Challenges for Theory of Computing

Report of an NSF-Sponsored Workshop on Research in Theoretical Computer Science

April 1999

Anne Condon, University of Wisconsin, Madison
Herbert Edelsbrunner, University of Illinois, Urbana-Champaign
E. Allen Emerson, University of Texas, Austin
Lance Fortnow, University of Chicago
Stuart Haber, Surety Technologies and InterTrust STAR Lab
Richard Karp, University of Washington
Daniel Leivant, Indiana University
Richard Lipton, Princeton University
Nancy Lynch, MIT
Ian Parberry, University of North Texas
Christos Papadimitriou, University of California, Berkeley
Michael Rabin, Harvard University
Arnold Rosenberg, University of Massachusetts
James S. Royer, Syracuse University
John Savage, Brown University
Alan L. Selman, University at Buffalo
Carl Smith, University of Maryland
Eva Tardos, Cornell University
Jeffrey Scott Vitter, Duke University

Executive Summary

This report is the culmination of a two-day workshop, funded by the National Science Foundation (NSF), that took place March 11–12, 1999 in Chicago. Fourteen of the authors and the Program Director of the Theory of Computing, Zeke Zalcstein, attended this workshop. All of the authors participated in extensive discussions by email. The purpose of this effort was to develop and offer a current perspective on research in theoretical computer science.

This is an especially opportune time to carry out this exercise. The President's Information Technology Advisory Committee (PITAC) Report calls for an increase in support of information technology research of roughly a billion dollars over the next five years, and the Administration's proposed Federal budget for FY 2000 demonstrates a commitment to sustained growth in information technology research through its initiative, Information Technology for the Twenty-First Century (IT^2). Since NSF will be the lead agency of IT^2 and NSF is essentially the sole agency funding research in theory of computing, it is certainly appropriate to inquire into the impact of theoretical research on computer science and to attempt to measure the importance of theoretical research for solving strategic long-range problems as called for by the PITAC Report.

We find that theory of computing is a vibrant and exciting discipline of essential importance, about which we will demonstrate the following major points:

- Fundamental discoveries by theoreticians shape computer science dramatically.
- Resources invested in theory of computing research yield significantly multiplied practical benefits.
- Current and future technology depends on research by theoreticians and on collaborative projects that involve both theoreticians and practitioners.

Theoretical research will need to play a critical role in order to meet the challenges set forth in the PITAC report. New fundamental advances will be necessary as will focused research in various applications areas, where collaboration between theoreticians and practitioners inform and enrich one another.

There is an historically demonstrated, enormous potential impact of theory of computing research, in view of which, funding for cutting-edge theory of computing should be increased dramatically. We find that there is a need to increase the number of researchers in theoretical computer science, increase interaction among researchers, and encourage various types of collaboration. To these ends, we conclude this report with a series of specific recommendations to NSF. We recommend a sizable increase in the number and size of awards, creation of a postdoctoral research program, and facilitation of proposals from groups of researchers at different institutions. Also, we recommend several new initiatives:

- Establishment of a national conference center for computer science activities,

- A professional development initiative to prepare theoreticians and practitioners for productive collaborations, and
- A virtual theory center.

In addition, we urge increased representation of theory of computing on the CISE Advisory Committee.

1. Introduction

Theory of computing is a critical component of the discipline of computer science. Theory of computing provides computer science with concepts, models, and formalisms to help reason about these concepts and models. Theory of computing addresses the question of what is and is not feasibly computable and creates the algorithms for the intellectual processes that are being automated. Software and efficient algorithms are the base of today's technology and of technology to come.

The historical role of theory of computing is clear: "Theory pervades the daily practice of computer science and lends legitimacy to the very identity of the field." (Funding a Revolution: Government Support for Computing Research, Computer Science and Telecommunications Board, NRC). Today theory of computing is thriving. In this report we will provide evidence to demonstrate the following major points:

- Fundamental discoveries by theoreticians shape computer science dramatically.
- Resources invested in theory of computing research yield significantly multiplied practical benefits.
- Current and future technology depends on research by theoreticians and on collaborative projects that involve both theoreticians and practitioners.

We will demonstrate that the National Science Foundation's broad-based Theory of Computing program is essential for computer science and the future of information technology. Our concluding section provides recommendations to NSF designed to ensure that we can meet the challenges that we herein describe.

2. From the Past toward the Future

The stored-program computer owes its intellectual origins to Alan Turing, who studied the fundamental nature of computation in the 1930s. The practices of programming computers and designing sequential circuits were significantly advanced by the development of the theory of automata and languages by Chomsky, Rabin, and Scott in the 1950s. Building on these foundations, Knuth and others introduced algorithms and data structures for the efficient parsing of high-level languages, thereby enabling the software revolution of the 1960s. In the 1970s, theoreticians, led by Cook and Karp, exploring the intrinsic complexity of computational problems, identified the large class of NP-complete problems, everyday problems that appear to be so difficult to solve that no foreseeable increase in computing power would enable their solution. Theoreticians interested in studying computational complexity were led to the discovery of computationally hard problems that serve as the underpinnings for modern computer-security systems, notably the RSA public-key cryptosystem. They demonstrated the utility of mathematical logic and automata theory to the verification of complex computer systems; model-checking technology, for example, is now widely used by hardware manufacturers.

Research innovations in the last ten to fifteen years have resulted in new formulations and results that promise a big impact in the future. Now we have fast (polynomial-time) algorithms that provide approximate answers to many NP-complete problems. We use randomized algorithms that provide almost surely fast solutions to hard problems. We employ interactive proof systems, serving to convince one player of the truth of a statement known to a second player, to verify electronic exchanges. These are but a few examples of promising research directions. It is important to realize that several of these innovations resulted from theoreticians' attempts to understand fundamental aspects of computing. They sprang out of the context of theory of computing and would not have occurred in the context of specific applications within technologies that existed at the time that those discoveries were made. Also, it is important to note that theoretical concepts often have taken decades to be assimilated into the mainstream of computing, but such assimilation has had profound practical impact.

Theory of computing "underlies many aspects of the construction, explanation, and understanding of computers. … Many…theoretical concepts from different sources have now become so embedded in computing and communications that they pervade the thinking of all computer scientists." (ibid.)

One strength of theory of computing is that it is not tied to any particular technology, but provides tools for understanding and reasoning about technology in general. For example, the concept of an abstract machine and the simulation of one abstract machine by another, though invented by theoreticians decades ago, can help to understand the modern Web browser. Namely, the Web browser provides users with a common abstract machine across different computing platforms. When a user follows a link to data, a browser invokes the proper interpreter (an abstract machine) to process the data, for example to view an image or run a Java program.

Prospects for the Future

In concert with the PITAC committee, it is safe to predict that future computer systems will be large, involved, and exhibit high emergent complexity. Understanding such systems will be an enormous intellectual challenge that requires the efforts of both theoretical and experimental computer scientists. Theoretical and software models of such systems must be developed and subjected to analysis before making large investments in their implementation. Emphasizing these challenges, George Strawn, of the office of the Assistant Director for CISE, said "we don't understand at a scientific level many of the things that we are building." (Science News, v. 155, Feb 27, 1999)

To summarize, "We need more basic research—the kind of groundbreaking, high-risk/high-return research that will provide the ideas and methods for new disciplinary paradigms a decade or more in the future. We must make wise investments that will bear fruit over the next forty years." (Information Technology Research: Investing in Our Future, PITAC Report, March 1998)

3. Challenges in Computing

"The Committee finds that the Federal agenda for information technology R&D has moved too far in the direction of near-term applications development, at the expense of long-term, high risk, fundamental investigations in the underlying issues confronting the field."
(PITAC Report, March 1998)

Here we explain how theoretical research will play a major role in carrying out the type of fundamental research that is called for in the PITAC report. As we proceed, we will substantiate the following points:

- Fundamental, unpredictable, theoretical discoveries literally change the field of computer science.
- Theory of computing must play a critical role in the development of the technology of the future.
- Addressing high-risk, speculative problems provides the seed corn for future technological innovations.
- Theory of computing will directly change the way science is carried out in the future. The degree to which new technology will be successful in solving the hard problems in science and society hinges on advances in theoretical research in computing.

We substantiate these points in two ways. First, we describe five areas that pose major technological challenges to computer science research and explain how theoretical computer science research (in collaboration and in conjunction with research in many other areas) will help in addressing these challenges. We emphasize that both basic, foundational research, and theoretical research that is focused on individual applications are needed in this endeavor. Second, we describe some types of fundamental theoretical research that will expand the boundaries of our field over time.

Theoretical Computer Science Research Will Benefit Technology

The first area we mention, electronic commerce, is fast becoming ubiquitous in our society. The next two areas focus on meeting the challenges of developing next-generation computer technologies. All three of these areas are listed as research priorities in the PITAC report. A fourth area, also discussed in the PITAC report, concerns innovative computing technologies that may be a reality in the 21st century. The last area, bioinformatics, is one in which theoretical computer science already has an impressive track record in shaping the way biological research is done. In each of these areas, theoreticians are currently successfully engaged.

Lasting contributions to these and other applications areas are likely to be rooted not only in theoretical research that has an applications focus, but also in foundational research whose relevance to a given application may not be apparent at first.

Reliable and Secure Electronic Commerce

The success of electronic commerce, fast becoming a trillion dollar-a-year industry, is crucially dependent on modern cryptography, and thus on the unproven hypothesis of the computational hardness of factoring integers. Yet, as far as we know, there may be a linear time algorithm for factoring! Also there is little evidence that cryptographic algorithms based on block ciphers (such as DES) or one-way hash functions (such as MD5) are secure. In light of the tremendous financial investment in the "hardness" of these problems, serious investment in complexity theory and, in particular, in complexity-theoretic lower bounds is a necessity.

The problems of providing reliable and secure electronic commerce are far from solved. There is a need to develop new cryptographic algorithms that will offer added security, improved efficiency, and solve newly arising technical problems.

Cryptographic algorithms may be the cornerstone of network security, but on top of these algorithms one must add multiple layers of network protocols, a common security infrastructure, and software to actually realize all this. The security of a network depends on the security and soundness of each of these components together with their interactions. Yet how can we tell whether the software that implements these protocols meets their specifications, or even if these specifications guarantee the desired level of security? To deal with these very subtle problems, one needs tools from concurrency theory, semantics, and formal methods. Theoreticians have and will continue to provide formalisms for reasoning about the correctness and security of protocols. (Formal approaches were used to identify the exact source of the Pentium bug.)

Scalable Information Infrastructure

The amount of information on the World-Wide Web is increasing at an astonishing rate, and our nation's dependence on such global information is also increasing. Immediate access to huge amounts of interesting and important information creates amazing opportunities, but also creates challenges that need to be addressed through research. In a rapidly evolving technology, theoretical work must come early, before aspects of the field have frozen in unprincipled, ad hoc states.

With the growing size of the Internet and the amounts of data available, the scalability of our computational methods is becoming increasingly important. Extremely large problems arise when searching the Web, and in managing huge servers that have to service millions of requests per day against exabyte (10^{18} bytes) data stores. Theoreticians have long focused on developing algorithms with low asymptotic complexity and methods that work well on large data sets that reside on distributed computing devices or external memories. With increased problem sizes this focus is becoming even more important. Current methods already incorporate many nontrivial theoretical ideas such as hashing, graph search techniques, and divide-and-conquer.

Theoreticians typically create algorithms that serve as the basic building blocks of large systems. An important way for NSF to engage in long-term research in this area is to invest more in such building blocks. New technologies are bringing many new algorithmic problems into focus. Research on scalable methods and asymptotic complexity provides the best basis for addressing the challenge that these problems present. The following paragraphs elaborate on specific building blocks that theory of computing will contribute to scalable information systems.

Theoreticians have developed methods that provide astonishing speedups for large problem instances. Two of the most successful methods are randomization, such as random sampling, and approximation algorithms. These techniques can provide speedups that are well beyond what was previously conceivable. Research in these areas is very active, producing a continuous stream of dramatically improved algorithms for important problems. Random sampling is essential in order to obtain answers in sublinear time. When a problem is known to be hard, typically the current response is to resort to heuristics with no guarantees. Instead, approximation methods can lead to simple and practical algorithms that guarantee a close to optimal solution. The difference between an approximation algorithm and a heuristic is the proof that the latter has a performance guarantee. Approaches such as these would have been nearly impossible to discover in the absence of a suitable theoretical framework.

Thinking of the Web as a huge graph is frequently important for the development of Web search techniques. Graph-algorithmic methods, studied by theoreticians for decades, are used in current commercial Web search engines. Many new issues arise due to the Web’s enormous size and the fact that the graph is dynamically changing and implemented as a distributed system. Another important aspect is the information latent in the Web's hyperlink structure. These issues lead to hard graph-algorithmic problems, which when solved will be valuable for future Web search engines.

The fields of data mining and discovery science seek ways to find interesting properties in information, with applications to commerce, user interfaces, and more. The challenge is to query and process huge amounts of information quickly and accurately. Researchers are exploring new methods that are rooted in graph algorithms, information theory, and learning theory to advance these fields.

Many of the known asymptotically faster methods use linear space. Implementation of these linear-space algorithms utilize external data storage. Researchers are trying to discover whether we can make effective use of small internal memory instead. Theoreticians are addressing the problem on multiple fronts. Some approaches seek to compress data, often by provably good sampling techniques, so that the online processing can be done in the much faster internal memory. Another important line of research is to exploit locality in data layout in order to optimize access to the data in external storage. Theory is needed to identify the performance limits of these approaches and to determine when each is applicable.

The rapid growth of the Internet, the demands on network bandwidth, and the stringent service requirements of real-time applications, such as on-demand video, highlight the need for improved data compression and caching. Theoreticians have made a variety of contributions to data compression. Often the contributions require expertise in more than one theory domain, further emphasizing the needed role of the theoretician. For example, prefetching and caching mechanisms based upon universal data compression techniques have been shown to capture locality present in data accesses and thus optimize bandwidth.

Safe and Verifiable Software

A broad spectrum of theoretical areas support the specification, verification, and synthesis of both hardware and software systems, e.g., the areas of algebraic specification, logics of programs, model checking, process algebras, program semantics, and type theory. Each of the aforementioned areas has been instrumental in the development of strong, practical tools in current use. We briefly consider the cases of model checking and logics of programs.

Model checking provides an automatic verification method for hardware and software. It has been very successful for large hardware designs because the regularity of hardware supports the use of compact data structures (e.g., binary decision diagrams) that can succinctly represent very large state spaces. To achieve similar success for software systems, more flexible representational techniques must be found. In regard to programming logics: modal and temporal logics provide the basis of model checking as specification languages, process calculi are commonly used to design and analyze network and cryptographic protocols, and the recent invention of proof-carrying code is an important new tool for network security.

While the tools and insights provided by model checking, logics of programs, and the like, are quite useful, each of these areas addresses only a very narrow set of problems. The practice of software development desperately needs a strong, sound mathematical foundation that encompasses the entire development process. The challenge for theory is to broaden and integrate these areas to deliver this foundation.

Speculative Computing Technologies

Theoretical computer scientists, along with physicists, chemists, molecular biologists, and other scientists, are exploring computing technologies at the molecular level, or technologies that are based on quantum mechanical principles. These emerging technologies, by their very nature, are hypothetical, and thus to understand their potential, theoretical models and simulations are essential. Results of theoretical computer scientists have given a major boost to the work in both biomolecular and quantum computing.

Related to biomolecular computing research is the area of molecular nanostructures, which seeks to reliably self-assemble large structures from a small number of molecular building blocks. Theoreticians have shown that the self-assembly process is controllable by appropriate design (“programming”) of the initial building blocks, thereby significantly expanding the potential for creation of interesting nanostructures.

In addition to the preceding speculative technological advances, there are numerous innovative advances that already exist, but which are not being exploited to capacity due to insufficient knowledge about how to use them. A prime example is the use of clusters of workstations as a platform for parallel computing. Whereas the technology for achieving cooperation among workstations in a cluster is well developed, the environment that they present is sufficiently different from that of a multiprocessor that we do not yet know how to schedule parallel computations effectively on clusters. A few recent theoretical studies are just beginning to develop the understanding and the algorithms necessary to remedy this situation.

Bioinformatics

Researchers in theory of computing have long been interested in modeling biological phenomena, for example by cellular automata and Lindenmeyer systems.
Much current research in molecular biology is devoted to understanding the structure and function of biomolecules, such as the human genome. Combinatorial algorithms play a major role in the computational tasks of deducing accurate DNA sequence data from short fragments that contain errors, of mapping out important “landmarks” in long sequences, and also of predicting protein structure from the underlying amino acid sequences. A second important role of theory of computing is to provide good models and methods for reconstructing the process of evolution from such sequence data, such as methods for constructing phylogenetic trees.

In addition, modeling of biomolecules raises new questions in geometry and physics. G. D. Rose wrote that “what role a protein takes in the grand biological opera depends on exactly one thing: its shape. For a protein molecule, function follows form.” (No assembly required, The Sciences, 36 (1996), 26–31). New models for biomolecules, such as alpha shapes, are being developed by theoretical computer scientists that can aid in the design of drugs, in the classification of molecular components, and in the simulation of docking and folding processes.

The proliferation of biological data, and the need for its systematic and flexible storage, retrieval and manipulation, is creating opportunities in the database field. Genomic databases are heterogeneous, distributed, and semistructured, or with a schema that is in flux, thus offering novel challenges to database design, including its more fundamental aspects. The study of gene expression, protein structure and cell differentiation, as well as the emergence of microarrays, are introducing potentially unlimited data (while the whole human genome is less than one gigabyte, a limited size that initially misled many database researchers to underestimate the field). There are new challenges for indexing of sequences and three-dimensional structures, for lab workflow storage and process modeling, and for processing queries that involve specialized approximate pattern matching and complex geometric relations such as docking. These problems would benefit much from the theory community's extensive experience with string matching, scheduling, and computational geometry. As it has happened in the past with several other challenges to database research, early principled theoretical investigation of these problems is already an important ingredient of the field's research agenda.

Continued Theoretical Research is Critical

Naturally, the areas above can only illustrate the need for theoretical research. The sole intent of the discussion here is to demonstrate, by a few examples, strategic areas in which theory of computing has been successful and in which continued theoretical research is critical. This list is not intended to be exhaustive. Also, we wish to stress that our list provides a view of the future from where we stand now. In truth, we do not know what the exciting challenges fifty years from now will be. Fifty years ago there were no more than a few computer experts and they could not have predicted the massive expanse of computing and communications in our society today. Sixty years ago DNA had not even been identified as the carrier of genetic information; the marshaling of science and technology to address the new questions raised by biomolecular data would not have been predictable.

Quite apart from technically-grounded evidence such as we present here, there are many other indicators of the importance, relevance, and health of theory of computing today. For example, consider the number of conferences centered on theoretical work, ranging from some that focus heavily on particular application domains to others that focus on foundational studies. Fundamental and innovative work in theoretical computer science has received favorable attention in broad-based scientific publications and in the national media over the past decade (for example, interactive proof systems, zero-knowledge, probabilistically checkable proofs, descriptive complexity); this has helped to make computer science research visible to those outside of our field. Researchers in theory of computing collaborate with computer scientists and engineers in many areas, with mathematicians, and with researchers in finance and the physical and biological sciences. Several leading theoreticians at major research institutions have founded their own companies in cryptography, security, web applications, and other areas. Many others are being lured from academia to be leading scientists at such companies, at large well-established corporations, and in arenas such as finance and biotechnology.

Foundational Research Transforms Computer Science

Recent successes strongly indicate that we can expect a continued flow of important results from theoretical work for the foreseeable future—results that can transform the course of computer science research and, ultimately, the way technology is used. In many cases, these results emerge in unpredictable ways from apparently unrelated investigations of fundamental problems.

While many of the deep theoretical problems that are attracting the best minds in our field are rooted in, or inspired by, challenges such as those listed above, it is often difficult for researchers to properly tackle such problems in the context of an application-driven research environment. One reason for this is the long time period needed for work on fundamental problems to come to full fruition. In addition, solutions to such problems draw on diverse mathematical and computational methods, and so the interaction of a broad community of theoretical researchers is essential.

Moreover, the best theoretical results typically influence the course of research in several application areas, and so it is extremely useful to maintain an identifiable corpus of theoretical knowledge that is accessible to the Computer Science community and to the scientific research community at large.

A third reason for the importance of foundational research which is not targeted at a specific application is that, because of its focus on high-risk and speculative problems, and because of the often surprising or unpredictable consequences that stem from theoretical research, such research provides the seed corn for innovations of the future. This is underscored by the introductory quote of this section.

For all of these reasons, unfettered research in foundational theoretical areas is vital, in order to provide better understanding of the capabilities and limitations of computers, and to ensure future innovations in science and technology.

Models of Computation and Information; Complexity Theory

Fundamental questions remain on the relationships among models of computation, information representation and manipulation, and on good ways to express algorithms. The P versus NP question is perhaps the most famous of these, the refinement of which has led to many other important questions in complexity theory. Progress on complexity-theoretic problems, even when of the “negative” type (such as providing evidence for the intractability of certain problems) can completely change computer scientists' approaches to practical problems in surprising ways. For one thing, researchers no longer waste time seeking efficient solutions to intractable problems. Instead, they invent and learn techniques for coping with intractability. As we mentioned earlier, the computational hardness of certain problems has been exploited for cryptography. Currently, computational hardness of certain problems is being harnessed to obtain efficient deterministic (error-free) algorithms for problems where randomness (and thus error-prone) algorithms previously seemed necessary.

The theory of computing community continues to produce wonderful fundamental ideas, and, over time, these influence practice in important ways. The interplay among concepts such as pseudorandom number generation, interactive proofs, and secure cryptographic protocols is beautiful and deep, and has significant potential to the practice of cryptography. The introduction of interactive proof systems and probabilistically checkable proofs has broadened and enriched our understanding of the concept of proof. Probabilistically checkable proofs have turned out to be a fundamental tool for studying the limits of polynomial-time approximation algorithms.

Foundational questions will require a concerted effort in the areas of classical Turing machine-like models and variants (such as randomized or quantum models), models of learning, formal methods and program inference, models of nonsymbolic reasoning, logical characterization of complexity classes, lower bounds, models of on-line computation, models for communication of information, models for inferring information from incomplete data, models for data storage and retrieval in a multimedia context, and parallel and distributed models. Study of connections between models and results in more than one of these areas can be particularly fruitful. For example, on-line algorithms, common in key frameworks such as operating systems, financial control, and real-time systems, have generated fundamental concepts that are important for distributed systems design, in particular where information flow is more complex than traditional input/output.

Design and Analysis of Algorithms

Foundational work on algorithms design has the goal of breaking long-standing barriers in performance. There are many situations where complicated algorithms, such as Strassen and Schönhage's multiplication algorithm, were deemed inferior for years, but with steady increases in problem sizes, such algorithms now are preferred on appropriate high-performance computing platforms.

In other cases, (such as linear programming) initial breakthroughs in reducing asymptotic running time, while not practical in and of themselves, serve to stimulate new research that eventually leads to practical algorithms. What tends to happen is that once a barrier, such as the existence of a polynomial-time algorithm for a problem, is broken, there is strong justification and real motivation for researchers to revisit a problem that previously appeared impenetrable. Very often, painstaking refinement of the seminal breakthrough technique leads to a truly practical algorithm. Ultimately, this type of research has long-lasting impact on the practice of computing. Tantalizing open questions remain, such as whether there is a polynomial time algorithm for graph isomorphism, or whether one can efficiently learn Boolean formulas that are in disjunctive normal form from random examples.

New Theories of Algorithms and Heuristics

While theoretical work on models of computation and methods for analyzing algorithms has had enormous payoffs, we are not done. In many situations, simple algorithms do well. Take for example the Simplex algorithm for linear programming, or the success of simulated annealing on certain supposedly “intractable” problems. We don't understand why! It is apparent that worst-case analysis does not provide useful insights on the performance of many algorithms on real data. Our methods for measuring the performance of algorithms and heuristics and our models of computation need to be further developed and refined. Theoreticians are investing increasingly in careful experimental work leading to identification of important new questions in the algorithms area. Developing means for predicting the performance of algorithms and heuristics on real data and on real computers is a grand challenge in algorithms.

On numerous occasions, theory of computing research has provided the insights that explain why popular, important heuristic algorithms work. Significantly, these insights have suggested major improvements to the heuristics. One example of this scenario was the study by Turner in the 1980s that explained why the decades-old Cuthill-McKee heuristic for minimizing the bandwidth of sparse linear systems works well in practice; this understanding allowed Turner to devise an improved heuristic. A second example, also from the 1980s, explained why the Kernighan-Lin graph bisection heuristic, which is important in the field of circuit layout, works well in practice.

4. Conclusions

This report demonstrated the following major points:

- Fundamental discoveries by theoreticians shape computer science dramatically.
- Resources invested in theory of computing research yield significantly multiplied practical benefits.
- Current and future technology depends on research by theoreticians and on collaborative projects that involve both theoreticians and practitioners.

We learned of the enormous potential impact of theory of computing research. For purposes of illustrating this impact, we examined five major technological challenges to computer science research, and explained the need for theoretical research in order to address these challenges. As one lists the areas of theory of computing research that relate to just these five challenges, the result is overwhelming and daunting: complexity theory and complexity-theoretic lower bounds, design of efficient algorithms, concurrency theory, semantics, formal methods, asymptotic complexity, approximation algorithms, graph algorithms, learning theory, algebraic specification, logics of programs, modal and temporal logics, model checking, process algebras, type theory, combinatorial algorithms, string matching, scheduling, and computational geometry. (We list these areas in the order in which they occurred in the previous section.) Let us add that several of these challenges will benefit from advances in descriptive complexity and finite model theory. Moreover, we reiterate that these five challenge areas were intended only to illustrate the need for theoretical research, and were not intended to be exhaustive.

We learned that unfettered research in foundational theoretical areas, research that is not targeted at specific applications, is vital. Historic evidence demonstrates that such research transforms computer science dramatically.

Recommendations

The current funding model and level of funding (single-PI grants, with funds for at most a single graduate student and limited travel money) is dangerously insufficient, to the extent that important research is not being carried out for lack of resources. In view of the potential impact of theory of computing research that this report has demonstrated, funding for cutting-edge theory of computing should be increased dramatically. There is a need to increase the number of researchers in theory of computing, enhance interactions between researchers, and encourage various types of collaborations.

Our recommendations fall into two broad categories. The first addresses ways in which traditional NSF funding for theory of computing needs to be strengthened. The second lists innovative new initiatives for enhancing research capabilities.

The traditional grant program should be improved by the following means:

- Increase the number of awards from the current level of approximately 30 per year to at least 60 per year.
- Increase the duration of awards from three years to five years.
- Increase the size of awards to allow for more than one graduate student, adequate travel money to foster collaboration, and sufficient equipment money to ensure a modest but up-to-date computing environment.
- Create a postdoctoral research funding program. Grants should be awarded to established researchers rather than individual postdoctoral students in order to increase the diversity of available research experiences.
- Encourage and facilitate proposals from groups of researchers at different institutions.

In addition to strengthening traditional grants support, we urge the NSF to undertake the following innovative initiatives.

Conference Center

We propose the establishment of a research conference center in a rural setting along the lines of Oberwolfach and Dagstuhl in Germany. We recommend establishing the conference center as an independent entity, and suggest that such a center would be of benefit if created to include all of the disciplines within CCR. This center would act as a crucible, bringing together researchers from around the country to participate in workshops and collaborations in an informal and congenial setting. We envision both specialized and interdisciplinary workshops to enable interactions.

Professional Development

We propose a new faculty development initiative to enhance the skills and knowledge of theoreticians and enable them to work with researchers from other fields on important problems. Under this initiative, a program will be established to provide funds for researchers to spend one or two semesters studying and learning in some other discipline preparatory to starting collaborative research activities.

Virtual Theory Center

We propose the establishment of a Virtual Theory Center along the lines of the PITAC recommendation of expeditions into the 21st century. We envision using web-based technology to foster interactions between geographically diverse researchers without the need for expensive and time-consuming travel, including activities such as the following:

- Real-time storage and retrieval facilities for research colloquia;
- Web-based video conferencing technology for interactions among members of a research team.

The Virtual Theory Center will enable researchers at under-represented, under-funded, and geographically isolated institutions to participate in research activities. Establishment of this center would require:

- A centralized computational server, with coordination, management, and technical staff, and
- Infrastructure to enable researchers to use the facilities of the Virtual Theory Center, including video cameras, electronic whiteboards, and high-speed network links.

The Virtual Theory Center can enable continuation of collaborations created as an outcome of the Professional Development Initiative, and the Conference Center.

CISE Advisory Board

We strongly recommend increased representation of theory of computing on the CISE Advisory Committee. As this report has demonstrated, theory of computing is essential to all CISE activities.

Acknowledgments

The authors appreciate the interest and encouragement of Zeke Zalcstein, Theory of Computing Program Director, Mike Evangelist, C-CR Division Director, and Ruzena Bajcsy, CISE Assistant Director. We thank Ed Clarke for his useful pointers and comments, and Juris Hartmanis, past CISE Assistant Director, for his many insights.