, the world's fastest) computer is Los Alamos' Roadrunner cluster of Opteron nodes and 64b Cell accelerators. The primary storage system for Roadrunner is a Panasas storage cluster of about 1800 object storage servers bound together a single distributed system employing novel RAID techniques and virtualized as one storage pool.

On a broader note I play a leadership role in academic and industrial storage system developments. I sit on the steering committee of the leading storage systems conference, the USENIX conference on File and Storage Technology (FAST). I have sat on the technical council of the Storage Networking Industry Association. And I chair the IEEE technical field award for information storage systems contributions. Of late I have been reviewing my research with international communities; specifically, I have recently spoken in China, Britain, Germany, Canada, and Israel, and I have joined a scientific advisory board for a storage systems institute in Singapore.

DOE Petascale Data Storage Institute: Chartered by the Office of Science at the US Department of Energy I lead a team of researchers from CMU, U. Michigan, U. of California, and five National Labs: Los Alamos, Sandia, Oak Ridge, Pacific Northwest and Lawrence Berkeley. Our job is to anticipate the challenges of and guide efforts toward scaling high performance storage by 100% per year for the world's biggest computers' needs over the next decade (Peta- to Exa- scale systems).

Los Alamos Institute for Reliable High Performance Information Technology: I co-direct a partnership between CMU's Parallel Data Laboratory, CMU's Institute for Software Research and Los Alamos National Laboratory. Projects such as database interfaces on the metadata for huge scientific file systems and software debugging tools for large scale parallel scientific applications augment our basic goal of integrating the advanced systems thought going on in academic and national laboratories communities.

File Systems and Databases at Scale: My newest research directions explore the reorganization of large scale storage services inspired by internet services like Google's GFS and Bigtable and Yahoo's Hadoop open source cluster software stack. Parallel file systems and unstructured databases are being reconsidered and reorganized to be much more inherently scalable. Data Intensive Scalable Computing (DISC) is a paradigm shift for how large scale computing serves a very broad range of science, web users and industry. This is a very exciting time to be a core scalable storage systems researcher!

Website: www.cs.cmu.edu/~garth
My research theme centers around the design and manufacture of systems which contain massive numbers of components. My goal is to understand how to build robust and useful systems which can scale towards millions, billions or even Moles of components. While my main emphasis is on manufactured artifacts, I am also interested in how distributed systems of people can be effectively designed. In pursuit of this goal I am working in several different areas of computer science, electrical engineering, robotics, and public policy. My current main projects are The Claytronics Project (programmable Matter and Emergent Behavior) and Government 2.0. Past projects include the Phoenix project (www.cs.cmu.edu/~phoenix).

The goal of the Claytronics project (www.cs.cmu.edu/~claytronics) is to develop a form of programmable matter which moves programmability from the domain of computing into the domain of everyday matter. The idea is to create an ensemble of millions of very simple particles (the size of a grain of sand), each of which can compute, communicate, move, and adhere to one another. When taken as a whole, the ensemble can run a program which will result in the ensemble forming arbitrary dynamic shapes which can exert forces in the real world. In pursuit of this goal we are experimenting with building robots, developing programming languages, designing new debugging tools, creating new distributed algorithms, and studying emergence.

Meld, a programming language for massively distributed and concurrent systems, is one of the outgrowths of the Claytronics project. We are currently looking at how to use Meld to program both ensembles and more traditional parallel systems including multi-core processors and the cloud.

In addition to building sub-mm robots for the Claytronics project we are also pursuing more near-term goals of creating ensembles of smart interacting Lego-like blocks on the scale of tens to hundreds of units. These blocks, known as BlinkyBlocks, have been used by students around the world to play with the ideas behind programmable matter and develop a new understanding of how to program massively distributed systems.

Government 2.0 is looking at how to harness prevalent web tools with model checking and simulation to construct a system which crowd sources governance at a local, state, and country wide level.

ANUPAM GUPTA
Associate Professor, Computer Science

My main research interests are in Network Design and Metric Embeddings; I also work on Approximation and Graph algorithms.

Network Design and Optimization. Given a graph and a collection of users who want to communicate with each other, the aim of network design is to provision "good" networks satisfying the communication requirements. Asstated, things are still underspecified, and lead to many questions: e.g., what are the criteria for goodness? Are the networks capacitated? What is the cost model for allocating bandwidth? Are we routing paths (as in telephone calls), or can we deal with traffic as flows (as in packet routing)? What do communication requirements look like, and how are they specified? Furthermore, things are made more complicated by the fact that data is often not available beforehand—how does one handle uncertainty? I have been working on modeling and designing provably good algorithms for some of the problems arising from these issues. In particular, I have been working on handling uncertainty in data, and on designing networks for cost models that incorporate economies of scale.

Metric Embeddings. The goal of this area is to study the structure of metric spaces, and to use this understanding in the design of algorithms for a variety of problems arising on metrics. The approach to expose the inherent structure of metrics is to map the metric into a conceptually "simpler" metric that can be used in algorithmic applications in lieu of the original metric; of course, the new simpler metric should resemble the given metric, and hence the map should not distort distances by too
much. For instance, if we wanted to solve the travelling salesman problem on a metric space, and we could map the metric into a tree changing the distances by only 10%, then we could solve the TSP on the simpler metric optimally, and this would be within 10% of the optimal solution on the original graph. In recent years, embeddings have become an indispensible tool in the algorithm designer's toolbox, being very powerful and versatile; they have been used for geometric algorithms, finding good graph separators, online algorithms, network design, data structures and many other applications. Despite these successes, many fundamental problems remain open in the area, including understanding how well given metrics can be embedded into Euclidean and other normed spaces; how given data sets can be embedded into low-dimensional spaces without distorting distances substantially; how the topology of graphs interacts with their metric properties, etc. All this research proceeds hand in hand with the algorithmic applications, primarily to providing approximation algorithms for a variety of problems on graphs.

VENKATESAN GURUSWAMI
Associate Professor, Computer Science

My research interests span several topics in theoretical computer science including algorithmic coding theory, the role of randomization in computation, pseudorandomness and explicit combinatorial constructions, the theory of probabilistically checkable proofs, and the computational complexity of approximate optimization. Below I briefly describe some of these research directions.

Coding theory: Error-correcting codes provide a systematic way to add redundancy to data so that errors that occur during storage/transmission can be tolerated. In addition to their obvious importance to reliable storage and communication, codes have found many "extraneous" applications in complexity theory and cryptography, and are by now indispensable items in the toolkit of theoretical computer scientists. The basic principle in the design of codes is to ensure that distinct messages get encoded into codewords that are far apart, so that one will not be confused with the other even in the presence of some errors. The broad challenges in coding theory are three-fold: (i) formally specifying the noise model and understanding the potential and limits of error-correction for the concerned model; (ii) finding and explicitly specifying good configurations of codewords; and (ii) exploiting the structure of the designed code to to efficiently recover the transmitted codeword from a noisy received word. The game thus is to construct codes with low redundancy together with fast algorithms to correct large amounts of noise. I have been working towards this long term goal for several years, and in particular have extensively studied the algorithmic, combinatorial, and complexity-theoretic aspects of a notion of error-correction called "list decoding" which allows recovery from approximately 2x more errors compared to traditional error-correction algorithms. I am also excited by the practical potential of list decoding and am interested in experiments to validate the utility of list decoding for realistic noise models.

Theory of approximation algorithms: Many computational tasks arising in practice can be cast as optimization problems, where the goal is to find a solution subject to some constraints that optimizes a certain objective value. Unfortunately, for most interesting optimization problems, finding an optimal solution is NP-hard. One versatile approach to cope with this intractability is to settle for approximation algorithms that find approximately optimal solutions with provable guarantees on quality (for example, a solution of cost at most 20% more than the optimal). Ideally, for each problem of interest, we would like to have an algorithm with a provable performance ratio along with a hardness result ruling out a better performance ratio, thereby identifying its approximation threshold. Tremendous progress has been made on this topic in the last two decades, thanks to both algorithmic techniques such as linear and semidefinite programming and metric embeddings, and breakthroughs on the hardness side using the machinery of probabilistically checkable proofs (PCP) and new coding-theoretic, probabilistic, and (Fourier) analytic tools. My own (ongoing) research has made many contributions to this subject using a wide range of techniques, proving strong, and often near-optimal, bounds on the approximability of a broad class of fundamental optimization problems, including constraint satisfaction problems, ranking problems, graph-theoretic problems such as coloring and independent sets, disjoint paths and low-congestion routing, graph partitioning, etc. The
power of semidefinite programming and its relation to the Unique Games (and related) problems is an important focus of our current research.

**Pseudorandomness:** Pseudorandomness is a broad area that deals with efficiently generating objects that exhibit the desirable properties of "random-like" objects despite being constructed either explicitly or with limited randomness. Such pseudorandom constructions are important in the study of error-correcting codes, complexity theory, combinatorics, cryptography, and high-dimensional geometry. My research in recent years has addressed some of these challenges and led to good constructions of error-correcting codes, expander graphs, randomness extractors, Euclidean sections, compressed sensing matrices, etc. Despite the seemingly different definitions and motivations for the study of these objects, much of this progress was based on insights uncovering intimate connections between them. I am interested in strengthening these constructions when possible, and more broadly in expanding this body of work to find explicit constructions of other pseudorandom objects that arise in emerging applications (a salient example is the concept of subspace-evasive sets which found a surprising application in some of our recent work on list decoding).

Home Page: [http://www.cs.cmu.edu/~venkatg](http://www.cs.cmu.edu/~venkatg)

**BERNHARD HAEUPLER**  
Assistant Professor, Computer Science

In my research I love to think about the design and analysis of (randomized) algorithms for combinatorial problems. Often I mix this broad and classical area with ideas from distributed computing, information theory, and (network) coding theory. I am always on the lookout for new topics and problems to play with.

A common theme in my research is the use of randomness. It is surprising and sometimes almost unbelievable how much simpler and more efficient algorithms can become if allowed to make random decisions.

Some examples of my work include:

**Network Coding and Gossip Algorithms** (or how to spread information quickly and reliably): We show that often the most efficient way to disseminate information in a (distributed) network is to blindly forward messages to some randomly chosen contacts. Moreover, if too much information is available than can be put into a message sending then sending random mixtures (e.g., XOR’s) leads to optimal protocols. Moreover, the throughput achieved in this way is often drastically and provably higher than what is feasible with classical routing or store-and-forward protocols. In addition to being super simple, distributed, and efficient those – so called – gossip and random network coding algorithms also add a shocking amount of reliability against network failures and even the most hostile network dynamics.

**Coding for Interactive Communication** (or how to have a conversation despite noise): Very recently, I got excited about designing coding schemes that make interactive communications robust to noise the same way error correcting codes provide reliability to one-way information transfers. The problem with conversations/interactive communications is that what one wants to tell the other party depends highly on their responses and which also allows small misunderstandings to completely derail a conversation. This makes it much harder to efficiently add redundancy. Still, a great number of new ideas have let to basic proofs of existence for such coding schemes. Over the last year we have made tremendous progress in understanding how much noise can be tolerated, how one can obtain computationally efficient coding schemes, and what communication rates are achievable.
Algorithms for the Lovász Local Lemma: The probabilistic method tells us that an amazing array of beautiful and useful combinatorial structures exist but it often fails to lead to explicit examples or efficient constructions. This is particularly true, for proofs involving the Lovász Local Lemma which states that a large number of bad events can be simultaneously avoided as long as they are not too dependent among each other. The work of myself and many others have led to simple (even deterministic) algorithms which allow us to constructivize almost all applications of this powerful tool.

MOR HARCHOL-BALTER  
Associate Professor, Computer Science

I am interested in the performance analysis and design of computer systems, particularly distributed systems. I work on finding analytical models which capture the important characteristics of a computer system and allow me to redesign the system to improve its performance.

I believe that many fundamental conventional wisdoms on which we base system designs are not well understood and sometimes false, leading to inferior designs. My research challenges these age-old beliefs. Here are just a few examples:

- Thousands of "load balancing" heuristics do exactly that -- they aim to balance the load among the existing hosts. But who said that's necessarily a good thing?
- Migration policies for networks of workstations and distributed servers direct jobs to the host with least load. That seems good from the job's perspective, but is it best for the system overall?
- Given a choice between a single machine with power $p$, or $n$ identical machines each with power $p/n$, which would you choose?
- Migrating active jobs is generally considered too expensive. Killing jobs midway through execution and restarting them from scratch later is even worse! Says who?
- Ever notice that the "proven best" scheduling policies like SRPT (shortest-remaining-processing-time-first) are never used in practice? There's a fear that the big jobs will starve. Is this true?

Half my students work on mathematical techniques to derive theorems such as those above. These techniques include: queueing theory, probability theory, scheduling theory, Markov chains, stochastic processes, Matrix-analytic methods, renewal theory, real analysis, and more.

The other half of my students work on applying these theorems to implement high-performance Web servers, database systems, and distributed supercomputing servers.

ROBERT HARPER  
Professor, Computer Science

The goal of my research is to develop a comprehensive theory of programming that integrates with the practice of software development. The premise of my research is that programming is an explanatory activity, a form of expression intended to convey an idea that is both comprehensible by other people and executable by a computer. Language therefore plays a central role in programming. The overall goal of my work is to develop a language for computation that serves both purposes.

The focus of my work is on the development and application of type theory as the language of computation. As the name implies, the central organizing principle of type theory is the concept of a type. For example, familiar tree and graph structures may be viewed as instances of the general concept of an inductive type, infinitary data structures such as streams of values may be viewed as coinductive types, and language features such procedures or objects may be viewed as instances of the general concept of a function. A beauty of type theory is that it provides a rich framework that accounts not only for the computational aspects of programming, but also the reasoning involved in ensuring that a program behaves correctly. The main tool is the propositions-as-types principle in which specifications, or propositions, are identified with types, and proofs are identified with programs. A proof, after all, is a step-by-step procedure for transforming assumptions into
conclusions; it is, therefore, a program that takes assumptions as inputs and produces proofs as outputs. The theorem statement is a specification, or type, of this input-output behavior. Another beauty is that type theory connects directly to the language of mathematics through the concept of a category, a very general kind of algebraic structure. Types correspond to structures, such as topological spaces, and programs correspond to structure-preserving mappings between them. This provides a pathway for integrating the language of mathematics with the language of programming.

Home Page: http://www.cs.cmu.edu/~rwh

ALEXANDER HAUPTMANN
Principal Systems Scientist, Computer Science

My research interests revolve around the integration of text, image video and audio analysis. In the Informedia Project we have built the News-on-Demand application, which is an instantiation of the Informedia Digital Video Library idea, based completely on automatic methods for processing television and radio news. Through the combination of the strengths of speech recognition, natural language processing, information retrieval and interface design, the system is able to overcome some of the shortfalls inherent in each of the component technologies.

My goal is to utilize large corpora of "found data", that is data that is already available through the Internet or other readily accessible open sources, to improve speech and natural language processing by exploiting advantages across different modalities. It has become clear in recent years that large volumes of text, image, video and audio can be easily stored and made available for research and applications. However, most of these text, image, video and audio sources were not produced with computer processing in mind. My intention is to design and build intelligent, understanding programs that help process data from these sources and make the data useful for other applications. This data can be used to improve speech recognition, image understanding, natural language processing, machine learning as well as information retrieval. The challenge is to find the right data, process it into suitable form for training, learning or re-use and build mechanisms that can successfully utilize this data.

Speech and multimedia technology is about to make a major impact on our daily interaction with computers. What is needed at this point are clear demonstrations of the advantages of integrated speech and multimedia interfaces.

JESSICA HODGINS
Professor, Computer Science, Robotics

Dr. Hodgins's research focuses on the coordination and control of dynamic physical systems, both natural and human-made and explores techniques that may someday allow robots and animated creatures to plan and control their actions in complex and unpredictable environments. Her current research focuses on generating motion for computer animation by using motion capture data in combination with physically realistic simulation. Hodgins and her students have used dynamic simulation to animate human behaviors such as running, bicycling, diving, and vaulting. She and her colleagues have also used human motion data to bias planning algorithms towards more natural postures, to construct interfaces for avatars, and to capture the motion of muscles and skin. Hodgins is now refining some of these techniques for the control of a humanoid robot. Hodgins and her students have also explored passive simulations for animating phenomena such as clothing, water, breaking objects, and explosions.

Web site: http://www.cs.cmu.edu/~jkh
My research interests are in the areas of computer vision, visual and multi-media technology, and robotics. Common themes that my students and I emphasize in performing research are the formulation of sound theories which use the physical, geometrical, and semantic properties involved in perceptual and control processes in order to create intelligent machines, and the demonstration of the working systems based on these theories. My current projects include basic research and system development in computer vision (motion, stereo and object recognition), recognition of facial expressions, virtual(ized) reality, content-based video and image retrieval, VLSI-based computational sensors, medical robotics, and an autonomous helicopter.

**Computer Vision.** Within the Image Understanding (IU) project, my students and I are conducting basic research in interpretation and sensing for computer vision. My major thrust is the "science of computer vision." Traditionally, many computer vision algorithms were derived heuristically either by introspection or biological analogy. In contrast, my approach to vision is to transform the physical, geometrical, optical and statistical processes, which underlie vision, into mathematical and computational models. This approach results in algorithms that are far more powerful and revealing than traditional ad hoc methods based solely on heuristic knowledge. With this approach we have developed a new class of algorithms for color, stereo, motion, and texture.

The two most successful examples of this approach are the factorization method and the multi-baseline stereo method. The factorization method is for robust recovering shape and motion from an image sequence. Based on this theory we have been developing a system for "modeling by video taping"; a user takes a video tape of a scene or an object by either moving a camera or moving the object, and then from the video a three-dimensional model of the scene or the object is created. The multi-baseline stereo method, the second example, is a new stereo theory that uses multi-image fusion for creating a dense depth map of a natural scene. Based on this theory, a video-rate stereo machine has been developed, which can produce a 200x200 depth image at 30 frames/sec, aligned with an intensity image; in other words, a real 3D camera!!

Currently, we are working on a rapidly trainable object recognition method, a system for modeling-by-video-taping, and a multi-camera 3D object copying/reconstruction method.

**Visual Media Technology for Human-Computer Interaction.** A combination of computer vision and computer graphics technology presents an opportunity for a new exciting visual media. We have been developing a new visual medium, named "virtualized reality." In the existing visual medium, the view of the scene is determined at the transcription time, independent of the viewer. In contrast, the virtualized reality delays the selection of the viewing angle till view time, using techniques from computer vision and computer graphics. The visual event is captured using many cameras that cover the action from all sides. The 3D structure of the event, aligned with the pixels of the image, is computed for a few selected directions using the multi-baseline stereo technique. Triangulation and texture mapping enable the placement of a soft-camera to reconstruct the event from any new viewpoint. The viewer, wearing a stereo-viewing system, can freely move about in the world and observe it from a viewpoint chosen dynamically at view time. We have built a 3D Virtualized Studio using a hemispherical dome, 5 meters in diameter, currently with 51 cameras attached at its nodes. There are many applications of virtualized reality. Virtualized reality starts with a real world, rather than creating an artificial model of it. So, training can become safer, more real and more effective. A surgery, recorded in a virtualized reality studio, could be revisited by medical students repeatedly, viewing it from positions of their choice.

Or, an entirely new generation of entertainment media can be developed - "Let's watch NBA in the court"; basketball enthusiasts could watch a game from inside the court, from a referee's point of view, or even from the "ball's eye" point of view.

Also, I am interested in and currently working on vision techniques for recognizing facial expression, gaze, and hand-finger gestures. Such techniques will provide natural non-intrusive means for human-computer interface by replacing current clumsy mechanical devices, such as datagloves.
Computational Sensor. While significant advancements have been made over the last 30 years of computer vision research, the consistent paradigm has been that a "camera" sees the world and a computer "algorithm" recognizes the object. I have been undertaking a project with Dr. Vladimir Brajovic that breaks away from this traditional paradigm by integrating sensing and processing into a single VLSI chip a computational sensor. The first successful example was an ultra fast range sensor which can produce approximately 1000 frames of range images per second an improvement of two orders of magnitude over the state of the art. A few new sensors are being developed including a sorting sensor chip, a 2D salient feature detector (2D winner-take-all circuits), and others.

Medical Robotics and Computer Assisted Surgery. The emerging field of Medical Robotics and Computer Assisted Surgery strives to develop smart tools to perform medical procedures better than either a physician or machine could alone. Robotic and computer-based systems are now being applied in specialties that range from neurosurgery and laparoscopy to ophthalmology and family practice. Robots are able to perform precise and repeatable tasks that would be impossible for any human. The physician provides these systems with the decision making skills and adaptable dexterity that are well beyond current technology. The potential combination of robots and physicians has created a new worldwide interest in the area of medical robotics. We have developed a new computer assisted surgical systems for total hip replacement. The work is based on biomechanics-based surgical simulations and less invasive and more accurate vision-based techniques for determining the position of the patient anatomy during a robot surgery. The developed system, HipNav, has been already test-used in clinical setting.

Vision-based Autonomous Helicopter. An unmanned helicopter can take maximum advantage of the high maneuverability of helicopters in dangerous support tasks, such as search and rescue, and fire fighting, since it does not place a human pilot in danger. The CMU Vision-Guided Helicopter Project (with Dr. Omead Amid) has been developing the basic technologies for an unmanned autonomous helicopter including robust control methods, vision algorithms for real-time object detection and tracking, integration of GPS, motion sensors, vision output for robust positioning, and high-speed real-time hardware. After having tested various control algorithms and real-time vision algorithms using an electric helicopter on an indoor teststand, we have developed a computer controlled helicopter (4 m long), which carries two CCD cameras, GPS, gyros and accelerometers together with a multiprocessor computing system. Autonomous outdoor free flight has been demonstrated with such capabilities as following pre-scribed trajectory, detecting an object, and tracking or picking it from the air.

J. ZICO KOLTER
Assistant Professor, Computer Science

My research focuses on computational approaches to sustainability and energy domains, focusing on core challenges arising in machine learning, optimization, and control in these areas. On the application side, my interests range from improving the efficiency of generation, controlling power in smart grids, and analyzing energy consumption in homes and buildings. To attack these problems I focus on techniques from machine learning, reinforcement learning, time series prediction, approximate inference, and convex optimization, amongst others. Some examples of specific projects I am working on include:

Home Energy Informatics. How can we understand a home's energy consumption by just looking at an (unlabelled) aggregated power signal for the whole home? And if we get such information, how can we use it to improve home efficiency? To address this problem, we are developing new methods in probabilistic inference for aggregation models, motif-based methods for HMM learning, and model predictive control for controlling appliances to conserve power. We have also worked on developing new data sets for this problem, to bring the tasks here into the ML community as a whole.

Power Routing for the Smart Grid. Power in the electrical grid is typically viewed as "unroutable;" unlike network traffic, for example, power flow must obey fundamental physical laws. This has raised many issues with efficiency (due to line capacity constraints) and economics in power grids, and has
largely prevented the adoption of "CS-style" networking in power grids. However, recent advancements in power electronics have given us the power to control power flow in local ways. Our research is looking at control and networking methods that would enable arbitrary routing of power in electric grids, offering the potential to transform the smart grid into something that looks much more like computer networks.

Understanding National Energy Consumption. How many wind farms would it take to power the United States? While it's not hard to compute the answer from a raw power perspective, this ignores the indeterminacy of renewables and the losses associated with transmission. Our work looks to use the large amounts of publicly available data, including historical hourly power consumption and weather information, to arrive at an honest account of whether a given set of generators will actually be able to power the country (or smaller regions). Getting optimal answers to such problems requires solving large-scale optimization and resource allocation problems at very high speeds, to quickly give feedback to users of such a system.

Fundamental Algorithmic Advances. Many problems in energy and sustainability require new, massively scalable algorithms for probabilistic inference, learning and controlling dynamical systems, and optimization. We have several projects focused on these more "core" algorithmic areas, including: learning high-dimensional, sparse representations for dynamical systems; improving probabilistic inference with low-rank semidefinite programming; learning structured auto-regressive models for high-dimensional time series; and others.

TAI-SING LEE  
Associate Professor, Computer Science, Neural Basis of Cognition

Research in my laboratory seeks to elucidate the computational principles and neural mechanisms underlying visual perception. We use computational, mathematical and neurophysiological experimental techniques to study the biological visual system with a view to discovering the secrets of neural computation, advancing computer vision technology and restoring vision to the visually impaired.

Statistical approaches to understand neural codes and perceptual computation. Our perceptual systems are shaped by the natural environment. Our investigation of the neural codes and computational circuits in the brain thus often begins with a rigorous study of the statistical regularities in our visual environment and a parallel computational study to understand how these statistical regularities or probability distributions can support effective perceptual computation in a Bayesian inference framework. We then test theoretical predictions from these computational works on neural coding and computational circuits by examining the activities of hundreds of individual neurons recorded using multi-electrode array techniques. We have pursued this agenda to elucidate how statistical structures in 3D surfaces are encoded in neuronal tunings and connectivity, and how they can be used to facilitate 3D perceptual inference. We have developed new machine learning techniques to perform large-scale neural data analysis to study these and other issues, particularly on how elementary codes can be used to flexibly compose higher order concepts in the visual hierarchy.

By elucidating the principles of neural coding and neural computation, we seek to discover new computational algorithms for perceptual computation to improve computer vision systems and to develop approaches to generate mental visual images in the visually impaired subjects by stimulating the appropriate neurons in their brains.

Learning, adaptation and development in neural systems. Learning and adaptation are what make biological systems so much more robust and powerful than current man-made vision systems. Our research explores the basic mechanisms and principles underlying adaptation and learning in the visual system at different time scales and at the level of neurons and of neural systems. We have studied theoretically how neurons adapt dynamically to the statistical context of the visual stimuli. We have determined biophysical features in spiking neurons and identified statistical features in natural stimuli underlying neuronal adaptation. We have demonstrated experimentally that the neural machinery of perceptual processing is very flexible and subject to modification by behavioral
experience. These works have lead to a new perspective on the functional role of the early visual
cortex in vision. We are exploring how these design principles can be used to develop new learning
algorithms for learning hierarchical codes and for building the computational structures in our visual
systems.

http://www.cnbc.cmu.edu/~tai

MATT MASON
Professor, Computer Science, Robotics
Director, The Robotics Institute

I work on robotic manipulation. The ability to deal with almost any random object that comes along is
the most astonishing thing that humans and other animals do, and the most intriguing challenge in
robotics.

My main focus is the *Simple Hands Project*. Here's the idea. Most robot hands can be classified
as either simple or complex. Simple hands are like pliers or tongs, but highly specialized. Often they
are designed to handle just one particular object shape. Despite this limitation, simple hands are by
far the most common in practical applications.

Because of the highly specialized nature of most simple grippers, many roboticists work on complex
hands, usually anthropomorphic, with lots of fingers, joints, motors and sensors. Despite decades of
work on these hands, their apparent potential for general-purpose manipulation has not paid off.
Autonomous general-purpose manipulation has proceeded very slowly.

Here's an interesting observation on simple hands. When attached to a brain, rather than a
computer, they are very capable. A human with a simple prosthetic hook can run circles around any
robotic system. General purpose manipulation is a function of the brain, not the hand. That being the
case, there is no reason to focus robotics research on complex hands. In fact, simpler hands are
easier to understand, and hence better suited to scientific research. Of course they are also tougher,
lighter, cheaper ... they are handier.

*Simple Hands* might sound like a mechanical design project, but it is primarily a computer science
project. The main tasks are perception and planning. Perception uses all available information to
decide what is in the hand, and planning chooses actions to achieve an end goal. We have good
results applying machine learning techniques to the perception problem, successfully classifying
objects in the hand and estimating the pose. Most recently we are making good progress on
planning. And we are just beginning a project to use crowd sourcing / citizen science to observe and
analyze human manipulation strategies.

ROY MAXION
Research Professor, Computer Science, Machine Learning

Research Goals. Our ambition is to design and build computer systems that are safe and robust
against various kinds of faults, including malicious faults from information-warfare and
insider/masquerader attacks, as well as from unanticipated random error and what many would
interpret as user bone-headedness. We are trying to build systems that work reliably, all the time, for
everyone ... and to understand what makes some systems unreliable, and what can be done about it.

Specific areas of interest are:

Keystroke Dynamics and Forensics. Can users be identified on the basis of their typing rhythms and
styles? Keystroke timing can be monitored and used as a (behavioral) biometric in much the same
way that handwriting and fingerprints have been used (as physical biometrics). If we use machine-
learning classifiers, profilers and anomaly detectors, how reliable can the identification process be?
What stimuli should users type? Does it matter if everyone knows your password, because no one will be able to type it like you do? Does it matter what the password is, or how long it is? Can snippets of email or other text be used to determine that the authorized user (not someone else) is typing at the terminal? Can keystroke patterns be used in two-factor authentication schemes? What if a machine identifies a user, with the consequence that someone is accused of a crime and ends up in court - what are the forensic properties of keystroke analysis? What kinds of experimental methodologies are required for answering these kinds of questions with high confidence? These kinds of issues are being addressed in this project.

Keystroke Dynamics and Health. Can a person's typing rhythms, and the changes to them over time, be used as an instrument to sense the onset of major musculoskeletal disorders, such as carpal tunnel syndrome or digital flexor tendonitis? It's not hard to imagine that if you have a disorder of the wrist, hand or fingers, the particular disorder would effect a characteristic signature in a person’s typing rhythms. If that signature appears in someone’s typing, the person could be directed early to a physician, hence avoiding further deterioration. Similarly, can typing patterns reveal cognitive deficits like dementia or early onset Alzheimer's? Can personal stress be seen in subtle changes in typing? Could signs of stress be indications of malicious insider activity? Preliminary studies are positive; further work is ongoing.

Masquerade Detection. A masquerader is someone who pretends to be another user while invading the target user's accounts, directories, or files. This project is building systems that will detect the activities of a masquerader by determining that a user's activities violate a profile developed for that user. Profiling is based on various machine-learning and classification techniques.

Synthetic-Data Environment / Fault Injection. How do we gain confidence in a system's ability to detect failures, anomalies or performance perturbations? One method is by synthetic fault injection. This project's goal is to build a synthetic environment that can be used for validating algorithms for fault/anomaly/ intrusion detection. It will be able to replicate environmental conditions faithfully and repeatably, and will be easy to use for both experts and novices. One example would be synthetic typing.

Dependability/Assurance Cases. We are exploring the use of formal argumentation to support various claims of system dependability. One example is justifying safety claims for fly-by-wire or drive-by-wire systems; another is justifying claims that a fault-diagnosis system will handle all unanticipated faults; a third is that a biometric system can reliably discriminate among all its subjects. We may say that a system is safe or dependable or secure, but how do we muster the evidence to show it? And once the evidence is in hand, how do we structure and assess it to support the claims that are being made? The process is similar to presenting evidence in a jury trial, and some of the work involves liaison with trial attorneys. In the future, highly automated decision-making systems will need to construct arguments and gather evidence autonomously to support the correctness of their decisions, and we seek to lay the foundations for that.

Security Metrics. You can't manage what you can't measure, and so far there have been no useful metrics for security that enable us to answer the following types of important questions: How secure is this system? Is System A more secure than System B? How much money/effort will it take to secure a system to a certain level? What is the risk that an attack will penetrate my system? What is the risk of a breach of availability, integrity or confidentiality on my system? Creating reliable metrics and measures to help answer such questions is of paramount importance to the security community.

Performance-Shaping Factors. Stress or fatigue can affect your performance on almost any task. There are also elements of the task itself that influence performance; for example, the peculiarities of a particular programming language may induce programmers to make certain kinds of mistakes, or the design of a user interface may induce user error. We are interested in identifying such performance-shaping factors, measuring their effects on human/computer error, and finding ways to change computing environments so that errors are committed less frequently, thereby making computing safer, more secure and more reliable. Two examples that beg for attention are: Why do most coding mistakes occur in the exception-handling routines, as opposed to in other places in the
code? Why do users and administrators make so many mistakes configuring systems such as routers, servers, firewalls, file protections, and encrypted email? Mistakes can introduce serious security and performance vulnerabilities into systems, so it's worth while to determine what factors influence human performance, particularly when that performance is flawed.

PROJECTS: Performance-shaping Factors; Fault Tolerance; Experiments in Cyberspace; Fingerprints in Cyberspace, Behavioral Biometrics, Typing Signals and Health Care.

GARY MILLER
Professor, Computer Science

My main interest is in sequential and parallel algorithm design. Of particular interest are problems that arise in scientific computation and image processing. We have been working on three classes of problems. Our work is both more theoretical yet more practical since we require two important properties of our algorithms: they should be both be fast and have strong guarantees of quality, size, and speed.

Mesh Generation. The question of correctly and efficiently partitioning space into tetrahedra with good aspect ratio so that the features appear in the mesh is at least a fifty year old problem. The computer science community has been working on this problem for about twenty years. The most accurate simulations in the science, engineering, and graphic require small size quality meshes. CMU has been a leader on this problem for the last fifteen years.

Spectral Graph. Theory The interplay between graph theory and linear algebra is possibly one of the most interesting and practical areas in modern algorithm design. Possibly the most famous example is Google PAGE-RANK which is the Perron-Frobenius eigenvector of the link graph. The second example is the interplay between fast linear solvers and graph theory. We have ongoing collaboration on fast solvers and eigen calculations. These new algorithm work in near linear time.

Image Processing. We are interested in fast and reliable algorithm for image processing. Of special interest is 3D medical image processing. Our main tool that allows us to get new fast algorithms is spectral graph theory. This has allowed us to compute eigenvectors of our image as speed comparable to many simple filtering algorithms. We are also using interior point methods from convex optimization to de-noise images. Again these methods all use our fast solvers for speed.

TOM MITCHELL
Fredkin University Professor of AI and Learning, Computer Science, Machine Learning
Director, Machine Learning

I am interested in many areas of computer science, but especially in how to construct computers that learn from experience. At the heart of the problem of machine learning is the question of how to automatically formulate general hypotheses given a collection of very specific training examples. My research has addressed a number of approaches to this question, including statistical approaches that find regularities over large numbers of training examples, and analytical approaches that generalize from very few examples and rely instead on prior knowledge and reasoning.

Much of my current research focuses around two projects:

Machine learning approaches to analyzing human brain activity. This project uses functional Magnetic Resonance Imaging (fMRI) to capture three-dimensional images of human brain activity at a spatial resolution of 1mm, once per second. This is a wonderful set of data for studying the operation of the human brain, and because it is relatively new, there is a great need for new algorithms to analyze the data. Recently we have demonstrated that it is possible to train machine learning algorithms to decode mental states of human subjects (e.g., to determine whether the word a person is examining is a noun or a verb) based on their observed fMRI brain activity. I am interested in developing new algorithms that will help discover the spatial-temporal patterns of activity associated...
with a variety of brain processes, and that will help us better understand the working of the human brain. We have access to the CMU-University of Pittsburgh Brain Imaging Research Center, to design and collect data for our own experiments.

This project raises interesting machine learning questions such as how to train classifiers in extremely high dimensional, noisy data, and how to learn temporal models that characterize the evolution of hidden cognitive states while humans perform tasks such as reading and answering questions.

**Intelligent workstation assistants that learn to help their users.** This is part of a large multi-researcher project to build enduring, personalized, learning assistants for users of computer workstations (like us!). We are working toward a software agent that can understand the user's email, calendar, text files, and actions, and that can learn the user's interests, habits, and tasks, in order to help in a wide range of activities. My specific interest lies in how to make the agent learn. For example, I am currently interested in the question of how the agent can learn to automatically extract information from text emails and files, and how it can learn what threads of activities the user is involved in, when, with whom, about what, etc. This project raises many interesting machine learning questions about learning from labeled and unlabeled data, about learning and statistical language processing, and about cumulative learning over long periods of time.

**JAMES MORRIS**  
Professor, Computer Science, Human Computer Interaction

I am interested in Computer-Mediated Human Communication. At the low end this involves the engineering of distributed communication systems and the design of document interchange strategies. At high, or human, end it involves activities like calendar management, goal reconciliation, and group processes. The specific projects I'm currently involved with The Prep Editor, a collaborative writing tool, and Team-Centered Design, a new study of collaborative tools and methods for tasks such as crisis management.

**TODD MOWRY**  
Professor, Computer Science, Electrical and Computer Engineering

The goal of my research is to dramatically boost the performance of future microprocessor-based systems. To accomplish this, we exploit various forms of parallelism through a combination of novel architectural, compiler and operating systems support. In particular, we have been focusing on the opportunities and challenges created by two important VLSI technology trends which are expected to reshape computer systems over the next decade: the potential for single-chip multiprocessing due to higher levels of single-chip integration, and the need to tolerate off-chip latency as the gap between processor speed and the speed of memory and I/O continues to widen.

**Single-Chip Multiprocessing: The STAMPede Project.** As advances in integrated circuit technology continue to provide more and more transistors on a chip, processor architects are faced with the pleasant challenge of finding the best way to translate these additional resources into improved performance. One of the more compelling options is to integrate multiple processors onto the same chip. While this will certainly increase computational throughput, it will only reduce execution time of a given application if it can be run in parallel. Hence the key question is how do we convert the applications that we care about into parallel programs? Expecting programmers to only write parallel programs from now on is unrealistic. Instead, the preferred solution would be for the compiler to parallelize programs automatically. Unfortunately, compilers have only been successful so far at parallelizing the numeric applications commonly run on supercomputers. For single-chip multiprocessing to have an impact on the majority of users, we must also find a way to automatically parallelize the non-numeric applications (e.g., spreadsheets, web software, graphics codes, etc.) which account for the bulk of the software run on commercial microprocessors. Based on our preliminary studies, we believe that a breakthrough in our ability to automatically parallelize non-numeric applications may be possible through "thread-level data speculation", which is a technique
that allows the compiler to safely parallelize applications in cases where it believes that dependences are unlikely, but cannot statically prove that they do not exist. To accomplish this, we add modest hardware support to track data dependence violations at run-time and alert the software so that it can recover appropriately. Developing the architectural, compiler, and operating system support necessary to turn this potential into a reality is the goal of the STAMPede (Single-chip Tightly-coupled Architecture for MultiProcessing) project.

Coping with Large Latencies. Processor speeds are continuing to increase far more rapidly than off-chip components such as DRAM, disk, and networks, largely due to physical limitations such as distance and the speed of light. The challenge presented by this trend is that from the processor's perspective, the latency of main memory and I/O is increasing at a dramatic rate, and thus threatens to become an increasingly important performance bottleneck. The good news, however, is that the bandwidth of these off-chip devices has been improving through innovations such as synchronous (i.e. pipelined) DRAM, disk arrays, and fiber optic networks. Therefore we are exploring new ways that the compiler (with varying degrees of help from the hardware and the operating system) can use prefetching and other techniques to intelligently trade off consuming more bandwidth to reduce overall latency. Recent work in this area has included prefetching pointer-based codes, prefetching to hide disk latency in out-of-core numeric applications, and hiding network communication latency in workstation clusters.

RYAN O’DONNELL
Associate Professor, Computer Science

My main research interests are in computational complexity, especially hardness of approximation and computational learning theory. I also have strong interests in discrete harmonic analysis and in probability.

There is a pat view of computational complexity which holds that almost every natural algorithmic task is either solvable efficiently (in polynomial time) or is NP-hard. Factoring and Graph-Isomorphism are frequently cited as the "only" natural problems to resist such a classification. However, this view is not at all accurate. In fact, there are two major genres of important, intensively-studied algorithmic tasks containing many problems not known to be in P and not known to be NP-hard:

- **Approximation problems** -- finding solutions to classic NP-hard problems (e.g., Max-Cut, Max-SAT, Vertex-Cover) that are within a small factor of being optimal.

- **Learning problems** -- identifying certain kinds of boolean functions given only random examples, \((x, f(x))\).

I am working on problems in both areas.

DAVID O’HALLARON
Professor, Computer Science, Electrical and Computer Engineering

I work in the area of computer systems. My interests within computer systems are quite broad, including online education, cloud computing, and scientific computing. My main interest these days is the phenomena surrounding autograding, that is, programs evaluating other programs.

The CMU Autolab group is developing a new cloud-based service that teachers around the world can use to offer programming labs for their computer classes. The service is based on the notion of autograding, that is, programs evaluating other programs. Our vision is that teachers can select the labs for their classes from a repository of high-quality labs written by other teachers and students. An author whose lab is adopted for a class receives community recognition, in the form of a public adoptions page, and possibly even a small royalty. Each time a student hands in their work for credit,
the service spins up a new VM and autogrades the student’s work in this new VM. The scores are displayed, anonymized, on a realtime scoreboard that is visible to everyone in the class.

Our aim with this work is to improve the quality of computer science education worldwide by providing a way for teachers to share their good labs with each other. This sharing will enable good teachers to get credit for their work from their peers. The assignment of credit, in turn, will help foster the birth of a new reputation-based community for teachers. Researchers have enjoyed this kind of community for years, but teachers have never developed one, working for the most part in isolation, with little opportunity to learn and benefit from each other.

Students who are passionate about education would enjoy working with us. Drop by any time if you want to chat.

ANDY PAVLO
Assistant Professor, Computer Science

Creating a large-scale database application is easier now than it ever has been, in part due to the proliferation of distributed system tools, cloud-computing platforms, and affordable mobile sensors. But now the processing and storage needs of Internet-scale, "Big Data" applications are surpassing the limitations of legacy database management systems (DBMSs). As a result, I am interested in the research and development of new DBMS technologies for these modern high-volume and data-intensive applications. In particular, my research is focused on novel distributed and parallel DBMS architectures for transaction processing applications (OLTP), analytical/business intelligence workloads (OLAP), and scientific computing. Much of my work is in applying techniques from machine learning and optimization research to enable these distributed DBMSs to execute workloads that are beyond what single-node systems can support. I am also interested in studying the performance characteristics of non-volatile memory devices in the context of Big Data systems in order to build the groundwork for new DBMS architectures that can take advantage of these emerging technologies.

CMU Database Research Group: http://db.cs.cmu.edu/

FRANK PFENNING
President's Professor of Computer Science
Head, Computer Science Department

At the heart of my research lies the desire to understand the principles of programming languages. Programming languages are the key to the programming process and therefore of fundamental importance to computer science. Well-designed programming languages allow fast program development, ease software maintenance, and increase confidence in the correctness of implementations. Poorly designed programming languages lead to verbose and impenetrable programs that are difficult to debug and maintain. One of the trends in computer science has been the development of a plethora of languages, often for very specific purposes. Unfortunately, many of these languages are woefully misdesigned, because their developers were unaware of or have disregarded basic principles of programming language design. My research thus aims at discovering such principles and experimenting with them through implementations and environments.

In support of this goal, I am pursuing three interconnected threads of research. The first is the development of meta-languages which codify programming language concepts and support formal reasoning about properties of programming languages. The second is the design of expressive type systems for practical programming languages which allow more program errors to be caught at compile-time without sacrificing conciseness or efficiency of programs. The third is the application of programming language techniques in domains where they are currently undervalued, such as mobile code or robotics. In all of these I collaborate closely with colleagues and students who are not mentioned explicitly below. Please refer to my home page for recent drafts and publications.

Meta-languages. In this area my research focuses on the development of a uniform meta-language
and environment which supports specification, implementation, and formal reasoning about programming languages and logics. The currently released implementation is Twelf 1.2 which embodies many of the representation and implementation techniques discovered in my research on logical frameworks. Underlying Twelf is a type theory which is used for specification, constraint logic programming, and meta-theoretic reasoning. Twelf is a significant step towards an environment for teaching and research in the areas of programming languages and logics. Current and future work on Twelf consists mainly in improving its expressive power to capture more language phenomena in a concise and natural way. There is ongoing work on a linear extension (to capture imperative and concurrent computation), an ordered extension (to capture adjacency and sequencing), and extension by constraints (to capture integers, rationals, and similar domains).

Type systems. In this area I have concentrated on extending the expressive power of type systems to allow more properties of programs to be checked statically. Invariants which can otherwise only be stated informally can be expressed in these systems and verified by a type-checker. This provides additional machine-checked documentation, more detailed interface specifications at the module level, and allows more errors to be detected at compile-time. All these properties combine to improve programmer productivity and simplify program maintenance. Concretely, I have developed refinement types (for inductively specified properties of data representation), dependent types (for array bound and similar constraints) and modal types (for run-time code generation).

Applications. I believe that programming language research cannot exist in a vacuum—it must address problems encountered by real programmers in real applications. Whether this is the case is often difficult to assess, since at present the gap between research on advanced programming languages and programming practice is, unfortunately, very large. One way to reduce this gap is to take problems faced by programmers today in specific areas and contribute to their solution with programming language techniques. The ConCert project at CMU is an example of this, where we develop techniques for combining modularity with safety and efficiency in the context of grid computing using tools from logic and type theory. In other research I am exploring questions of programming language design for applications in robotics and manufacturing.

ANDRE PLATZER  
Associate Professor, Computer Science  
I work in the areas of logic in computer science, cyber-physical systems, programming languages, and formal methods. Below are short descriptions of some of my research topics and directions.

Logic of Dynamical Systems  
Logic of dynamical systems studies logics and proof principles for properties of dynamical systems. Dynamical systems are mathematical models describing how the state of a system evolves over time. They are important in modeling and understanding many applications, including embedded systems and cyber-physical systems (CPSs). CPSs combine cyber capabilities (computation and/or communication) with physical capabilities (motion or other physical processes) to solve problems that no part could solve alone. Cars, aircraft and robots are prime examples, because they move physically in space in a way that is determined by discrete computerized control algorithms. Designing these algorithms to control CPSs is challenging due to their tight coupling with physical behavior. At the same time, it is vital that these algorithms be correct, since we rely on CPSs for safety-critical tasks like keeping aircraft from colliding.

"How can we provide people with cyber-physical systems they can bet their lives on?" [Jeannette Wing]

In our group, we develop the logical foundations of cyber-physical systems and study the theory, practice, and applications of logics of dynamical systems and their use in formal verification and validation. This relatively young area is a promising direction for future research, unique in
its manifold connections to other pure and applied sciences, including many areas of mathematics, physics, and control theory.

**KeYmaera: A Hybrid Theorem Prover for Hybrid Systems**

KeYmaera is a hybrid verification tool for hybrid systems that combines deductive, real algebraic, and computer algebraic prover technologies. It is an automated and interactive theorem prover for a natural specification and verification logic for hybrid systems. KeYmaera supports differential dynamic logic (dL), which is a first-order dynamic logic for hybrid programs, a program notation for hybrid systems. KeYmaera supports hybrid systems with nonlinear discrete jumps, nonlinear differential equations, differential-algebraic equations, differential inequalities, and systems with nondeterministic discrete or continuous input. For automation, KeYmaera implements a free-variable sequent calculus and automatic proof strategies that decompose the hybrid system specification symbolically. This compositional verification principle helps scaling up verification, because KeYmaera verifies a big system by verifying properties of subsystems. To overcome the complexity of real arithmetic, we integrate real quantifier elimination and other decision procedures following an iterative background closure strategy. The KeYmaera verification tool is particularly suitable for verifying parametric hybrid systems and has been used successfully for verifying case studies from train control, car control, air traffic control, and robotics.

**NANCY POLLARD**

Associate Professor, Computer Science, Robotics

I am interested in understanding physical interaction with the environment --- how do we select and apply exactly the right forces to maneuver bulky and heavy objects, scramble over large rocks using both hands and feet, or use hand held tools?

In robotics, a better understanding of these interaction forces can help us create more dexterous robots that are able to operate in an environment such as the home. In particular, we would like to create natural grasping and manipulation behavior using measured human examples as a resource. In initial experiments we have demonstrated a humanoid robot tumbling a variety of large, heavy objects using a strategy derived directly from a human example. Some of the questions that remain to be answered are "what does it really mean for a robot to perform a task in the same way as a person?", and "how can we convert a collection of measured human examples into a robust control policy for a robot?"

In computer graphics, an understanding of interaction forces can help us to create more natural looking motion when a character climbs, performs athletic maneuvers, or manipulates objects. We have developed fast techniques for computing optimal, physically plausible motion. We are also exploring the importance of physical correctness in graphics applications. How physically incorrect can motion be before people start to notice? In other words, how much can we cheat?

One of my particular areas of interest in both robotics and graphics is the hand. Modeling convincing hand motion is very difficult; in fact the hand itself has almost as many degrees of freedom, or directions of motion as is typically used to model the entire rest of the body! However, observed motion of the hand often appears to be much less complex. By studying examples of human hand motion and studying human hand anatomy, we hope to characterize hand behavior in a way that can be exploited for easier control of animated hands and effective control of robot hands.

**ARIEL PROCACCIA**

Assistant Professor, Computer Science

I am interested in the mathematical foundations of artificial intelligence, and mainly think about questions at the (pairwise) intersections of AI, social choice, and game theory. Specifically, my research interests include, but are certainly not limited to: multiagent systems, computational social
choice and preference handling, computational mechanism design and fair division, machine learning, social networks and reputation systems, decision making under uncertainty, and human computation.

Some examples of projects I am currently working on:

**Dynamic social choice:** I am augmenting social choice theory using models and techniques from the AI research on decision making under uncertainty to create and analyze new social choice models where preferences can change dynamically.

**Incentives in machine learning:** I am studying the role of incentives play in machine learning, and designing machine learning algorithms that are immune to manipulation by strategic agents.

**Voting and human computation:** I wish to design optimal voting rules for use in online labor markets such as Amazon Mechanical Turk.

**Fair division of resources:** How does one cut a cake fairly? This is a long-standing question that is algorithmic in nature but has received little attention from computer scientists. Related issues include allocation of shared resources in cloud computing environments.

---

**RAJ REDDY**

Moza Bint Nasser University Professor, Computer Science, Institute for Software Research, Robotics

Dr. Reddy's research interests include the study of human-computer interaction and artificial intelligence. His current research projects include spoken language systems; gigabit networks; universal digital libraries; and distance learning on demand.

---

**STEVEN RUDICH**

Professor, Computer Science

The main focus of my research is in complexity theory and the foundations of cryptography. The wide applicability of the technical ideas in these two areas has allowed me to extend my work into learning theory, probability theory, and combinatorics.

I concern myself with the fundamental questions of each field: How can one obtain a separation result for interesting complexity classes? How can one reduce the security of a cryptographic protocol to the security of a simple primitive? What is a powerful inference method? How can one generalize probability theory to a theory where the events are almost independent?

I work most actively is the meta-theory of the above areas. This means questions like: What are the reducibilities among the above questions? What are the reducibilities among approaches to these questions? When are the perceived technical barriers to their answers related? When are they inherent?

I will mention two examples of these meta-theoretical results. Much of cryptography concerns the reduction of the security of complex protocols to the security of simpler ones. The most important question along these lines is whether a public-key cryptosystem can be based on a black-box, private-key system (such as the Clipper chip). Russell Impagliazzo and I showed that any proof of this particular reduction would contain inside it a proof that P<NP. It follows as a corollary that some 42 similar reductions between "high" and "low" cryptography are also difficult, because they too are disguised forms of the P v.s. NP question. I have shown that the same sort of thing applies to reductions between high and low interaction public-key protocols.
Curiously, current approaches to the P v.s. NP question are themselves disguised forms of a problem in cryptanalysis. In order to prove that a function $f$ is not in a complexity class $C$, the standard approach is to exhibit some combinatorial property of the function that provably prevents it from being in the class $C$. Alexander Razborov and I have argued that all the known lower bound proofs in circuit complexity use what we call Natural combinatorial properties. Any proof that uses a Natural property and shows that some function is not contained in a complexity class $C$, can be transformed into a successful cryptographic attack against any private-key cryptosystem which is implementable in the class $C$. Thus, these proof techniques cannot apply to classes (plausibly $P$, $NC^1$, or even $TC^0$) that contain unbreakable cryptography. A corollary greatly extends my previous work with Len Berman by showing that the restriction methods, which were so successful in showing $AC^0$ lower bounds, are essentially useless in resolving frontier questions in circuit complexity. At this time it remains unclear if proof methods which avoid Natural properties exist. Razborov has shown that in a weak fragment of arithmetic all proof methods are Natural in our sense; it follows under a cryptographic assumption that P v.s. NP is independent of this fragment. I have extended the notion of Natural property to that of NP-Natural property. This generalization also extends Razborov’s independence result to more general (but still quite weak) proof systems.

ALEXANDER RUDNICKY  
Research Professor, Computer Science

My research centers on interactive systems that use speech. I am currently interested in the following problems:

Implicit Learning: Human-computer interaction generates information that the system could use to modify its behavior. For example, a speech recognition error that is repaired leaves information about a misclassification that could be used to improve subsequent accuracy. Exploiting such situations (and in fact contriving to create them) can provide a rich set of experiences that drive self-improving systems

Automatic Detection and Recovery from Error: Humans easily detect and recover from breakdowns in communication. Automatic systems are less successful at this, however we can use features of recognition, understanding and dialog to predict the likelihood of misunderstanding at a given point in an interaction, and then apply heuristic strategies for guiding the conversation back onto track.

A Theory of Language Design for Speech-Based Interactive Systems: Speech-mode communication predisposes the user to make certain word choices and to exhibit certain grammatical preferences. An understanding of the principles that underlie these preferences (and how these can be influenced by the system's language) leads to better language design for interactive systems.

The Role of Speech in the Computer Interface: Speech is an effective means of communication, but it is not always suitable for all types of interaction. Ideally we can analyze an interface in terms of the task(s) it will be used for, the costs of specific interactions and the value perceived by the user. To date we've studied models based on time, system error and task structure. These models turn out to be useful for simple systems and appear to be extensible to more complex systems.

Many of these issues are being explored in the context of working systems, for example a language interface for a team of humans and robots working together (Treasure Hunt) or a information access for conferences (ConQuest) or for scheduling court time (Let's Play).
TUOMAS SANDHOLM  
Professor, Computer Science

My main research interests are in artificial intelligence, electronic commerce, game theory, multiagent systems, automated negotiation, contracting, auctions, exchanges, coalition formation, voting, algorithmic mechanism design, and normative models of bounded rationality. A key goal of this research is to construct electronic marketplaces that are efficient both in terms of the resulting economic allocations and the computational processes. We approach this by developing incentive-compatible market mechanisms and the algorithms that execute those mechanisms efficiently, as well as by designing software agents that act optimally on behalf of the real-world parties that they represent in such markets. A central subject of this research is the deep interaction between the game-theoretic incentive issues and the computational issues that stem from computational limits on the mechanism and the agents. Our research involves theoretical work such as models of limited computation, mechanism design, equilibrium analysis, and algorithm design as well as computational experiments and system building. Many of our systems have been commercially fielded.

The following are just some examples of my current projects:

**Solving Games, Poker.** We design algorithms for abstracting games and algorithms for solving for the equilibrium of a game, i.e., the best way to play it. For example, we have developed a lossless abstraction algorithm that enabled us to exactly solve poker games over 10,000 times larger than the largest solved previously. Our algorithms have also yielded some of the best poker playing programs for Texas Hold'em. We are also developing opponent modeling and exploitation techniques. Our work also involves complexity analysis and developing new game-theoretic solution concepts.

**Expressive Auctions and Exchanges.** We are developing the theory and methodology of how one should design markets when the participants have rich preferences over outcomes.

**Advertising Markets and Electricity Markets.** We are developing techniques, and founded a new startup company, in the space of expressive advertising markets, such as for TV advertising. We are also working in electricity markets.

**Kidney Exchange.** We have developed the market designs and algorithms that run the nationwide kidney exchange at UNOS. Our current research focuses on developing better models and algorithms for the dynamic problem.

**Search and Integer Programming.** For example, how should those tree search algorithms branch? Also, we are developing techniques for automated and recursive decompositions.

**Revenue Maximization and Automated Mechanism Design.** We are pioneering the idea of using optimization algorithms to automatically design the rules of a game (such as an auction) so that the designer's objective is maximized even though the participants play based on self-interest. An example application is revenue-maximizing combinatorial auctions. Another example is channel abstraction (bundling, automated item definition), such as in TV and Internet display advertising.

**Mechanisms for Computationally Limited Agents.** If agents are computationally limited, how should they allocate their deliberation? What should good mechanisms (e.g., auctions) look like for such agents? It turns out that sometimes mechanisms that lead to greater systemwide good can be designed for computationally limited agents than unlimited ones!

You can check out my papers via my home page www.cs.cmu.edu/~sandholm.
M. SATYANARAYANAN  
Carnegie Group Professor, Computer Science

As an experimental computer scientist, I design, implement and evaluate computing systems that expand our knowledge and also have real-world impact. My past work includes the Andrew File System (OpenAFS), Coda File System, Odyssey and Aura. See my bio for more details of these systems and their impact. My current research involves two major thrusts

The Diamond project explores interactive search of Internet data repositories that store vast amounts of complex, non-indexed data such as digital photographs, video streams, and medical images. At the heart of this work is the concept of early discard, or the ability to reject irrelevant data items very close to their point of storage. Since the knowledge needed to recognize irrelevant data is domain-specific, early discard requires application code called a searchlet to be executed close to storage. Diamond embodies the concept of self-tuning, which allows it dynamically adapt to different hardware configurations, workloads, and data content in a manner that is completely transparent to users and applications. Medical and pharmaceutical researchers at UPMC, University of Pittsburgh School of Medicine, and Merck are collaborating with Diamond researchers to apply Diamond to their domain-specific tasks. This may open the door to research and diagnostic strategies that were not considered feasible until now.

The Internet Suspend/Resume (ISR) project explores a new model of mobile computing that cuts the tight binding between PC state and PC hardware. By layering a virtual machine on distributed storage, the ISR system lets the VM encapsulate execution and user customization state; distributed storage then transports that state across space and time. ISR makes extensive use of Content Addressable Storage (CAS) concepts for storage efficiency and good performance. Current ISR goals include VMM agnosticism, support for transient thin-client mode, guest-aware migration, and cross-parcel data sharing. We are also exploring the use of smart phones as trusted assistants for improving the safety and performance of this new style of computing.

www.cs.cmu.edu/~satya/

BRADLEY SCHMERL  
Senior Systems Scientist, Computer Science

My research interests are in the general field of software engineering, focusing on Software Architecture, Self-adaptive Systems, and Everyday Computing.

Software architecture: Techniques and tools for precisely defining the design of a software system, specifically so that its quality attributes can analyzed early in the development lifecycle. My current specific interests are in how to use architectural modeling techniques to design system that have high degrees of software and physical elements, so-called cyber-physical systems.

Self-adaptive systems: Many systems today have the requirement that they must be kept running 24/7, and be robust and responsive even in changing environments. This requires software to dynamically change. This research is exploring the use of software architectures to provide advice as the system runs on how the system should adapt and change, while still maintaining quality of service requirements.

Software for everyday people: Software is becoming all pervasive and everyday non-computer scientists must interact with software to get their daily tasks done. In this research, I am interested in how people's tasks can be better supported by software, either by capturing mundane tasks so that they can be automated, or by providing simple specification techniques that allow people to define how these tasks should be automated. Current research is focusing on providing software platforms and tools to aid in the automation of workflows for scientists studying human social, cultural, and behavioral structures.
SRINIVASAN SESHAN
Professor and Associate Department Head for Graduate Education, Computer Science

My primary interests are in the broad areas of network protocols and distributed network applications. In the past, I have worked on topics such as transport/routing protocol interactions with wireless networks, sensor networking, fast protocol stack implementations, RAID system design, performance prediction for Internet transfers, firewall design, and improvements to the TCP protocol. The following two projects are examples of my current research efforts.

Next Generation Network Architectures. While the Internet has been a great success and supported a wide range of interesting applications, its design is beginning to show its age. My current research is exploring new network protocols and entire network architectures that address the network’s shortcomings and provide better support for the next generation of Internet applications. The eXpressive Internet Architecture (XIA) project is a part of this work that explores a clean-slate redesign of the Internet architecture. The primary goal of the XIA architecture is to create a robust and reliable network that easily supports the evolution of in-network functionality over time. In addition to XIA, my research also explores implications of the fact that video content delivery now dominates all forms of traffic on the Internet and is likely to continue dominating for the foreseeable future. My group is exploring both new protocols, content delivery mechanisms and network management techniques that both address the challenges and take advantage of the opportunities created by this flood of video traffic.

Wireless Networks and Mobile Systems. Over the past few years smartphones have gone from being a relatively rarity to becoming nearly ubiquitous. These phones are one of the few items that we take almost everywhere. People use phones to take photos, make appointments, find restaurants, browse the Web and keep in touch with their social and business contacts. The information that we reveal to these smartphones and their ability to make observations about their surroundings give them the unique ability to make observations about our lifestyles and our environment in a highly detailed fashion. My research explores the systems challenges in making these observations useful and accessible. This includes designing the software on devices to accurately collect data in a power-efficient fashion, designing changes to the infrastructure to enable the efficient and scalable collection of data from handsets, and managing the privacy issues that arise from this data collection.

MARY SHAW
Alan J. Perlis University Professor, Institute for Software Research, Computer Science

Software now accounts for the lion's share of the cost of developing and using computer systems. My long-term goal is to establish a genuine engineering discipline to support the design and development of software systems. Currently I’m working on design methods and analytic techniques for building complete software systems out of subsystems and their constituent modules. This is the software architecture level of design. I am particularly interested in value-based techniques for making good design choices early in the design process. My current research investigates two aspects of architectural design for software.

Value-Driven Software Design. Current software design concepts largely overlook a simple but fundamental idea: the goal of software design decision making is to create the maximum value added for any given investment of valuable resources. Businesses value profit, but also opportunities, as seen in valuations of profitless Internet companies. Philanthropic foundations value solutions to social problems. Universities value creation and dissemination of knowledge. End users value hassle-free access to information and a sense of control over their applications.
Software design decisions today are made in an economics-independent Flatland, where concerns for technical properties dominate. Past work on software economics is relevant but it focuses on cost minimization, rather than value maximization. This research pursues scientific foundations for software design decision-making approaches that are explicitly tied to value-maximization objectives. It explicitly balances costs and benefits as seen by particular stakeholders, and it emphasizes models, methods and tools that can be applied early in the design process, before code is available for analysis.

**Open Resource Coalitions.** Widespread use of the Internet is enabling a fundamentally new approach to software development: computing through dynamically formed, task-specific, coalitions of distributed autonomous resources. The resources may be information, calculation, communication, control, or services. Unlike traditional software systems, which are at least nominally under control of the designer, these coalitions are formed from autonomous network-based resources, and the developer lacks direct control over the incorporated resources. These autonomous resources are independently created and managed. The resources may be transient, either because of the resource proprietors actions or because of service interruptions; indeed, the proprietor of a resource may be unaware of the ways the resource is used. Development tools for resource coalitions will require new degrees of autonomy and automation in order to identify, compose, and track the resources. Computing through resource coalitions will thus create novel architectural challenges and opportunities.

Achieving useful results from such resources requires a new level of openness in the sense that responsibility for individual resources is distributed much more widely than responsibility for the results. The aggregations of resources are better treated as coalitions than as systems, because individual resources are operated under their own policies, and it may be necessary to reconstitute the coalition when the selection of available resources changes.

This style of software creation is of particular significance for everyday users who roll their own individually tailored applications from available resources.

**DANIEL SIEWIOREK**
Buhl University Professor, Computer Science, Electrical and Computer Engineering
Director, Quality of Life Technology Center
http://www.cmu.edu/qolt/

My major interest is in the modular design and rapid prototyping of dependable computing structures. Three research projects support this interest: Mobile/Wearable/Context Aware Computers and Virtual Coaches, Concurrent Design, and Reliable Systems.

Mobile/Wearable/Context Aware Computers and Virtual Coaches. The information processing industry is undergoing a paradigm shift. In the 1990's wearable computers allow mobile users to remotely access information and collaborate with exerts. We have built over two dozen generations of mobile and wearable computer systems in such diverse areas as heavy vehicle maintenance, aircraft manufacturing, plant operations, language translation, and medical monitoring. Systems involve hardware architecture, software architecture, wireless communications, interaction between energy consumption and functionality, ergonomic design, and human computer interaction. By adding low cost sensors and machine learning algorithms, small platforms such as smart phones and wristwatches can be made context-aware and respond proactively to situations based upon learned user preferences. User intention, situation, and emotional state can be inferred and context appropriate real-time feedback provided to engage, encourage, and improve quality of life in such diverse activities as rehabilitation exercises, completing complex tasks, and learning something for the first time. Research is in cooperation with the School of Design, Electrical and Computer Engineering, and the Human Computer Interaction Institute.
Concurrent Design. The goal is to support the generation of designs from high level systems specifications into completely assembled electronic, mechanical, user interaction, and software systems and reduce design time by 1 to 2 orders of magnitude. The Concurrent Design methodology has been used in all generations of mobile systems described above.

Groups of up to 30 designers representing up to five disciplines design and fabricate functional prototypes in less than four months, and develop tools to support the concurrent design process. For all levels of design there are common issues that must be addressed including design data bases, design information representation, human-computer interfaces, simulation/validation/ verification, automatic synthesis, test generation, system evaluation, and design selection criteria.

Reliable Systems. For over three decades, computer system design and evaluation has been based upon performance benchmarks. Comparable benchmarks do not exist for evaluating the quality and robustness of computer hardware/software systems. This project is developing a family of portable benchmarks for a variety of operating systems and programming language environments. The benchmarks are based upon over a decade of experimentation with fault injection including the next generation air traffic control system. Another goal of the project is to develop technology to enable the construction of reliable systems from commercial-off-the-shelf (COTS) hardware and software. Studies indicate that the majority of system downtime is due to human errors in either design or operation. This project also explores the design of software systems and interfaces to reduce human errors. Research is in cooperation with Roy Maxion and Phil Koopman.

Web site: www.cs.cmu.edu/~dps

REID SIMMONS
Research Professor, Computer Science, Robotics

My research focuses on the area reliable autonomous systems (especially mobile robots) operating in rich, uncertain environments. The goal is to create intelligent systems that can operate autonomously for long periods of time in unstructured, natural environments. This necessitates robots that can plan, reason about uncertainty, diagnose and recover from unanticipated errors, interact with other robots and humans, and learn. I am interested in the use of probabilistic reasoning to explicitly model and plan for uncertainty. I am currently focusing on multi-robot coordination and human-robot social interaction.

*Multi-Robot Coordination*. We are researching issues of how multiple, heterogeneous robots can coordinate to carry out high-level tasks, especially those that cannot be accomplished by a single robot. Issues include having the robots negotiate to dynamically form teams and assign tasks, monitoring each other's performance, and adapting dynamically to changing situations. The work extends traditional three-tiered architectures to multiple robots and extends planning and scheduling algorithms to handle uncertainty and failing plans. Application domains include dexterous, multi-robot large-scale assembly.

*Human-Robot Social Interaction*. The goal here is to make robots more useful and acceptable by enabling them to interact with humans using social rules and conventions. This includes both conversational and spatial social interaction. In the area of conversational interaction, we have developed the roboceptionist, in conjunction with the School of Drama, to have a robot with character and personality interact over long periods of time. In the area of spatial interaction, we are looking at socially appropriate navigation and rhythmic interaction. We are also interested in culturally appropriate interaction.

*Probabilistic Reasoning in Robotics*. We are exploring the use of probabilistic reasoning techniques in controlling autonomous robots. We are investigating methods for making mobile robots more robust and self-reliant, especially using Markov models and partially observable Markov models.
I have worked in a variety of different areas of computer science, including amortized analysis of algorithms, self-adjusting data structures, competitive algorithms, natural language parsing, computer game playing, synthesis of musical sounds, and persistent data structures.

**Natural Language:** I (jointly with co-author Davy Temperley) wrote a parser for English. The system (which we call a link grammar) is unlike phrase structure parsing or context free parsing. The scheme is elegant and simple, and our grammar captures a very wide variety of complex phenomena in English. We (John Lafferty and I) plan to use this as a basis for a new statistical model of language. This work on language is described in two technical reports: CMU-CS-91-196, CMU-CS-92-181.

**Competitive Algorithms:** Consider the idealized problem of deciding whether to rent or buy skis. You're about to go skiing. The cost of renting skis is $20, the cost of buying them is $400. Clearly if you knew that you were going to go skiing more than twenty times, then you could save money by immediately buying skis. If you knew that you would go skiing fewer than twenty times, then it would be prudent to always rent skis. However, suppose that you cannot predict the future at all, that is, you never know until after one ski trip ends if you will ever go skiing again. What strategy would you use for deciding whether to rent or buy skis? Your goal is to minimize the ratio of the cost that you incur to the cost you would incur if you could predict the future. (Hint: you can come within a factor of two.)

Since the simple principle behind this example turns out to be very useful we have given it a name. A competitive algorithm is an on-line algorithm (it must process a sequence of requests, and it must process each request in the sequence immediately, without knowing what the future requests will be), whose performance is within a small constant factor of the performance of the optimal off-line algorithm for any sequence of requests. (In the skiing example, there is only one type of request, and the only uncertainty is in knowing how long the request sequence will be.)

My collaborators and I have discovered a surprising variety of practical problems for which there exist very efficient competitive algorithms. We have also developed a partial theory of competitive algorithms. However there remain many interesting open problems, from discovering competitive algorithms for specific problems, to answering general questions about when such algorithms exist.

**Data Structures:** Data structure problems are typically formulated in terms of what types of operations on the data are required, and how fast these operations should take place. A worst-case analysis of the performance of a data structure is a bound on the performance of any operation. An amortized analysis of a data structure bounds the performance of the structure on a sequence of operations, rather than a single operation. It turns out that by only requiring amortized efficiency (rather than worst-case), a variety of new and elegant solutions to old data structure design problems become possible. My collaborators and I have devised a number of such solutions (splay trees, skew heaps, fibonacci heaps, self-adjusting lists, persistent data structures, etc.), and I continue to have a strong interest in data structures and amortized analysis.

---

**PETER STEENKISTE**

Professor, Computer Science, Electrical and Computer Engineering

My research interests are broadly speaking in computer networking and distributed systems. My research is very applied: we build prototype systems that incorporate our research ideas and we evaluate these systems using real applications in a realistic setting. I currently have active research projects in the following areas:

Future Internet Architecture – The architecture of today’s Internet is more than 30 years old. Of course, the network has changed significantly in terms of scale, technologies, applications, and traffic. An interesting research question is whether we can define a new network architecture that will not only better support today’s Internet, but that can also better support future changes in technology and
usage. We have proposed a new architecture, called the eXpressive Internet Architecture or XIA. Within the context of XIA and today's Internet, we are now explore a wide range of networking research topics including congestion control, mobility, network diagnostics, and routing.

Wireless Networking – Wireless networks have become ubiquitous and most of us use a variety of wireless technologies on a daily if not hourly basis. This rapid growth in usage and deployment, combined with the emergence of new technologies such as smart antennas and software radios, has made wireless an incredibly exciting research area. The wireless networking projects I am involved in cover a wide range of topics including self-managing and self-optimizing wireless deployments, dynamic spectrum access, and vehicular networks.

DAVID TOURETZKY  
Research Professor, Computer Science, Neural Basis of Cognition

My current research focuses on robotics and computing education. I have previously done work in computational neuroscience, connectionist modeling, and knowledge representation.

Tekkotsu Framework for Robotics Education

When robots incorporate vision systems and high degree-of-freedom manipulators, simple reactive programming strategies no longer suffice. How can state of the art robot programming be made accessible to beginners? Tekkotsu (“framework” in Japanese) is an open source software framework used to introduce computer science undergraduates to essential robotics topics such as computer vision, landmark-based navigation, path planning, and manipulation. I am interested in how robot programming systems can be made more sophisticated while remaining transparent and easy to experiment with.

The Tekkotsu software and curriculum materials are used by a number of universities, as well as in my own course here at CMU. Observing how students actually use the software allows us to refine it and explore new approaches to high level robot programming.

Teaching Children Programming Via Idioms

In computer programming, idioms are familiar bits of code that perform common or important functions. A programming language’s level of abstraction determines the kinds of idioms it affords. "Incrementing a counter" is a low level idiom common to most languages. "Pursue and consume" is an important idiom in Kodu, a programming language designed for young children that provides high level primitives inspired by behavioral robotics. I'm working on developing a Kodu-based curriculum for K-6 students that explicitly teaches idioms and also places a heavy emphasis on finite state machines. Acquiring the concept of state machines at an early age is an important step toward computational thinking, and can equip a child to master a wide range of technologies.

Creating Affordable Mobile Manipulators

One of the obstacles to improving undergraduate robotics education has been the lack of affordable mobile manipulators. I have been developing mobile robots with serious vision and manipulation capabilities that are cheap enough to use in a classroom. The first was the Chiara hexapod, which went on to play chess at AAAI and even play a bit of Bach on YouTube. More recently, the Calliope2SP robot, which includes a 2-dof arm and a webcam on a pan/tilt, with an iRobot Create as the base, has become the standard platform for the ARTSI Robotics Competition, part of an NSF-funded Broadening Participation in Computing project. The Calliope5KP is a more powerful version that incorporates a 5-dof arm and a Kinect. I'm currently working on a new hexapod robot inspired by the praying mantis insect, and beginning to incorporate 3D printing technology into my designs.
ADRIEN TREUILLE
Assistant Professor, Computer Science, Robotics

Dr. Treuille studies the interactive simulation of very high-dimensional non-linear phenomena like deformable objects, human motion, crowds, and large fluid systems. One thread of this research addresses the complexity of such systems by developing model reduction tools that generate compact representations. A complimentary thread seeks to control such systems, which means learning to set inputs to produce desired effects.

Dr. Treuille is also one of the principle creators of Foldit. By having fun playing the game, some 50,000 people around the world are actually working together helping to solve important problems in biochemistry.

MANUELA VELOSO
Herbert A. Simon Professor, Computer Science

I research in the area of Artificial Intelligence. My long-term research goal is the effective construction of autonomous agents where cognition, perception, and action are combined to address planning, execution, and learning tasks.

I am interested in the continuous integration of reactive, deliberative planning and control learning for teams of multiple agents acting in dynamic and uncertain environments.

I am interested in adversarial modeling, reuse, and abstraction in control learning for multiple agents. I also continue to investigate effective planning, execution, and learning algorithms for deterministic and nondeterministic multiagent domains within the research projects CORAL (Collaborate, Observe, Reason, Act, and Learn), MAPEL (Multi-Agent Planning, Execution, and Learning), and the MultiRobot Lab.

With my students, I have used robotic soccer as a concrete testbed for research. We have developed teams of robotic soccer agents in three different leagues that have been RoboCup world champions several times: simulation (1998,1999), CMU-built small-wheeled robots (1997,1998), and Sony four-legged robots (1998).

I also research on the integration of planning and information retrieval, and the application of evolutionary computation and machine learning to the performance prediction of signal processing algorithms.

LUIS VON AHN
Associate Professor, Computer Science

At the height of its construction, 44,733 people worked on the Panama Canal. The Great Pyramid of Giza required 50,000 workers and the Apollo Project 400,000. No matter what you put on this list, humanity’s largest achievements have been accomplished with less than a few hundred thousand workers because it has been impossible to assemble (let alone pay!) more people to work together—until now. With the Internet, we can coordinate the efforts of billions of humans. If 400,000 people put a man on the moon, what can we do with 100 million? My research aims to develop theories and build computer systems that enable massive collaborations between humans and computers for the benefit of humanity. I am working to develop a new area of computer science called human computation, which studies how to harness the combined power of humans and computers to solve problems that would be impossible for either to solve alone.
HOWARD D. WACTLAR
Alumni Research Professor, Computer Science, Robotics

My current research interests are in information retrieval systems, multimedia, speech/image/natural language understanding and distributed systems. My emphasis in multidisciplinary projects has led to my current large project, The Informedia Digital Video Library, which integrates several CS areas.

The Informedia Digital Video Library. Vast collections of video and audio recordings which have captured events of the last century remain a largely untapped resource of historical and scientific value. Increasingly, such libraries are becoming available on the Internet. The Informedia Digital Video Library project at Carnegie Mellon University has pioneered new approaches for automated video and audio indexing, navigation, visualization, search and retrieval and embedded them in a system for use in education, information and entertainment environments. Initiated in 1994 as one of six Digital Library Initiative (DLI) projects funded jointly by NSF, DARPA and NASA, Informedia is the only one focused on the video medium.

The Informedia system provides full-content search and retrieval of current and past TV and radio news and documentary broadcasts. The system implements a fully automated process to enable daily content capture, information extraction and storage in on-line archives by applying artificial intelligence and advanced systems technology. The current library consists of more than 1,500 hours (one terabyte) of digitized daily news captured over the last two years, and documentaries produced for public television and government agencies. This prototype database allows for rapid retrieval of individual video paragraphs which satisfy an arbitrary spoken or typed subject area query based on the words in the soundtrack, closed-captioning or text overlaid on the screen. There is also a capability for matching of similar faces and images.

Our newest phase of research continues the fundamental goal of enabling for video all the functionality and capability existing for textual information retrieval, while leveraging its temporal and visual qualities for richer information delivery. To this end, Informedia-II establishes an era focused for the user as we introduce new paradigms for video information access and understanding. We aggregate and integrate video content on-demand to enable summarization and visualization that provides responses to queries in a useful broader context, perhaps with historic or geographic perspectives.

Diverse technologies and disciplines applied and integrated within Informedia include:

- Speech understanding for automatically derived transcripts and spoken queries
- Image understanding for video paragraphing (segmentation) and similarity matching
- Natural language for processing transcripts and queries
- HCI for creating novel user interfaces for video display, manipulation, and reuse
- Network authentication and billing for controlled access
- Data architectures for network retrieval-on-demand
- User studies for validation and assessment

Additional research efforts underway apply Informedia technology to the domains of education, health care, defense intelligence and the coordination and understanding of human activity. Informedia also has international digital library collaborations in Europe and Asia.
A long-term goal that drives my research is to provide, where appropriate, as rigorous as possible a foundation to the design and implementation of software. My specific interest is the application of formal specification and analysis techniques to reason about complex software systems.

My current research interests focus on trustworthy computing: reliability, security, privacy, and usability. For many years, I have worked on reliable software; in recent years, I have worked on security. I am now mostly interested in the foundations of security and privacy. Much as we have algorithms, impossibility results, and formal methods for building reliable, distributed systems, I am interested in laying similar kinds of foundations for security and privacy. For security, I seek compositional techniques and properties that allow one to reason locally in order to build large systems with clean interfaces and well-defined behavior. I am now also interested in the technical challenges of privacy, e.g., how to specify and reason about privacy policies, how to determine whether a software system is compliant with a given privacy policy. I am particularly keen on investigating formal logics for reasoning about privacy and applications in healthcare.

Here is a sample of some current projects:

* Information Flow Experiments: How can one detect what data is being used by a website? To answer this specific question, Michael Tschantz, Anupam Datta, and I formalize information flow analyses where the analyst has neither control nor a complete model of the analyzed system. We prove that generalizations of our specific problem are ones of causal inference. Leveraging this connection, we push beyond traditional information flow analysis to provide a systematic methodology based on experimental science and statistical analysis.

* Science of Security: Rather than the band-aid approach we currently use to patch our systems after we detect an exploit of a security vulnerability, suppose we could build software systems based on theories, laws, and principles with predictable value with respect to security properties? Predictability requires security modeling, e.g., game-theoretic models, and security analysis, including quantitative, not just qualitative analysis. To address scalability, I am interested in understanding when security properties do or do not compose, and how to achieve compositionality. I am also interested in security metrics.

* Trust in Networks of Humans and Computers: The advent of social media gives rise to this fundamental question "How can I (a human) trust the information I receive through the Internet?" Virgil Gligor and I are exploring the combination of trustworthiness from computer science the concept of behavioral trust from economics. Our goal is to build a general theory of trust for networks of humans and computers.

* Information Flow

My past research projects include:

1. Larch, a family of specification languages noted for its two-tiered approach to specifying program modules;

2. Miro, a visual specification language for specifying file system security constraints;

3. Avalon, extensions to C and CommonLisp to support distributed transactions;

4. Venari, extensions to Standard ML to support concurrent multi-threaded transactions; and signature and specification matching of software components.

5. TinkerTeach, software infrastructure for electronic delivery of courseware, but best known for providing the TOM conversion service which is in heavy daily use worldwide;
6. Revere, invention of a fully automatic analysis technique called “theory generation,” as applied to authentication protocols and simple electronic commerce protocols (the thesis topic of my former student, Darrell Kindred);

7. Verifiable Secret Redistribution, a protocol for recovering shared secrets in a survivable storage system (the thesis topic of my former student, Ted Wong).

8. Attack Graphs, tools for generating attack graphs automatically through an all counterexamples extension of model checking

9. Attack Surface Security Metric, a method for measuring the “attack surface” of a software system, useful for comparing whether one version of a system is "more secure" than another (the thesis topic of my former student, Pratyusa Manadhata)

10. Privacy Specification and Compliance: Formalization "use" and "purpose," notions common in privacy policies, in terms of planning modes from artificial intelligence (the thesis topic of my former student, Michael Tschantz, co-advised with Anupam Datta)

Personal web site: www.cs.cmu.edu/~wing/

HUI ZHANG
Professor, Computer Science, Electrical and Computer Engineering

My research interests include Internet-scale peer-to-peer systems, scalable QoS and multicast solutions for the Internet, metro networking technology, and network management. My work involves both designing practical algorithms with sound theoretical foundations and building prototype systems in real life environments.

More details can be found at http://www.cs.cmu.edu/~hzhang.
AFFILIATED FACULTY

JONATHAN ALDRICH
Associate Professor, Institute for Software Research and Computer Science
My research goal is to improve the quality and security of software and the productivity of engineers by providing novel ways to express and enforce structural and behavioral aspects of software design within source code, typically through language design and type systems. I both prove formal properties of the systems I develop, and also evaluate their practical benefits via case studies, human subjects experiments, and corpus studies with open source codebases.

A Programming Language for Secure Web and Mobile Development. We are currently designing a new programming language called Wyvern that is designed to help in developing highly secure web and mobile applications, while at the same time enhancing developer productivity. Research on this project includes a novel object model and versioned module system that provides a foundation for security, and an extensible language and type system, which supports domain-specific languages for UI layouts, distributed application architectures, and security policies.

Lightweight Formal Methods show great promise for helping software engineers write secure software, avoid defects, and achieve high parallel performance and other non-functional goals. I am interested in how language and type system design can be used to more effectively check a range of critical software properties. We are currently developing an approach, for example, that provides a simple typed interface to a parallel library, while using more sophisticated formal methods to prove that subtle concurrency tricks used in the library are correct. This approach may thus be able to provide both high usability and high performance—an elusive combination in past work. More broadly, I am interested in using formal abstractions that capture engineers’ informal reasoning, especially in the presence of challenges such as concurrency and mutable aliased state.

Web page: http://www.cs.cmu.edu/~aldrich/

DAVID BRUMLEY
Assistant Professor, Electrical and Computer Engineering, Computer Science
My primary research area is software security techniques that give users guarantees. My research is at the intersection of model checking, formal methods, compilers, and logic, all of which are applied to security problems. Some of the central techniques that we are developing in order to accomplish our goals include efficient symbolic execution, reasoning about bit-level arithmetic in finite fields, sound decompilation, and decision procedures. I also work in other areas of computer security, such as network security and applied cryptography. In these areas, we look at efficient protocols, efficient signature schemes, and privacy-preserving cryptography.

Current questions we are working on include can we automatically generate exploits? Can we automatically develop recognizers for exploits of a particular vulnerability? What are the limits of reverse engineering? Can we show that Microsoft implements RSA correctly, and that they aren't stealing our secrets? Can we show a crypto algorithm doesn't leak secrets via side-channels? What do we need to change to keep software patches from breaking systems? How do things change in critical systems, e.g., securing an MRI machine in a hospital? Would you want to be the first to get an MRI after a software update?

In our setting, we typically assume the software in these settings is only available in binary (i.e., executable) form. Binary code analysis is attractive for several reasons. First, binary-level code analysis allows us to argue about the security of the code that actually executes, not just what was compiled. Second, everyone has access to the programs they run in binary form, thus binary-only techniques promise to be widely applicable. Finally, a binary program is a program in the most basic
and primitive form. If we can reason about security concerns at the binary level, we can faithfully reason about software security problems in any language.

---

**GREG GANGER**  
Professor, Electrical and Computer Engineering, Computer Science (c)

I have broad research interests in computer systems, including cloud computing, storage/file systems, operating systems and distributed systems. I am particularly interested in developing new ways of structuring computer systems to address technology changes and enable new applications. I am involved in many ongoing projects in such areas as data-intensive computing, distributed system diagnosis, cloud computing, home storage, and exploitation of new storage technologies.

**Cloud Computing Infrastructure**

We are exploring the many systems challenges hidden behind the hype surrounding cloud computing, such as elastic storage, resource allocation/scheduling, and automation (e.g., performance problem diagnosis). In addition, to gain first-hand experiences, we maintain and measure several deployed cloud resources with real users, in collaboration with industry partners.

**Data-intensive Computing (DISC, a.k.a. Big Data)**

DISC refers to the rapidly growing style of computing characterized by extraction of information from huge and often dynamically growing datasets. We are exploring new distributed storage and computing system designs for achieving robust, efficient data-intensive computing. We are particularly interested in new frameworks for supporting advanced machine learning on large data sets, shifting the work of coordinating parallel threads and data from the programmer without losing efficiency.

**Parallel Data Lab (PDL)**

As Director of the Parallel Data Lab, I lead a number of storage-related projects in areas such as storage system architecture, survivable storage, file systems, storage security, and automated storage management. As one example, we are exploring how system software should change to accommodate new storage technologies like non-volatile RAM (e.g., PCM) and shingled magnetic recording (in disks). As another, we are exploring new approaches to sharing and access control in distributed home storage.

Personal Web Page: [http://www.ece.cmu.edu/~ganger/](http://www.ece.cmu.edu/~ganger/)

---

**ONUR MUTLU**  
Strecker Early Career Professor, Electrical and Computer Engineering, Computer Science (c)

My research interests are in computer architecture, especially the hardware/software interface and cooperation to improve the design of parallel system architectures. I am interested in designing novel and efficient hardware/software cooperative techniques to overcome fundamental performance, security, robustness, reliability, and efficiency challenges in current and future computer systems. I am especially interested in understanding and improving the interactions between hardware and My major research interests are in computer architecture and systems.

I am interested in designing novel and efficient hardware/software cooperative techniques to overcome fundamental performance, energy-efficiency, robustness, predictability, and reliability challenges in current and future computer systems and applications. To this end, my research spans both the hardware and the software layers, from circuits to system/application software, with a focus on computer architecture.

I am involved in many different projects towards the same end, focusing on the design and
management of innovative computation, communication, and memory/storage architectures. I am interested in exploring the limits of efficiency and performance we can achieve with old and new technologies and with hardware/software/device cooperation and co-design.

A major current focus of mine is to invent computer platforms that can efficiently, quickly and reliably store, manipulate and communicate massive amounts of data. To do this, we are rethinking how to design the three fundamental functions of all computers, computation, storage (memory) and communication, at both the hardware and the software levels.

Another major current focus of mine is to design systems and devices that can speed up medical and biological applications, such as DNA sequence analysis, by orders of magnitude. A goal of my research in this area is to design robust devices that can cost-effectively enable personalized medicine.

Some of the major projects/topics my group focuses on are listed below, with links when appropriate. Please note that these projects change and evolve, so it is best to contact me for up-to-date information.

- Memory Systems
  Summary paper from 2013 with pointers to example research ideas and directions:
  Memory Scaling: A Systems Architecture Perspective

- Heterogeneous Computing, Accelerator Based Computing
  Position presentation from 2010:
  Asymmetry Everywhere
  Example papers in ASPLOS 2009-2012, ISCA 2010-2013, DAC 2013, ICCD 2012, MICRO 2010

- Extremely Energy-efficient Systems
  Position paper from 2012:
  Some Ideas and Principles for Achieving Higher System Energy Efficiency

- Enabling and Exploiting Emerging Technologies
  Example position paper from 2013:
  A Case for Efficient Hardware-Software Cooperative Management of Storage and Memory

- Predictable and QoS-aware Many-core Systems
  Example papers:
  Memory Performance Attacks: Denial of Memory Service in Multi-Core Systems, USENIX Security 2007
  MISE: Providing Performance Predictability and Improving Fairness in Shared Main Memory Systems, HPCA 2013

- Novel algorithms and computer architectures for health, biological, medical, and bioinformatics applications
  Example papers in Nature Genetics 2009, BMC Genomics 2013

- GPU Architectures, General-purpose GPU Systems
- Efficient interconnection networks
- Novel computer architectures for cloud computing
- Architectural support for robust and secure operating systems (OS), OS/architecture interaction
- Architectural support for high-level/safe/managed programming languages (PL), PL/architecture interaction
- Fault and bug tolerant architectures
- Latency-tolerant architectures
- Resource management, fairness, QoS in systems

For more information, please visit the following pages:

My website:  http://users.ece.cmu.edu/~omutlu
My publications: http://users.ece.cmu.edu/~omutlu/projects.htm

RUSSELL SCHWARTZ
Associate Professor, Biological Sciences, Computer Science

My research is broadly in the area of computational biology, with particular emphasis on models and algorithms for studying complex systems in biology. My group is currently pursuing two major areas:

Genetic variation analysis. Since the human genome was first sequenced in 2002, the field of human genetics has turned its attention to identifying the millions of small differences that distinguish one human being from another. Most of these differences are in the form of single DNA bases that vary from one person to another, which are called single nucleotide polymorphisms (SNPs). My group has worked on models and algorithms to analyze large datasets of SNPs and infer evolutionary trees (phylogenies) and more complex population models that tell us how modern human populations arose from the earliest human ancestors and how our genome has evolved over that time. Our largest area of research in recent years has been the development of similar phylogenetic and population genetic methods to study evolution of cell populations in tumors.

Algorithms for macromolecular assembly simulation. One of the recurring features of molecular biology is self-assembly, a process by which isolated molecules spontaneously join together to build structures or molecular machines. Self-assembly is required for nearly every important function a cell undergoes, including division, movement, shape control, and synthesis and degradation of DNA, RNA, and proteins. Biological self-assembly systems are also an important model for the development of novel nanotechnology. They are, however, very challenging to standard methods for simulating biochemistry because of their large size and the long time scales on which they operate. My group develops algorithms to accelerate stochastic models of these assembly systems, builds simulation systems based on these algorithms, and applies them to investigate properties of assembly systems that are difficult to explore through laboratory experiment. In recent years, we have particularly been interested in combining such models with numerical optimization algorithms to fit stochastic models to experimental data and, in the process, learn how these systems function at much finer scales than can be measured experimentally.

In addition to these two core areas, we are involved in many side projects, usually in collaboration with experimentalists. Over the past few years, these projects have included work on modeling biomedical systems, developing more realistic models of biochemistry in the cell, and developing new methods for deconvolving complex genomic data sets from heterogeneous cell populations, primarily with application to cancer genomics. These projects draw on a wide variety of computational tools from discrete algorithms, operations research, applied mathematics, statistics, and machine learning.

RICHARD STERN
Professor, Electrical and Computer Engineering, Computer Science

My research interests involve a number of topics joined by the common threads of signal processing, sound, and acoustics. At present I am most actively working on topics related to automatic speech recognition and signal processing in the auditory system. I have also been involved in projects in the areas of biomedical instrumentation, particularly with regard to the auditory system, physical acoustics, computer music, and computer-aided instructional systems.
**Automatic Speech Recognition.** The SCS speech group is developing speech technology that can perform unlimited-vocabulary speech recognition on a speaker-independent basis under difficult acoustical conditions. We are also developing practical applications that make use of spoken language interfaces to perform useful tasks.

The major goal of my own work speech research is to enable CMU's SPHINX recognition system to become as robust to changes in acoustical environment and ambience as it is to changes in speaker. In particular, we must deal with problems in recognition accuracy resulting from additive noise sources, background music, competing talkers, change of microphone, and room reverberation. We are developing several different types of solutions for these problems including improved noise cancellation and speech normalization methods, the use of representations of the speech waveform that are based on the processing of sounds by the human auditory system, and the use of arrays of microphones to improve signal-to-noise ratio. In previous knowledge-based speech-recognition systems I had also worked on statistical classification, speaker adaptation, and the integration of syntactic, grammatic, and semantic information.

**Signal Processing in the Auditory System.** The general goal of this research has been to develop a better understanding of how the auditory system processes sound, to apply this knowledge to the treatment of various kinds of hearing impairments, and to apply this knowledge to the development of more robust speech recognition systems. I am presently carrying out psychoacoustical measurements of various aspects of monaural and binaural perception, and developing models based on communications theory and linear system theory to relate the results of these experiments to neural coding of sounds by the auditory system. Most of my work in hearing has been concerned with the localization of sound and other aspects of binaural perception.

---

**ERIC P. XING**  
Professor, Machine Learning, Language Technologies, Computer Science

The major theme of Professor Xing's research lies in the development of machine learning and statistical methodology; especially for building quantitative models and predictive understandings of the evolutionary mechanism, regulatory circuitry, and developmental processes of biological systems; and for building intelligent systems for a wide range of applications in vision, IR and NLP that involves computational learning and reasoning under uncertainty.

**Foundations of Statistical Learning**, including theory and algorithms for: 1) Time/space varying-coefficient models with evolving structures; 2) Sparse structured input/output models in high-dimensional problems; 3) Nonparametric Bayesian techniques for infinite-dimensional models; 4) RKHS embedding, nonparametric inference, and spectral methods for graphical models; 5) Distributed and online algorithms for optimization, approximate inference, and sampling on massive data.

**Large-scale Information & Intelligent System**: 1) Development of scalable parallel architecture, protocol, programming interface, generic algorithms and models, for Big Learning; 2) Multi-view latent space models, topics models, and sparse coding for image/text/relational data mining; 3) Evolving structure, stable metrics, and prediction for dynamic social networks, goal-driven network design and optimization; 4) Web-scale image understanding, search, prediction, and storyline synthesis; 5) Information visualization, indexing and storage, web/mobile app development.

**Computational Biology**: 1) Understanding genome-microenvironment interactions in cancer and embryogenesis via joint analysis of genomic, proteomic, and pathway signaling data; 2) Genetic analysis of population variation, demography and evolution; 3) Statistical inference of genometranscriptome-phenome association in complex diseases; 4) Personalized diagnosis and treatment of spectrum diseases via next generation sequencing and computational "omic" analysis; 5) Biological image and text mining.
FACULTY BY INTERESTS

**Adaptive systems**: Garlan, Maxion, Schmerl
**Abstraction (cf. Type Theory)**: Aldrich, Garlan, Platzer, Shaw, Wing
**Algorithmic mechanism design**: Balcan, A. Blum, Procaccia, Sandholm
**Algorithms**: Acar, Balcan, Blelloch, A. Blum, L. Blum, M. Blum, Bryant, Gupta, Guruswami, Haeupler, Harchol-Balter, Kolter, Miller, O'Donnell, Procaccia, Rudich, Sandholm, Sleator, Xing
**Algorithms, approximation**: Balcan, A. Blum, Gupta, Guruswami, Miller, O'Donnell, Procaccia
**Algorithms, distributed**: Acar, Blcloch, Fink, Goldstein, Haeupler, Harchol-Balter, Rudich
**Algorithms, graph**: Acar, Balcan, Blelloch, A. Blum, Gupta, Guruswami, Haeupler, Miller, O'Donnell
**Algorithms, learning**: Balcan, A. Blum, Carbonell, Fink, Kolter, O'Donnell, Xing
**Algorithms, on-line**: A. Blum, Haeupler
**Algorithms, networks**: Gupta, Haeupler
**Algorithms, parallel**: Acar, Blelloch, Miller
**Algorithms, robotics**: Erdmann
**Analogue reasoning**: Carbonell, Mitchell, Veloso, Xing
**Animation**: Hodgins, Pollard
**Anomaly detection**: Maxion, Shaw
**Application-specific computing systems**: Garlan
**Artificial Intelligence**: Brunskill, Carbonell, Fahiman, Fink, Kolter, Mason, Mitchell, Procaccia, Reddy, Sandholm, Veloso, Xing
**Aspect-oriented programming**: Aldrich
**Assurance cases**: Maxion
**Auctions**: Harchol-Balter, Sandholm
**Auditory modeling**: Stern
**Automata theory**: M. Blum
**Automated negotiation**: Sandholm
**Automated theorem proving**: Brumley, Clarke, Pfenning, Platzer, Wing
**Automatic hardware generation**: Goldstein
**Automatic parallelization**: Aldrich, Goldstein
**Automatic programming**: See Program Synthesis.
**Autonomic computing**: Garlan, Ganger, Gibson, Maxion, Schmerl
**Autonomous mobile robots**: Kanade, Simmons
**Autonomous rovers**: Simmons
**Autonomous spacecraft**: Simmons
**Autonomous systems**: Kanade, Platzer, Simmons

**Bayesian networks**: Xing
**Big Data**: Acar, Balcan, Blelloch, Bryant, Ganger, Gibson, Miller, Pavlo
**Biology, computational**: Erdmann, Xing
**Biometrics**: Maxion
**Bounded rationality**: Sandholm
**CareMedia**: Hauptmann, Wactlar
**Causal reasoning**: Carbonell, Maxion
**Cellular phone systems and infrastructure**: Seshan
**Cloud computing**: Andersen, Ganger, Gibson, Goldstein, O'Hallaron, Pavlo, Satyanarayanan
**Coalition formation**: Sandholm
**Cognitive robotics**: Touretzky
**Cognitive science**: Maxion
**Combinatorics**: Haeupler, Miller, O'Donnell, Rudich, Sleator
**Compilers**: Acar, Blelloch, Brumley, Crary, Goldstein, Harper, Platzer
**Complexity and real computation**: L. Blum, Platzer
**Complexity theory**: M. Blum, Guruswami, O'Donnell, Rudich
**Computation theory**: Brookes, Clarke, Miller, Platzer, Rudich
Computational geography: Miller
Computational group theory: O'Donnell
Computational learning theory: Balcan, A. Blum, O'Donnell, Procaccia
Computational linguistics: Carbonell, Fahlman, Hauptmann, Rudnicky, Xing
Computational molecular biology: Durand, Erdmann, Xing
Computational neuroscience: T. Lee, Touretzky
Computational social choice: Procaccia, Sandholm
Computational statistics: Xing
Computational sustainability: Kolter
Computational topology: Erdmann
Computer-aided design: Bryant, Clarke, Garlan, Platzer, Siewiorek
Computer architecture: Acar, Blelloch, Fatahalian, Ganger, Goldstein, Hoe, Mutlu, Siewiorek, Wactlar
Computer music: Dannenberg
Computer security: Brumley, Datta, Wing
Computer systems: Andersen, Fatahalian, Gibson, Goldstein, Mowry, O'Hallaron, Pavlo, Satyanarayanan, Seshan, Steenkiste, Zhang
Concept formation: Maxion, Mitchell
Concurrency, semantics of: Brookes, Clarke, Platzer, Wing
Concurrent systems: Blelloch, Platzer, Wing
Congestion control: Seshan
Connectionist networks: See Neural nets.
Constraint directed reasoning: Sandholm, Veloso
Constraint logic programming: Pfenning
Constructive mathematics: Harper, Pfenning, Platzer
Control theory: Brunskill, Kolter, Platzer
Cooperating robots: Erdmann, Goldstein, Platzer
Cryptology: M. Blum, Datta, Rudich
Cyber-physical systems: Kolter, Platzer, Wing

Data mining: Brunskill, Carbonell, Faloutsos, Fink, Kolter, Mitchell, Xing
Data structures: Fink, Haeupler, Sleator
Database management systems: Faloutsos, Harchol-Balter, Pavlo
Databases (cf. Information retrieval): Faloutsos, Satyanarayanan
Databases (cf. Storage Systems): Ganger, Gibson
Data types: See Abstraction, see Type Theory.
Dependability: See Reliability.
Design automation: See Computer-aided design.
Diagnosis and diagnostic reasoning: Maxion
Dialog systems: Rudnicky
Digital video: Faloutsos, Hauptmann, Wactlar
Digital hardware design: Hoe
Discovery, scientific: Maxion, Mitchell
Distributed systems: Acar, Andersen, Clarke, Dannenberg, Ganger, Goldstein, Harchol-Balter, Pavlo, Platzer, Satyanarayanan, Seshan, Steenkiste, Wing, Zhang
Dynamical systems: Kolter, Platzer

Education, computer science: Brookes, Pfenning, Shaw
Educational technology: Brunskill, Dannenberg
Electronic commerce: Platzer, Procaccia, Sandholm
Embedded systems: Platzer, Wing
Emergent computation: Goldstein
Energy and smart grid: Kolter
Experimentation: Carbonell, Harchol-Balter, Maxion, Mitchell, Satyanarayanan
Expert systems: See Knowledge-based systems.
Explanation: Mitchell
Eye-tracking: Maxion

Fault tolerance: See Reliability.
File systems: Ganger, Gibson, Satyanarayanan
Fine-grained parallelism: Goldstein
Formal methods: Bryant, Clarke, Garlan, Platzer, Wing
Formal methods in AI: Garlan, Mitchell, Sandholm
Formal verification: Aldrich, Brumley, Platzer, Wing
Foundations of mathematics: Harper, Platzer
Fourier analysis - O’Donnell
Functional programming: Acar, Blelloch, Crary, Harper, Pfenning
Future internet architecture - Seshan

Game-playing, computer: Rudich, Sandholm, Sleator
Game theory: Balcan, A. Blum, Platzer, Procaccia, Sandholm
Geometric reasoning: Erdmann
Graph embeddings: Gupta, O’Donnell
Graph mining: Xing
Graph partitioning: Miller
Graphical models: Kolter, Xing
Graphics: Fatahalian, Hodgins, Pollard

Hand-eye systems: Mason, Mitchell
Heuristic search: Fink, Sandholm
Higher-order logic: See Type theory.
Human-computer interaction: Brunskill, Carbonell, Dannenberg, Fahlman, Hauptmann, Maxion, Reddy, Rudnicky, Shaw
Human-robot social interaction: Hodgins, Pollard, Simmons, Veloso
Human factors: Hauptmann, Maxion, Rudnicky
Humanoid robotics: Hodgins, Pollard
Hybrid systems: Platzer

Image understanding: See Vision.
Image processing: Miller
Information retrieval: Carbonell, Fink, Mitchell, Xing
Information systems: Wactlar
Information technology for international development: Brunskill
Integrated-services networks: Harchol-Balter, Seshan
Intelligent architectures: Carbonell, Fink, Mitchell, Simmons
Intelligent tutoring systems: Brunskill, Dannenberg
Internet: Harchol-Balter, Seshan, Zhang
Internet security: Brumley, Wing
Interprocess communication: Acar, Blelloch

Knowledge acquisition: Carbonell, Fink, Mitchell
Knowledge-based systems: Carbonell, Fahlman, Fink, Mitchell
Knowledge representation: Carbonell, Fahlman, Fink, Mitchell

Lambda calculus: Acar, Blelloch, Harper, Pfenning
Language design: Aldrich, Wing
Language implementation: Acar, Blelloch, Harper
Language modelling: Carbonell
Learning theory: Balcan, A. Blum, Brunskill, Procaccia, Rudich, Veloso, Xing
Library: Wactlar
Linear algebra, computational: Miller, Platzer
<table>
<thead>
<tr>
<th>Field</th>
<th>Authors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lisp</td>
<td>Clarke, Fahlman</td>
</tr>
<tr>
<td>Logic</td>
<td>Acar, Brookes, Crary, Harper, Pfenning, Platzer, Wing</td>
</tr>
<tr>
<td>Logical frameworks</td>
<td>Aldrich, Crary, Harper, Pfenning</td>
</tr>
<tr>
<td>Logics of programs</td>
<td>Brookes, Clarke, Harper, Pfenning</td>
</tr>
<tr>
<td>Machine learning</td>
<td>Balcan, A. Blum, Brunskill, Carbonell, Fahlman, Fink, Kolter, Maxion, Miller, Mitchell, Procaccia, Reddy, Sandholm, Xing</td>
</tr>
<tr>
<td>Machine translation</td>
<td>Carbonell, Xing</td>
</tr>
<tr>
<td>Manipulation</td>
<td>Erdmann, Mason, Mitchell, Pollard</td>
</tr>
<tr>
<td>Manufacturing</td>
<td>Mason</td>
</tr>
<tr>
<td>Markov models</td>
<td>Brunskill, Harchol-Balter, Kolter, Platzer, Procaccia, Simmons, Wing, Xing</td>
</tr>
<tr>
<td>Mathematics of computation</td>
<td>Miller, O'Donnell, Platzer, Rudich</td>
</tr>
<tr>
<td>Measurement and evaluation</td>
<td>Harchol-Balter, Maxion, Satyanarayananan, Seshan, Wing</td>
</tr>
<tr>
<td>Medical image processing</td>
<td>Miller</td>
</tr>
<tr>
<td>Medical robotics/computer-assisted surgery</td>
<td>Kanade</td>
</tr>
<tr>
<td>Memory Systems</td>
<td>Blelloch, Mowry, Mutlu</td>
</tr>
<tr>
<td>Metro Networks</td>
<td>Zhang</td>
</tr>
<tr>
<td>Microarchitecture</td>
<td>Mutlu</td>
</tr>
<tr>
<td>Mobile computing</td>
<td>Satyanarayananan, Seshan, Siewiorek</td>
</tr>
<tr>
<td>Mobile robotics</td>
<td>Goldstein, Mitchell, Simmons</td>
</tr>
<tr>
<td>Model checking</td>
<td>Aldrich, Bryant, Clarke, Platzer, Wing</td>
</tr>
<tr>
<td>Monitoring and error recovery</td>
<td>Simmons</td>
</tr>
<tr>
<td>Motion planning</td>
<td>Erdmann, Mason</td>
</tr>
<tr>
<td>Multiagent learning</td>
<td>Sandholm, Veloso</td>
</tr>
<tr>
<td>Multiagent systems</td>
<td>Platzer, Procaccia, Sandholm, Veloso</td>
</tr>
<tr>
<td>Multicast</td>
<td>Seshan, Zhang</td>
</tr>
<tr>
<td>Multi-core systems</td>
<td>Blelloch, Fatahalian Goldstein, Mowry, Mutlu, Pavlo</td>
</tr>
<tr>
<td>Multimedia</td>
<td>Dannenberg, Hauptmann, Kanade, Wactlar</td>
</tr>
<tr>
<td>Multimodal interfaces</td>
<td>Rudnicky</td>
</tr>
<tr>
<td>Multiprocessors</td>
<td>Acar, Garlan, Goldstein, Harchol-Balter</td>
</tr>
<tr>
<td>Multirobot coordination</td>
<td>Goldstein, Simmons</td>
</tr>
<tr>
<td>Multirobot systems</td>
<td>Goldstein, Veloso</td>
</tr>
<tr>
<td>Music, computer</td>
<td>Dannenberg</td>
</tr>
<tr>
<td>Nanotechnology</td>
<td>Goldstein</td>
</tr>
<tr>
<td>Natural language processing</td>
<td>See Computational Linguistics.</td>
</tr>
<tr>
<td>Network-aware applications</td>
<td>Ganger, Harchol-Balter, Seshan, Steenkiste</td>
</tr>
<tr>
<td>Network Management</td>
<td>Zhang</td>
</tr>
<tr>
<td>Network multiplayer games</td>
<td>Seshan</td>
</tr>
<tr>
<td>Network protocols</td>
<td>Haeupler, Harchol-Balter, Satyanarayananan, Seshan, Steenkiste</td>
</tr>
<tr>
<td>Network security</td>
<td>Brumley, Wing</td>
</tr>
<tr>
<td>Network simulation</td>
<td>Seshan</td>
</tr>
<tr>
<td>Networking</td>
<td>Andersen, Haeupler, Harchol-Balter, Seshan, Steenkiste, Wactlar</td>
</tr>
<tr>
<td>Neural Networks</td>
<td>Blum, Fahlman</td>
</tr>
<tr>
<td>Numerical methods</td>
<td>Miller, Platzer</td>
</tr>
<tr>
<td>Object-oriented programming</td>
<td>Aldrich, Wing</td>
</tr>
<tr>
<td>On-chip networks</td>
<td>Mutlu</td>
</tr>
<tr>
<td>On-line education</td>
<td>Gupta, O'Hallaron</td>
</tr>
<tr>
<td>Operating systems</td>
<td>Andersen, Crary, Ganger, Gibson, Wactlar</td>
</tr>
<tr>
<td>Optimization</td>
<td>Carbonell, Kolter</td>
</tr>
<tr>
<td>Overlay networks</td>
<td>Seshan</td>
</tr>
<tr>
<td>Parallel AI</td>
<td>Fahlman</td>
</tr>
<tr>
<td>Parallel applications</td>
<td>Harchol-Balter</td>
</tr>
<tr>
<td>Parallel architectures</td>
<td>Acar, Blelloch, Goldstein, Harchol-Balter, Miller</td>
</tr>
</tbody>
</table>
Parallel models of computation: Acar, Blelloch, Haeupler, Miller
Parallel object oriented languages: Goldstein
Parallel processing: Acar, Blelloch, Brookes, Bryant, Clarke, Platzer
Parallel programming: Acar, Aldrich, Blelloch
Parallel systems: Acar, Blelloch, Harchol-Balter, Pavlo, Platzer
Parallelizing compilers: Goldstein
Parsing, natural language: Carbonell, Sleator
Pattern recognition: Kolter
Peer-to-Peer systems: Zhang
Perception, auditory: Stern
Performance evaluation: Ganger, Harchol-Balter, Maxion, Satyanarayanan, Seshan
Performance modeling: Harchol-Balter
Pervasive computing: Garlan, Schmerl
Planning: Blum, Brunskill, Carbonell, Erdmann, Fahlman, Fink, Mason, Mitchell, Simmons, Veloso
Polymorphism: Harper, Pfenning
Power management: Satyanarayanan, Siewiorek
Privacy: Balcan, A. Blum, Brumley, Datta, Gupta, Sandholm, Wing
Probabilistically checkable proofs (PCP) - O'Donnell
Probability theory: Haeupler, Harchol-Balter, O'Donnell, Platzer
Problem-solving: Carbonell, Erdmann, Fahlman, Fink, Mitchell
Probabilistic planning: Brunskill, Kolter, Simmons
Program analysis: Aldrich, Brumley, Platzer
Program optimization: Goldstein
Program result checking: M. Blum
Program synthesis: Erdmann, Mason, Platzer
Program transformation: Brookes
Program visualization: Garlan, Schmerl
Programmable matter: Goldstein
Programming environments: Dannenberg, Fahlman, Garlan, Scherml, Shaw
Programming languages: Acar, Aldrich, Blelloch, Brookes, Brumley, Clarke, Crary, Dannenberg, Datta, Fahlman, Harper, Pfenning, Platzer, Shaw, Wing
Programming methodology: Brookes, Clarke, Garlan, Pfenning, Schmerl, Shaw, Wing
Proof theory: Harper, Pfenning, Platzer
Proteomics: Erdmann
Protocols: Harchol-Balter, Rudich, Satyanarayanan, Seshan
Protocol analysis: Aldrich, Harchol-Balter, Seshan
Psychology: Maxion, Rudnicky
QoS management: Mutlu, Seshan, Zhang
Quality of Life Institute: Hauptmann, Wactlar
Queueing theory: Harchol-Balter
Queueing analysis: Harchol-Balter
Random number generation: M. Blum
Randomization: Erdmann, Gupta, Haeupler, O'Donnell, Procaccia, Rudich
Real-time networks: Harchol-Balter, Seshan
Real-time systems: Dannenberg, Harchol-Balter, Platzer, Seshan
Reconfigurable computing: Garlan, Goldstein, Platzer, Schmerl
Recursion theory: M. Blum, Platzer
Reflection: Mitchell
Reinforcement learning: Brunskill, Kolter
Reliability: Ganger, Goldstein, Maxion, Mutlu, Platzer, Satyanarayanan, Shaw, Siewiorek, Wing
Requirements analysis: Maxion
Resource-bounded reasoning: Sandholm
Resource management: Harchol-Balter, Mutlu, Seshan
Robot programming: Erdmann, Goldstein, Mason, Touretzky
Robotics: Erdmann, Goldstein, Hodgins, Kanade, Kolter, Mason, Mitchell, Platzer, Pollard

Scalable systems: Zhang
Scheduling: Acar, Blelloch, Fink, Harchol-Balter
Scientific discovery: Maxion, Mitchell
Scientific computing: Fatahalian, Fink, O'Hallaron, Miller, Platzer
Search: Fink, Sandholm
Security: Datta, Ganger, Maxion, Platzer, Satyanarayanan, Wing
Self-managing systems: Garlan, Ganger, Gibson, Schmerl
Semantic networks: Fahlman
Semantics of programming languages: Acar, Aldrich, Brookes, Clarke, Harper, Pfenning, Platzer, Wing
Sensing, action, prediction: Erdmann
Separation Logic: Aldrich
Signal processing: Dannenberger, Stern
Simulation: Bryant, Harchol-Balter, Hodgins
Software architectures: Aldrich, Garlan, Schmerl, Shaw
Software design methods: Garlan, Schmerl, Shaw
Software engineering: Aldrich, Garlan, Schmerl, Shaw, Wing
Software security: Brumley, Garlan, Wing
Spatial reasoning and representation: Erdmann, Mason
Specialized architectures: Goldstein
Special-purpose systems: Goldstein
Specification: Acar, Aldrich, Brookes, Brumley, Clarke, Garlan, Harper, Pfenning, Platzer, Shaw, Wing
Spectral graph theory: Miller
Speech recognition: Stern
Speech understanding: Hauptmann, Reddy, Rudnicky, Stern
Statistics: Harchol-Balter, Kolter, Platzer, Xing
Stochastic modeling: Brunskill, Haeupler, Harchol-Balter, Kolter, Platzer, Procaccia, Xing
Stochastic processes: Brunskill, Haeupler, Harchol-Balter, Platzer
Storage Systems: Ganger, Gibson
Structural homology: Erdmann
Sufficient correctness: Shaw
Survivable systems: Ganger, Maxion, Wing
Supercomputers: Acar, Blelloch, Goldstein, Harchol-Balter, Reddy
Symbolic computation: Bryant, Platzer
Systems nanotechnology: Goldstein

Tactile sensing: Mason
Task representation: Schmerl
Task-level control: Simmons
Text mining: Carbonell
Text processing: Xing
Theory formation: Mitchell
Transfer principles for complexity theory: L. Blum
Trustworthy computing: Brumley, Wing
Type theory: Acar, Aldrich, Crary, Harper, Pfenning

Ubiquitous computing: Garlan, Schmerl
Usability evaluation: Maxion
User-centered requirements: Shaw
User models: Maxion

Verification: Acar, Aldrich, Brookes, Brumley, Bryant, Clarke, Datta, Garlan, Harper, Platzer, Schmerl, Simmons, Wing
Virtual(ized) reality: Kanade
Vision: Hauptmann, Kanade
Vision, 3D: Kanade
Visual programming: Garlan, Schmerl
VLSI: Brookes, Bryant, Clarke, Goldstein
VLSI-based sensors: Kanade

Web servers: Harchol-Balter
Wireless networks: Seshan, Steenkiste

(c) Courtesy Appointment