Emerging Opportunities for Theoretical Computer Science

Alfred V. Aho
Columbia University

David S. Johnson
AT & T Research

Richard M. Karp (Chair)
University of Washington

S. Rao Kosaraju
Johns Hopkins University

Catherine C. McGeoch
Amherst College

Christos H. Papadimitriou
University of California at Berkeley

Pavel Pevzner
University of Southern California

October 15, 1996

Abstract

The principles underlying this report can be summarized as follows:

1. A strong theoretical foundation is vital to computer science.
2. Theory can be enriched by practice.
3. Practice can be enriched by theory.
4. If we consider (2) and (3), the value, impact, and funding of theory will be enhanced.

In order to achieve a greater synergy between theory and application, and to sustain and expand on the remarkable successes of Theory of Computing (TOC), we consider it essential to increase the impact of theory on key application areas. This requires additional financial resources in support of theory, and closer interaction between theoreticians and researchers in other areas of computer science and in other disciplines.

The report does not make a detailed assessment of the overall state of theoretical computer science or fully chronicle the achievements of this field. Instead, it has the specific objective of recommending ways to harness these remarkable achievements for the solution of challenging problems emerging from new developments such as the information superhighway.

Section 1 describes the events leading up to this report and delineates the report's objectives. Section 2 establishes the context for the report. It traces the history of TOC, describes the impact that TOC has achieved in the areas of core theory and fundamental algorithms, points out the differences between these areas and application-oriented theory, and calls for an intensified effort to bring the methods of TOC to bear on applications. It then goes on to define the four main categories into which our recommendations fall: building bridges between theory and applications, algorithm engineering, communication, and education. Section 3 discusses some specific opportunities for stimulating interactions between TOC and applied areas. Section 4 proposes an applied research initiative, *Information Access in a Globally Distributed Environment*, which identifies an exciting current technological area that we believe presents challenging opportunities for excellent theoretical work. Section 5 proposes a second applied research initiative, *The Algorithmic Stockroom*, that would exploit and extend the body of theoretical knowledge in the field of algorithms. Section 6 proposes a broadening in graduate education with two purposes in mind: to better prepare theoreticians to interact creatively with practitioners, and to provide future practitioners with the background they will need to benefit from this exchange.

1 History and Scope of This Report

This report grew out of an NSF-sponsored workshop held on June 2, 1995 with the purpose of assessing the current goals and directions of the Theory of Computing (TOC) community and suggesting actions and initiatives to enhance the community's role in improving computing technology and in technology transfer.

The workshop was preceded by an open meeting on the same topic at the 1995 ACM Symposium on Theory of Computing(STOC), at which a number of position papers were presented. A preliminary version of this report was presented orally and discussed in an open meeting at the October, 1995 IEEE Symposium on Foundations of Computer Science. A version of the report was submitted to NSF and posted to the Worldwide Web on Feb. 15, 1996. A revised version was distributed at the May, 1996 STOC Conference. At that conference a further open meeting was organized to explore the overall goals of TOC. There was strong agreement that uncompromising insistence on the highest quality of pure theoretical research has been instrumental in the remarkable breakthroughs in the field, and that the same tradition must continue. There was a consensus that TOC has exerted a vast influence on the fields of software and system design. Several speakers correctly pointed out that there will always exist a considerable time lag between the development of a new theoretical idea and its eventual transfer into the applications domain.

There was some disagreement on the future directions TOC should pursue. Several highly respected researchers felt that, even though applications-oriented research is of great value, TOC will have the greatest impact if it concentrates exclusively on the fundamental mathematical principles. We also strongly endorse the view that our primary task should be the pursuit of theoretical developments. However, we believe that there are many exciting areas of opportunity that involve building bridges between the Theory of Computing and the rest of computer science and other disciplines.

In this report, we do not undertake to make a detailed assessment of the overall state of theoretical computer science or to fully chronicle the achievements of this field. These essential tasks will require a continual effort involving many members of the TOC community. The Loui report [3] is a good first step in this direction.

For the sake of emphasis this report often states our recommendations in prescriptive language. However, our purpose is not to coerce, but to communicate our point of view and stimulate members of the Theory of Computing community to make thoughtful choices about how they can best contribute to the progress of our field.

2 Introduction

Theory and theoreticians have played a major and brilliant rôle in the history of Computer Science. In fact, two of the pioneers of Theoretical Computer Science are also among the founders of Computer Science (CS): Alan M. Turing and John von Neumann. During the two decades after the death of these giants (roughly 1955 to 1975) theoreticians developed a set of basic concepts and methodologies that transcend application domains and set our science apart from other applied and natural sciences. These fundamental contributions include *automata- and language-theoretic models*, *data structures and algorithms*, *methodologies for evaluating algorithm performance*, *the theory of NP-completeness*, *logics of programs and correctness proofs*, and *methods of public-key cryptography.* During this period theoreticians also contributed crucially to some of the most celebrated engineering triumphs of CS, especially in the areas of compilers, databases, and multitasking operating systems.

By 1975 the field had discovered a realm of deep problems (especially those relating to the P vs. NP question) of the kind that take decades to solve and bring to a young science the aura of depth and respectability, and a sophisticated and powerful methodology was emerging, proudly borrowing from logic and combinatorics. Also, commonality of methodology and objective brought social cohesion: By 1975 theory conferences had already been running for about ten years (FOCS and STOC in the U.S., ICALP in Europe).

During the next twenty years theoretical CS attracted to its ranks some of the brightest young scientists, and celebrated some major achievements in fundamental research. Many insights into the nature of efficient computation were gained. The pervasive role of randomness in computation came to be understood, and a clear understanding of the meaning and uses of pseudorandomness was achieved. The concepts of interactive proof and probabilistically checkable proof were set forth and found to have deep and unexpected consequences. During this period the field branched out into a number of promising new areas. These include *computational learning theory*; *probabilistic, approximate and on-line algorithms*; *computational geometry*; *models and algorithms for distributed and asynchronous computation*; *secure cryptographic protocols*; *logics of knowledge*; and *quantum computing*.

Theoretical computer science provides a conceptual framework that is indispensable for the practice of computer science. Basic concepts of algorithmic efficiency and algorithmic correctness, NP-completeness, data structures, public key cryptography and parsing and compilation have become second nature for all computer scientists. Fundamental ideas of more recent vintage concerning such topics as probabilistic algorithms, interactive computation, parallel and distributed algorithms, secure protocols, on-line computation, logics of knowledge, temporal logic and relational databases are gradually finding their way into the general consciousness of the computer science community, and many specific algorithms related to geometric, algebraic and number-theoretic computing, linear and nonlinear programming, network flows, combinatorial optimization and fast text searching are being put to use in specific applications. The Theory of Computing also provides orientation for other scientific disciplines, ranging from neurophysiology to economics, that study complex information processing systems and rational decision making.

These remarkable results are achieved with very low levels of research funding. Low levels of available funding resulted in many unexplored opportunities for important scientific work that bridges the gap between theoretical computer science and its areas of application, both within computer science and within other disciplines. The development of these bridging areas can enrich theoretical computer science and enhance its impact on society. In addition, work at the interface will strengthen the connection between TOC and the rest of computer science. Such an expansion of research thrust must be based on the availability of additional research funding. Our report aims to identify some of these promising bridging areas and demonstrate their promise.

In undertaking this task we found it useful to identify three distinct (although heavily interacting and overlapping) types of research activity within TOC.

1. The field of **core theory** studies computation as an abstract phenomenon, and explores the mathematical techniques that are (presently or potentially) useful in computer science. In recent years research in core theory, supported mainly by the NSF Theory of Computing program, has established solid foundations for cryptography, interactive computation, computational learning, parallel and distributed computation, randomized computation, on-line computation and reasoning about knowledge. These notable and largely unexpected developments contribute greatly to the appeal and distinctive flavor of theoretical computer science. Core theory has achieved a remarkable unity, in which seemingly unrelated topics are found to have surprising connections and to be amenable to investigation by common mathematical techniques. One example is the use of probabilistically checkable proofs to prove that the approximate solution of certain optimization problems is NP-hard. A second is the variety of interrelationships among cryptography, pseudorandom number generation, complexity theory and computational learning theory. A third is the broad applicability of recurrent mathematical themes such as rapidly mixing Markov chains, pairwise independent random variables, expander graphs, and discrete Fourier analysis.

2. The field of **fundamental algorithms** uses sophisticated mathematical techniques to develop solutions to problems that transcend application domains. This area includes such central topics as combinatorial algorithms, approximation algorithms, on-line algorithms, numerical algorithms, algorithms for computational geometry, and data structures for search, graph operations, and point set manipulation.

3. Finally, **application-oriented theory** refers to theoretical work performed in the context of a specific application. This work usually requires close collaboration between theoreticians and applied researchers, and entails a commitment on the part of the theoreticians to have an impact on the applied problem. This area of research often exploits the concepts of core theory and the developments in fundamental algorithms, but more detailed modeling and analysis, based on the specific requirements of an application or the current state of computer technology, is usually required. The Human

Genome Project is an example of an application area that has stimulated many recent algorithmic developments. For another example, there is a large community of theoreticians who develop models and algorithms tuned for specific kinds of parallel machines, computer networks, operating systems or database systems.

We view computer science, like all successful scientific and engineering disciplines, as a continuum ranging from the theoretical to the applied, in which theoreticians can derive stimulation from exposure to applied problems, and, by applying their abilities at modeling, abstraction and mathematical analysis, can contribute to the revolutionary developments that are surely coming in the field of computing. Most of our recommendations follow from this view of computer science.

Although our focus in this report is on outreach to related fields, we consider it essential to maintain a strong activity in core theory and fundamental algorithms, since these are the areas that provide the most fundamental and long-lasting advances in TOC, give the subject its unique character, and produce the most dramatic new developments. We strongly believe that enhanced activity in application-oriented theory will be complementary to activity in the core areas. Such an outreach creates exciting opportunities for technology transfer and stimulates additional funding for applied theory from funding agencies and directorates concerned with specific application areas.

In the following paragraphs we state some broad recommendations for the future development of application-oriented theory. These are followed in later sections by more specific recommendations of research directions at the applied end of the TOC spectrum.

Our broad recommendations fall into four main categories: building bridges between theory and applications, algorithm engineering, communication, and education.

Building bridges between theory and applications. There are a number of striking recent cases in which the theory of computation has exerted a direct influence on applied fields:

1. Algebraic algorithms for robot motion planning, originally studied from a complexity-theoretic point of view, have influenced applied developments in robotics.

2. Close ties have been established between theoreticians working in computational learning theory and the more applied machine learning community within artificial intelligence.

3. Theoretical computer scientists in industrial research laboratories are developing and implementing protocols for secure communication over computer networks, based on fundamental complexity-theoretic results in cryptography.

4. A new generation of efficient large-scale linear programming codes has been developed, building on theoretical analyses of interior-point methods for linear programming.

5. Theoretical ideas from coding theory have improved the error resilience of the transmission of multimedia data over packet networks.

Many further examples could be drawn from such fields as geometric modeling, relational databases, network optimization, and computational biology.

In each of these cases, the exploitation of theoretical results was far from trivial, and required a deep understanding of both the underlying theory and the application domain. Such applied studies require theoreticians to cope with the messiness of the real world, often at the expense of elegance and universality. Nevertheless, in order to maximize its impact on the rest of computer science and on other fields, the TOC community will have to encourage the strenuous intellectual effort that is required to carry a theoretical idea through to the point where it actually begins to have a practical impact. For example, it is particularly important that the TOC community take a stronger role in the emerging information technology revolution. The information superhighway is having a profound impact on society and it provides challenging research avenues to computer scientists. In addition, the theoreticians we train in such applied areas will have the tools to discover their own exciting new opportunities for interesting and productive work.

Algorithm engineering. Within theoretical computer science algorithms are usually studied within highly simplified models of computation and evaluated by metrics such as their asymptotic worst-case running time or their competitive ratio. These metrics can be indicative of how algorithms are likely to perform in practice, but they are not sufficiently accurate to predict actual performance. The situation can be improved by using models that take into account more details of system architecture and factors such as data movement and interprocessor communication, but even then considerable experimentation and fine-tuning is typically required to get the most out of a theoretical idea. Efforts must be made to ensure that promising algorithms discovered by the theory community are implemented, tested and refined to the point where they can be usefully applied in practice. This can only happen if the TOC community comes to recognize algorithm engineering — the experimental testing and tuning of algorithms — as integral to its mission.

Communication. We must *explain* to the rest of the CS community, to the broader scientific community, and to the society as a whole our past accomplishments, our present endeavors, and our potential for outstanding future contributions. There is a need for a continuing flow of expository articles making developments in TOC accessible to a broad audience.

We can reach out to nonspecialists by publishing broadly accessible expository articles in publications of general interest. These should include realistic assessments of the applicability of our results, making sure that real opportunities for applications are pointed out.

The unique, universal, and ubiquitous themes of computer science can only be developed by theoreticians who maintain strong communication ties with one another, and for this reason theoreticians often band together in "theory groups" within their academic departments or research organizations. It is important that such organizational structures should not insulate theoreticians from other computer scientists. Theory researchers who commit themselves to a particular applied topic should be willing to step outside their theory groups for a period of time or even establish a permanent dual affiliation.

Education. Graduate students broadly trained in theoretical computer science will be well prepared for intellectual growth and will have excellent opportunities in the job market. For the health of the applied branches of computer science, it is equally important that applied computer scientists receive a good exposure to the methods and results of the Theory of Computing.

The remainder of the report fleshes out the picture developed in this Introduction and suggests a series of actions consistent with the broad recommendations we have given.

Section 3 proposes some opportunities for stimulating interactions between TOC and applied areas. Section 4 proposes an applied research initiative, *Information Access in a Globally Distributed Environment*, which identifies an exciting current technological area that we believe presents challenging opportunities for excellent theoretical work. Section 5 proposes a second applied research initiative, *The Algorithmic Stockroom*, that would exploit and extend the body of theoretical knowledge in the field of algorithms. Section 6 proposes a strengthening of graduate education with two purposes in mind: to better prepare theoreticians to interact creatively with practitioners, and to provide future practitioners with the background they will need to benefit from this exchange.

3 Bridges to Applications: Some Promising Directions

It is essential for the TOC community to maintain a strong presence in the core of theoretical computer science. In addition, it is important to bring the work on fundamental algorithms into closer contact with computational practice, and to develop a stronger presence in selected application-oriented areas.

In this section we describe some areas where there are good opportunities for theoreticians to have an impact on applications. We have selected areas where there is a good possibility of attracting funding from sources which have not traditionally supported theoretical computer science. Our recommendations should be read in conjunction with those of Raghavan [4], who has provided an excellent overview of the opportunities for application-oriented theory. He stresses the current importance in the marketplace of the business sector and the home/personal sector, and points out the massive amount of data that is now available on-line, as well as the opportunities provided by the Web to make material accessible to a wide audience. He then lists the following applied areas that are ripe for impact from theoreticians: massively distributed computing, mobile computing, security, data mining, data visualization, user models, systems management, design automation and verification, geometric computing, scientific computation, coding and compression, information retrieval, combinatorial optimization, decision support and financial applications. Our recommendations are intended to complement Raghavan's suggestions.

Application: Theory of Software Construction. As each generation of computer technology brings radical reductions in the cost of hardware, our inability to achieve a significant reduction in the cost of software development becomes a greater cause for concern. The demand for new software systems far outstrips our ability to construct them. There is a large gap between theory and practice in the area of software construction. At the interface between computer science and mathematical logic there have been impressive advances in our theoretical understanding of programming language semantics, specification languages and formal theories of program correctness, but these developments have not led to comparable advances in the pragmatic task of producing reliable software. To encourage researchers to focus on this task we recommend continuing support of research on the scientific foundations of software engineering and the verification of software systems, but with a strong preference given to investigations that are directly oriented toward the software development process.

Application: Computational Biology. The field of biology has been undergoing a conceptual revolution, in which cells and organisms are viewed as information processing systems. Biologists have learned that digital information in the form of DNA governs the chemical processes of the cell and constitutes the genetic endowment that is passed from parent to child. A vast effort is underway to sequence the human genome, elucidate the information it contains, and identify the abnormalities within it that account for genetic diseases. This quest has led to the development of bioinformatics as an important new professional specialty, and has introduced a need for new combinatorial algorithms and database techniques to support the acquisition and interpretation of genetic information. There are exciting opportunities for interaction between TOC researchers and biologists. A growing number of theoretical computer scientists are already rising to this challenge. Additional initiatives to foster close interaction between algorithm researchers and biologists will pay great dividends to society. We are hopeful that this effort will draw enthusiastic support from the biology side as well as from the traditional funding sources for theoretical computer science.

Future Application Areas. The choice of application areas for future major funding efforts should address the roadblocks that limit the effectiveness of computing and telecommunications in our society. Proposals for theoretical work motivated by the ubiquity of computers and communications, parallel and distributed systems, and large knowledge repositories should be warmly encouraged. Encouragement should also be given to theoretical research that supports the efficient construction of portable and reusable software. In the following two sections we propose two specific initiatives intended to focus theoretical research on these objectives. The first of these is entitled "Information Access in a Globally Distributed Environment" and the second is called "The Algorithmic Stockroom." The first initiative represents a timely opportunity for theory because the necessary abstractions and theoretical models are only beginning to be understood. The second initiative is an opportunity for extending the reach of theory to the development of high-quality reusable software.

4 Information Access in a Globally Distributed Environment

From the inception of computer science as a discipline, theoretical work motivated by the random-access machine model of computation has had significant impact on the practice of computing; and the practice of computing has helped nourish theoretical development and use of new models of computation. There is growing evidence that the development of a ubiquitous global information infrastructure offers abundant opportunities for similar contributions from theory to practice, as well as strong motivation for the development of new theoretical models and the identification of new problems. The return on investments in theoretical research in this area should be substantial because much of the new infrastructure is still undeveloped.

Timely, accurate information is vital for peoples' health, education, and economic well-being. As computers and communications become ubiquitous, it should become possible for people wherever they are to find the information they need, in the media they desire, and in a form they can understand. However, we are far from achieving this goal:

- New models and methods are needed for the organization, storage, retrieval, processing, and presentation of heterogeneous multimedia information. Relational database theory had a profound impact on the design of today's database management systems but at present we do not have a similar foundation for the effective design of large, distributed, multimedia knowledge repositories. General models and efficient algorithms for multimedia query processing are at the moment rudimentary and unsatisfactory.

- We need seamless mechanisms for allowing diverse information sources to interoperate with all intended applications. Our ultimate goal is to develop models around which to effectively organize and access all of human knowledge.

- We need methods for ensuring appropriate levels of security, safety, and privacy among users of highly-interconnected resources. Theoretical research in cryptography, communication protocols, system security and network reliability already informs technological developments in this area, but more work is needed to handle the questions and decisions likely to arise in the future.

- To support the rapid development of the new multimedia applications, we need tools and services for discovering, locating and managing information distributed throughout interconnected networks. The structure, form, function, and location of these tools and services pose difficult algorithmic problems in a number of areas including network and protocol design, routing, flow control, scheduling, bandwidth optimization, and software production.

We therefore propose a research initiative on Information Access in a Globally Distributed Environment. The intent is to advance theoretical research motivated by the ubiquity of computers and communications, parallel and distributed systems, and large knowledge repositories. The goal is to provide the scientific foundation upon which we can design an infrastructure that permits universal access to globally distributed information resources in as cohesive and cost-effective a manner as possible.

5 The Algorithmic Stockroom

Although the area of algorithmic research has flourished for many years, these algorithmic advances have not consistently made their way into practice. Many promising algorithms described and analyzed in the theoretical algorithms literature have never been implemented, and it is often difficult for software designers to find public domain implementations even of classical algorithms covered in elementary textbooks.

Isolated steps have been taken over the years to remedy this situation in certain domains, for instance Grigoriadis's minimum cost flow code, the MINOS mathematical programming code, and more recently Goldberg's minimum cost flow and shortest path codes. Moreover, a large set of key algorithmic building blocks with uniform interfaces have been made available over the last few years by the LEDA project at the University of Saarbrucken.

To build and expand upon this foundation, we propose the creation of an Algorithmic Stockroom, a distributed repository of well-tested and documented implementations (in reasonably portable source code) of potentially useful algorithms and data structures. This proposal is both more and less ambitious than the LEDA approach. We wish to greatly increase the supply of easily-obtainable algorithm implementations, but in order to speed the process and allow the incorporation of many already-existing codes, we do not at the start want to impose strict rules as to algorithmic interfaces or the programming languages used in the implementations.

Operationally, the Algorithmic Stockroom could consist simply of a well-publicized website, with URL's pointing to locations of the relevant source codes, documentation, and test results, although some local archiving might be desirable. The site would also contain pointers to locations containing test data, and would serve as a clearing house for reports on user experience. There would be no a priori restrictions on the types of problem domains addressed by the algorithms in the stockroom, although one of the jobs of the Stockkeeper would be to maintain lists of key algorithms not yet incorporated, so as to help direct the efforts of those who want to participate in the project.

A funding agency such as the NSF would provide basic support for the Stockroom, but the major funding need we see is for the experimental research that would generate, test, and refine implementations to be added to the Stockroom, and for research that addresses practicality issues by incorporating components from the Stockroom into real-world applications.

6 Education

Theoretical computer science has a long tradition of insistence on broad-based education. This has facilitated the transition of many outstanding theoreticians to other areas of computer science and to industry. Consequently it is not surprising that many of the current leaders in other areas of computer science grew out of the ranks of theory. We need to reemphasize the importance of a broad-based education especially when rapid changes are happening in technology and applications. It is also imperative that the entire computer science community derives full benefit out of the vast array of fundamental ideas generated by the TOC community.

Thus graduate programs in computer science must be shaped to meet two broad goals. First, all graduate students, theoreticians and non-theoreticians alike, should receive a broader exposure to the developments in TOC. Second, young theoreticians must obtain greater exposure to practical research problems in computer science and other disciplines.

It would be beneficial for academic computer scientists at institutions throughout the United States to undertake a critical self-assessment of the relationships between theory and other research areas within their own programs. How much exposure do young theoreticians get to other areas of computer science? How is theoretical research presented to graduate students in other areas? Each institution has a responsibility to shape its educational program to meet the needs of its graduate students; perhaps as never before, those needs include familiarity with a broad spectrum of computer science rather than narrow specialization.

We also recommend several measures by which funding agencies can support efforts to strengthen the ties between theoreticians and practitioners. We focus on graduate and postdoctoral education, since curriculum development and research at the undergraduate and K-12 levels involve broader constituencies than just the TOC community.

- Develop fellowship programs to support summer or year-long visits by theory graduate students and postdocs to industrial research sites. These visits would allow students to collaborate on projects involving application of theoretical results to real problems.

- Develop programs to support summer or year-long visits by theory *faculty* to industrial research sites. Educators who are not familiar with applications of their research will not be able to pass this knowledge on to their students.

- Support multi-disciplinary, multi-institution, project-oriented research efforts, both large and small. The Human Genome Project is a good example of a large coordinated effort that supports the integration of theory and practice. It would be desirable to fund a range of smaller projects representing joint efforts by researchers with different areas of expertise.

7 Summary of Recommendations

Recommendation 1. It is essential to maintain a strong activity in core theory and fundamental algorithms.

Recommendation 2. Research which builds bridges between basic theory and applications should be given much greater priority. Funding for such bridging activity should come not at the expense of core theory, but rather by attracting funding from sources that have not traditionally supported theoretical computer science.

Recommendation 3. Efforts should be made to ensure that promising algorithms discovered by the theory community are implemented, tested and refined to the point where they can be usefully applied in practice.

Recommendation 4. Theoretical computer scientists should provide accessible expositions of their work, in expository articles aimed at a broad audience.

Recommendation 5. We recommend continuing support of research on the scientific foundations of software engineering and the verification of software systems, but with a strong preference given to investigations that are directly oriented toward the software development process.

Recommendation 6. The ongoing revolution in biology, with its emphasis on information processing within organisms, offers great challenges and opportunities for theoretical computer scientists. We recommend augmented funding for collaborations between theoretical computer scientists and biologists. This funding should come from the biology side as well as the computer science side.

Recommendation 7. Research in application-specific theory should address the roadblocks that limit the effectiveness of computing and telecommunications in our society. Proposals for theoretical work motivated by the ubiquity of computers and communications, parallel and distributed systems, and large knowledge repositories should be warmly encouraged.

Recommendation 8. Funding agencies should consider a new initiative concerned with information access in a globally distributed environment. Theoretical computer scientists can play a significant role in developing the foundations of this subject.

Recommendation 9. We recommend a new funding initiative, the Algorithmic Stockroom, as a vehicle for exploiting the knowledge of the TOC community for the production of efficient, portable and reusable software.

Recommendation 10. Funding agencies should support mechanisms by which theoreticians and practitioners (students and faculty) can work together and exchange problems and solutions. These include summer or year-long visits by theory graduate students, postdocs and faculty to industrial research sites and multi-disciplinary, multi-institution, project-oriented research efforts, both large and small.

Recommendation 11. Graduate students in theoretical computer science should be exposed to applications and experimental research, and graduate students in other areas of computer science should be exposed to the methods and results of theoretical computer science.

Acknowledgements Professors Ajoy Datta and Lawrence Larmore of the University of Nevada at Las Vegas hosted the NSF-sponsored workshop at which this committee began its work. Brian Bourgon and Jerry L. Derby of UNLV prepared a transcript of the discussions. Andrew Adamatzky, Eric Allender, Dana Angluin, Ronald V. Book, Anne Condon, Charles Elkan, Lance Fortnow, Peter Gacs, Susan Landau, Michael Luby, Jack Lutz, Leonard Pitt, Panos Pardalos, Serge Plotkin, David Shmoys, Carl Smith, D. E. Stevenson, Eva Tardos and Michael S. Waterman contributed to the preparation of white papers presented at the open meeting that preceded the workshop. David Applegate, Yoav Freund, Oded Golreich, Mike Kearns, Maria Klawe, Dana Latch, Nancy Lynch, Nick Pippenger, Leslie Valiant, Avi Wigderson and Mihalis Yannakakis are among the many who provided advice and assistance to the committee. We thank all these people for their help.

References

[1] A.V. Aho, D.S. Johnson, R.M. Karp(Chair), S.R. Kosaraju, C.C.McGeoch, C.H. Papadimitriou, P. Pevzner, "Theory of Computing: Goals and Directions," University of Washington Technical Report CSE-96-03-03, March 15, 1996.

[2] O. Goldreich, A. Wigderson, "Theory of Computing: A Scientific Perspective," Available from URL: http://theory.lcs.mit.edu/oded/toc-sp.html

[3] A. Condon, F. Fich, G.N. Frederickson, A. Goldberg, D.S. Johnson, M.C. Loui(Chair), S. Mahaney, P. Raghavan, J. Savage, A. Selman, D.B. Shmoys, "Strategic Directions for Research in Theory of Computing." Available from URL: http://geisel.csl.uiuc.edu/ loui/sdcr.html

[4] P. Raghavan, "Position Paper for the Working Group on Theory of Computation," Available from URL: http://www.almade.ibm.com/cs/people/pragh/position.html