COMPUTER SCIENCE STATEWIDE REVIEW

DRAFT PREFACE

Confidential Report and Recommendations

to

The Commissioner of Education of the State of New York

from

The Computer Science Rating Committee

Dr. Ruzena K. Bajcsy
University of Pennsylvania

Dr. Allan B. Borodin
University of Toronto

Dr. Barbara H. Liskov
Massachusetts Institute of Technology

Dr. Jeffrey D. Ullman, Chair
Stanford University

September 1992

Report of the Rating Committee for Computer Science: Preface

Part I: Computer Science — Is it a Discipline?

While the preface for a more traditional science or engineering discipline would never question its own existence, it seems necessary to raise a question that was once quite commonplace in academia. When computer science departments were first beginning to emerge during the mid and late 1960's, there was a great deal of debate as to what is computer science: could it really be viewed as a separate discipline with its own theories, paradigms and methodologies, could it be more than just the mastery of good programming techniques, or at best a subfield of applied mathematics and/or electrical engineering.

Part of the debate may very well be attributed to the terminology. Indeed, while "computer science" has become the most accepted name for the discipline in North America, alternative names still exist and many would argue that "information science" would have been a better choice. That is, one argument against "computer science" as a separate discipline was the (incorrect) assumption that the field is based solely on the study of a device[1] rather than being a broad-based quantitative and qualitative study of how information is represented, organized, algorithmically transformed, and used.

It seems safe to say that at present the debate is no longer concerned with the fundamental question of whether or not there is enough intellectual content to justify a discipline; rather the current debate is whether or not computer science is a "science," or is it really an engineering discipline and, as such, is it really synonymous with "computer engineering." Whatever the debate, the salient fact is that computer science has become an essential discipline in every major university. It has become so not only because of the widespread interest in undergraduate and graduate programs (and the continuing demand for graduates of these programs), but also because of the gradual recognition of the intellectual diversity and depth of the field and the essential role of computation in almost every other discipline.

- We believe that computer programming is an essential subject for every college graduate, and we believe that every student of Science or Engineering needs to study the concepts of Computer Science beyond programming, at a level comparable to university Physics or Multivariable Calculus.

---

[1] Webster's Dictionary defines computer science as "the study of computer hardware and software technology and operations." Not much of a science in that definition. Mathematics doesn't fare that much better being defined as "the science of expressing the relationship between quantities and magnitudes as represented by numbers and symbols."

A Taxonomy of Computer Science

Computer science is the discipline that deals with representation, implementation, manipulation, and communication of information. It is unique in that it deals with abstractions — representations of physical and mental processes — in all these four aspects. Thus, while mathematics deals with abstraction, it does not significantly study the implementation or processing of abstractions, and while physics deals with abstractions, it is limited to the study of physical phenomena. Computer science is in one sense similar to an engineering discipline that deals with designs, algorithms, and processes. But unlike any engineering discipline, computer science also turns inward and studies the process of creating designs and algorithms itself.

In this section we shall attempt to lay out the approximate boundaries of the field called computer science. It is impossible to make the borders precise, not only because the field is still evolving rapidly, but because there are a number of subareas that touch or overlap fields such as mathematics, electrical engineering, and psychology.

We shall divide computer science into three broad categories:

1. Computer science theory.
2. Computer systems.
3. Artificial intelligence and computer applications.

We should emphasize that the boundaries among these three are often fuzzy, and there are subareas that cut across boundaries.

Before proceeding, let us also mention the term "Computer Engineering," which is often heard in contrast to "Computer Science." Recent examinations of the field, such as the Taulbee Surveys to be mentioned later, have in recent years identified certain programs as Computer Engineering programs. These tend to be engineering-oriented programs, often closely allied with Electrical Engineering, and tending to emphasize hardware over software aspects of the broad field of computer science.

Theory

We can further divide computer science theory into two parts:

a) Algorithmic theory, and
b) Models and notations.

The theory of algorithms involves the discovery of new ways to solve problems quickly by computer. A great variety of algorithms and methodologies, each with some generality of application have been developed. This subfield has also given us what is probably the most "scientific" of developments: the theory of intractability. This theory can be used to demonstrate that many common problems can be solved by computer only through programs that take an extremely long time to execute. It is somewhat analogous to the physical laws of thermodynamics, which tell us we cannot build a "perpetual motion machine."

There are also many important notational systems that have importance in computer science. Many of these are a form of logic, used in applications ranging from the design of electronic circuits to proving that programs do what they are claimed to do.

Systems

Many different kinds of hardware and software systems are important today. These areas include:

a) *Computer Architecture*: the design of the computer itself. The cost of physical components has shrunk at an extremely fast rate for decades, and it is always a challenge to use what is available to increase the capability of machines. For example, today there is great hope that very large numbers of processors can be brought to bear on a single problem simultaneously ("parallel computing").

b) *Programming languages*: the notations in which one programs a computer, the abstractions that underlie these languages, and the technology for translating them into the language of the machine ("compiling").

c) *Operating systems*: the systems that manage the resources of the computer. Of special interest now are distributed operating systems and computer networks, which make resources of many, widely separated machines available to users at any one of them.

d) *Interfaces*: the systems that help the user to access and keep track of the resources of the computer. For example, graphics and systems involving interaction by mouse, pen, video, audio, and other media are of current interest.

e) *Database systems*: the systems that manage large amounts of data, usually shared among many users and often distributed among many machines, with different formats and assumptions about the meaning of data.

Artificial Intelligence and Applications

There is a core of artificial intelligence (AI) that addresses issues such as the representation of information and the algorithms for reasoning about information. In addition, there are a number of application areas that are very difficult and that are generally considered the province of AI. Interestingly, there are a number of technologies besides AI that are being brought to bear on such problems. For example, numerical analysis, which has its own core science, has recently been found critical in applications such as computer vision and manufacturing. An incomplete list of some of the important application areas is:

a) *Natural language*: the understanding and use of English or other languages by computers. Much harder than originally thought, high-performance natural-language systems seem to depend on developing a model of the domain of discourse suitable for use in the computer.

b) *Computer vision*: the extraction from a digitized scene of the essential elements of that scene.

c) *Computer-aided design and manufacturing*: the use of computer tools to facilitate the rapid and error-free development of products. Simulation of complex physical and electrical systems are part of the problem, as is the automatic fabrication of artifacts from their design.

d) *Robotics*: The control and planning necessary to have mechanical devices perform useful physical tasks.

Part II: A Brief History of Computer Science

We shall now trace the development of computer science as a separate discipline from its beginnings in laboratories to the present.

Computer Science — History and Trends up to 1970

North American academic interest in computing and algorithmic processes can certainly be evidenced in the late 1940's (if not before) at major universities such as Harvard, Princeton, and the University of Pennsylvania. Interest in computing and in computing-related academic subjects continued throughout the 1950's. This activity was primarily centered in mathematics and/or electrical engineering departments, although significant interest could also be found in other departments and in computer centers.

By the early 1960's, a relatively large number of universities had either established separate departments, or had well defined separate degree programs within other existing departments (again, primarily electrical engineering and to a lesser extent mathematics) or had formed an interdisciplinary degree granting program with members cross-appointed from a wide variety of related disciplines (e.g., mathematics, electrical engineering, physiology, linguistics, physics). By 1966, there were over 60 universities offering programs (under a diverse set of perspectives and program names) that would today be considered as part of computer science[2]. And while debates about the long term viability of computer science was probably at its peak in the late 1960's, it was already clear that the importance of computing, not only in science and engineering but in all aspects of human endeavor, and the pragmatic concerns of promoting the subject within universities based on departmental structure would insure that separate (or de facto separate divisions within say electrical engineering) computer science departments would be the norm.

These early programs led, reflected, or followed (depending on your perspective) the development of computing as an industry. That is, the first applications and studies were based on numerical analysis, or "scientific computing," as was the first successful high level programming language (Fortran). And because the design and organization of computers was still in such a formative period, the early programs also stressed topics such as logic design. Theoretical aspects growing out of logic design and computability theory gave rise to the "foundations of computer science" as then reflected in topics such as automata theory and formal languages. Also, in a few schools, the field of artificial intelligence was being developed.

Perhaps the easiest way to get a sense of the discipline as it is taking root in the mid-60's is to consider the list of undergraduate major courses as proposed by the ACM Curriculum Committee on Computer Science[3] in Fig. 1.

---

[2] F.S. Beckman, "Graduate Computer Science Programs at American Universities," *University Education in Computing Science* (A. Finerman, ed.), Academic Press, 1968, pp. 39–59.

[3] F.S. Beckman, *ibid.*

Introduction to Algorithmic Processes
Computer Organization and Programming
Numerical Calculus
Information Structures
Algorithmic Languages and Compilers
Logic Design and Switching Theory
Numerical Analysis
Computer and Programming Systems
Combinatorics and Graph Theory
Systems Simulation
Mathematical Optimization Techniques
Constructive Logic
Introduction to Automata Theory
Formal Languages
Heuristic Programming

Fig. 1. Course titles for "Curriculum 68" proposal.

The 1970's and the Enrollment Crisis

By the early 1970's, computer science had become a recognized and distinct program of graduate and undergraduate study in almost every major university (whether or not a separate department existed). Throughout the 1970's, the discipline both enjoys the opportunities and suffers the problems of dramatic enrollment increases both in graduate and especially in undergraduate programs. Enrollment pressures, coupled with the very limited supply of new Ph. D.'s being produced in the 70's, are the main ingredients for what the 1980 Snowbird Report[4] documents as a Crisis in Computer Science.

Reflecting the enormous growth and diversity in the computer industry, there is a corresponding growth in the research activities being pursued in the universities. In addition to a continuing interest in theoretical topics, there is now a widespread interest in the study of subjects such as programming language definition and compiling, large scale and interactive systems software (e.g., operating systems), graphics, and topics related to large scale "data processing" (to become known as "data base management systems"). It becomes exceedingly difficult for departments to make appointments in these latter areas because of the shortage of qualified doctoral graduates and because the academic departments are not able to compete with industry on the basis of salaries and, more important, on the basis of the space, equipment and infrastructure needed to carry on research in such areas.

In 1970, Orrin Taulbee begins to tabulate and document the problems concerning the shortage of qualified faculty and these annual reports (still called the Taulbee Survey)

[4] P. J. Denning et al. "A discipline in crisis," *Comm. ACM* 24:6, pp. 370–374.. The Snowbird Conference is a biannual conference originally attended by the chairs of Ph. D. granting computer science departments. Recently, it has enlarged its scope to include major industrial research centers and has become the conference of the Computing Research Association.

continue to the present time. Throughout the 1970's there are never more than 260 doctoral graduates per year at a time when various sources estimate a need for 1600 Ph. D.'s per year.

Because of the diversity of research interests and the shortage of staff, it becomes apparent that few universities will be able to pursue all or even most of the active research areas (nor, of course, produce Ph. D.'s with expertise in such areas that are not supported). Most departments opt to view topics such as digital design and computer architecture as being in the domain of electrical engineering. Many departments effectively decide not to pursue topics in artificial intelligence. Indeed even by 1980 there are really only three centers for artificial intelligence (Carnegie, MIT, and Stanford) in the U.S., although a number of other excellent programs do have some limited activity in this area. One other noticeable trend is that scientific computing becomes relatively less important as departments become staffed either by theoreticians or "systems people."

At the end of this period, a study of the nature of computer science is commissioned by the National Science Foundation.[5] It divides the field into ten subfields, listed in Fig. 2.

Numerical Computation
Theory of Computation
Hardware Systems
Artificial Intelligence
Special Topics (principally algebraic manipulation, natural language, and computer vision)
Programming Languages
Operating Systems
Database Management Systems
Software Methodology
Applications

Fig. 2. The COSERS subfields.

The 1980's: Computer Science Matures

During the early 1980's, the conditions of the 70's that have led to this crisis in the discipline have begun to be addressed. Reacting to arguments put forth in reports such as the Feldman Report,[6] NSF begins a successful program to provide leading departments with the equipment and infrastructure needed to carry out the more experimental activities of the discipline.

The discipline gets some "enrollment relief" in the mid 80's in that enrollments at the undergraduate level undergo a sudden reversal. There is a significant reduction in both course enrollments and degree majors at the undergraduate level. Graduate enrollment

[5] B. W. Arden (ed.) *What Can Be Automated: The Computer Science and Engineering Research Study*, MIT Press, Cambridge.

[6] J. A. Feldman and W. R. Sutherland (eds.), "Rejuvenating experimental computer science," *Comm. ACM* **22:9**, pp. 497–502, Sept., 1979.

levels do not experience the same reduction but do level out by the mid 1980's. But even at these somewhat reduced levels, computer science departments continue to have relatively high student/faculty ratios especially when compared to any of the mathematical or physical sciences. And the large student/faculty ratio is also present in Ph. D. supervision levels, particularly at the major universities which are producing the bulk of the new doctoral graduates.

Increased Ph. D. production and faculty growth are the major factors in alleviating the documented crisis of the 70's. The 1984–85 Taulbee Survey reports 103 Ph. D.-granting computer science departments (8 in Canada), with a total of 1741 faculty, graduating 326 Ph. D.'s. The 1984 Snowbird Conference senses that "the corner had been turned."[7] Indeed, the 1986 Snowbird Conference now begins to be more concerned about research funding for all these new faculty members and in the 1987–88 Survey, there is for the first time a real concern about possible Ph. D. overproduction. In that year, 127 Ph. D. granting computer science departments (12 in Canada) with a total of 2427 faculty, graduate 577 Ph. D.'s.[8] Throughout the 1980's, the Taulbee Survey estimates the domestic demand for Ph. D.'s to be between 600 and 1000 graduates per year.

Algorithms and data structures	Theory of computation
Programming languages	Languages
Software methodology and engineering	(included in "Languages")
Architecture (of hardware)	Computer Systems
Operating systems	(included in "Computer Systems")
Numerical and symbolic computation	Scientific computing
Databases and information retrieval	Information systems
Artificial intelligence and robotics	Artificial Intelligence
Human-computer interaction	Graphics/User interface
(a) Denning Report	(b) NSERC

Fig. 3. Two modern taxonomies of computer science.

Near the end of the 80's, a new curriculum proposal attempts to define the field and revolutionize education in computer science.[9] This report emphasizes that each subfield contains elements of theory, abstraction, and design, and it then partitions the discipline into the subfields listed in Fig. 3(a).[10] Notice how these correspond closely to the topics in the COSERS report (Fig. 2) and also to a modern partition into subfields used by

---

[7] P. Freeman and P. Young, "The 1990 Snowbird Report: A Broadening Perspective," *Computing Research News*, 2:4, Oct., 1990.

[8] D. Gries and D. Marsh, "The 1987–1988 Taulbee Survey," *Comm. ACM* 32:10, pp. 1217–1224, Oct., 1989.

[9] P. J. Denning et al., "Computing as a discipline," *Comm. ACM* 32:1, pp. 9–23, Jan., 1989.

[10] This list was also endorsed by the recent report on Computer Science: *Computing the Future: A Broader Agenda for Computer Science and Engineering* (J. Hartmanis and H. Lin, eds.), National Academy Press, Washington, 1992.

the Canadian NSERC (National Science and Enginering Research Council) for grants in computer science shown in Fig. 3(b), although the terminology used in the three taxonomies differs.

Computer Science — The Demand and Trends for the 90's

As mentioned previously, the Taulbee Survey continues to estimate the annual demand for Ph. D.'s to be in the range of 600–1000. In this regard, it might be advisable to consider the combined production of Ph. D.'s in computer science and computer engineering which in academic year 89–90 was 734 – 173 = 907.[11] If the estimated demand is correct, then (depending on the economy) it appears that the present supply versus demand is in balance.

It is important to recognize that academic demand for new Ph. D.'s is clearly going to be much less dramatic in the 1990's than in the previous decade. Moreover, the demand will be mostly concentrated in the lower ranked and smaller Ph. D. granting departments, and in the non Ph. D. granting departments. The "planned growth" by the mid 1990's over all Ph. D. granting departments (both CS and CE) is about 24%, but historically these estimates are always overly optimistic. The present budgetary restraints in almost all universities will severely restrict faculty growth no matter how warranted the case for such growth. And computer science can expect another decade with relatively little attrition due to death or retirements. (In the late 1980's, computer science departments lost only 0.7% per year of its faculty due to death or retirement.) Computer science is still a "young discipline" in that there are more assistant professors than full professors.

It is also important to recognize that while certain areas of computer science will continue to be in short supply, employment opportunities in both academia and industry will be more problematic for certain subfields (e.g., in the more theoretical areas). On the other hand, any computer science graduate who is well educated in a sufficiently broad set of interests will continue to find many opportunities in industry and (at least) the small academic departments. In particular, one observes in industry a growing need for more precise (i.e. mathematically precise) specification of systems requirements and performance analysis, and thus one sees a general willingness within industry to consider even the more theoretical areas of computer science as relevant areas of study. Universities are often more specialized than industry in their hiring criteria.

While the potential demand for Ph. D. computer science graduates should remain high throughout the present decade, it will also be the case that more will be expected of such graduates. For example, a compiler designer for a language to be executed on a parallel multiprocessor architecture will have to have some expertise about computer architectures and communication primitives, about the mathematical analysis of data dependencies and code optimization, about the variety of critical code sections used in scientific computing and engineering design, about the menu of interactive tools users now expect, and so on. That is, in contrast to the diversification of the discipline throughout its history, new Ph. D.'s will be expected to be able to synthesize contributions from all these areas while still displaying a somewhat unique expertise in some area of specialization. In short, those

[11] D. Gries and D. Marsh, "The 1989-1990 Taulbee Survey Report," Computing Research Assn., Dec., 1990.

who can combine broad-based knowledge and experience with demonstrated expertise in any well motivated aspect of computer science will be in great demand by both industry and academia.

Some would argue that such pressures for synthesis and for obtaining industrial support will inevitably move computer science more towards engineering with the eventual merger of computer science and computer engineering departments wherever they now co-exist. To be sure, a number of major U.S. departments have gravitated from Arts and Science faculties to become part of Engineering faculties. But such changes are often based more on pragmatic concerns about space, budget, and new faculty positions rather than on any discernible intellectual shift. (Of course, we cannot underestimate the influence of such pragmatic concerns.) But for the foreseeable future, academic computer science will continue to be "somewhere between the sciences and engineering." Those faculty members with major interests in theory, foundations, scientific computing and artificial intelligence will generally feel more comfortable viewing the discipline as being, at least in part, a scientific discipline. Those faculty with major interests in languages, software systems, software engineering, architectures and networks may indeed generally welcome a shift towards engineering, which has traditionally been more supportive of prototype research projects (not necessarily oriented towards publications) and consulting activity.

Throughout the 90's, there certainly will be more opportunities for consulting and more faculty demands for "buying out" of teaching. But most computer science departments have long standing informal or formal understandings about such arrangements and most departments strongly welcome industry/university collaborative research. Perhaps if any trend will characterize the 90's it will be the continued emergence and growth of collaborative computing research consortiums, funded by both industry and government. Such consortiums will combine a number of different departments from geographically close universities together with major industrial partners and sponsors. The overwhelming consensus is that such consortiums are essential if the United States is to be able to maintain its traditional leadership (and past dominance) of computing technology. Here the challenges from Japan and the Pacific Rim, and from the European Common Market are all too apparent. Given the intellectual and economic challenges of the discipline, computer science will remain an attractive and exciting field for doctoral work.

Demographics

Some tables abstracted from the 1989-1990 Taulbee Survey will give the reader a sense of how the doctoral population has grown over the years. Figure 4 gives North American Ph. D. production for computer science only (not computer engineering) at five year intervals, and every year for the most recent half-decade. It also indicates the percentage of women and certain minorities in each graduating class.

We can observe certain trends.

- After relatively stagnant growth over almost two decades, the number of doctoral graduates has recently exploded and shows no sign of diminishing.
- The fraction of women in the population has grown slowly over the years, but has never exceeded 14%.

Year	Ph. D.'s	Women	Black	Hispanic	Foreign
1970	112	1	1	?	22
1975	256	21	1	?	68
1980	230	28	0	?	82
1985	326	32	3	7	122
1986	412	50	6	6	184
1987	466	51	1	8	181
1988	577	60	4	5	238
1989	625	87	0	6	248
1990	734	97	3	8	331

Fig. 4. Growth in computer science Ph. D. population.

- Blacks and Hispanics are severely underrepresented in the population, and the numbers of graduates in these two categories has been relatively constant recently.
- The number of foreign (not US or Canada) graduates has increased both in numbers and percentage, growing from 20% in 1970 to 45% in 1990.

Certain groups that are formally designated as "minorities," such as Asian-Americans, have never been underrepresented in computer science. Thus, when we refer to "minorities" in the balance of this report, we are speaking specifically of underrepresented groups: Blacks, Hispanics, and Native-Americans.

First Jobs of Doctoral Graduates

The 1990 Taulbee survey also contains some significant data about where the Ph. D. graduates took jobs. Figure 5 breaks down the 555 graduates whose employment was known and who are located in the US or Canada, by type of first job.

Academia	285	51%
Industry	228	41%
Government	14	3%
Self-employed	17	3%
Unemployed	11	2%

Fig. 5. Distribution of first jobs in 1990.

Conclusions

Computer science remains a vital and important field, one for which doctoral production has just recently become adequate. It is not unreasonable to predict a tightening of the job market, since doctoral production shows no sign of turning downward, and we expect at least a 6-7 year lag between the time that a job market (for Ph. D.'s) tightens and the time we begin to see decreased production in response. Further, the youth of the CS doctoral population indicates that there will be little demand generated by retirements in the near future.

On the other hand, there are still hundreds of 4-year colleges that have as yet been unable to hire a single computer scientist for its faculty, so we believe the demand for new Ph. D.'s is not likely to dry up quickly. It is interesting to note that of the 285 Ph. D.s entering academia (see Fig. 5), 198, or 69%, took positions in Ph. D.-granting institutions. We thus believe that opportunities in the field will remain good, although many graduates will have to reduce their expectations of a job at a first-rate CS department or research lab.

Part III: The Ideal Computer Science Department

Many of the criteria used to evaluate a CS department or its doctoral program apply equally to all or most other disciplines. These include quality of faculty and students, availability of space and facilities, national awards and fellowships, and similar criteria. There are also a number of conditions special to CS that require consideration. Among these, we count the "mainstreamedness" of the program and issues relating to the source and destination of the doctoral students.

What Makes a "Mainstream" Computer Science Department?

Being a relatively young field, we find that many of the faculty now teaching the subject were educated in other disciplines; indeed twenty years ago, that was true of essentially every teacher of the subject. There is a "computer-science culture," which includes a readiness to develop algorithmic solutions to problems, a concern for dealing with extremely complex systems, and/or a concern with modeling, abstraction, and notation. It is increasingly hard for people not educated in this culture to substitute for people who were so educated.

Scientists from nearby fields such as Mathematics and Electrical Engineering, from which most early computer scientists came, melded to varying degrees with this culture as it was forming. Some became founders of the principal subareas of CS, while others taught CS but retained research interests in areas that were outside the core of CS. Schools whose first computer scientists became fully part of the culture were at an advantage. They produced the first of the "native" computer scientists; those who were educated in the culture, and these institutions were typically the places most attractive for the new wave of computer scientists, who had their pick of research labs and academic institutions.

As a result, we have had a bimodal distribution of schools. Some have had mainstream computer science activities for two decades, and these tend to be the better programs. Others have had to staff very large undergraduate classes (undergraduate CS enrollment

peaked at 8% of all undergraduates in the mid-80's) without having access to the very limited supply of "native" CS faculty.

It is only in the past few years that the supply of CS-trained Ph. D.'s has reached a level allowing this second group of schools to hire such people. However, since schools have had to staff their CS courses somehow, there is a tendency for each institution to create a CS department out of whatever cloth it has had available. Now that people with more central interests are available, it is not always feasible or humane to substitute them for the people on the periphery who have seen the institution through very hard times and have performed admirably.

Nonetheless, we feel it is not appropriate today to produce CS Ph. D.'s who do not have a very solid grounding in the basics of the field, any more than it would be appropriate in Physics or Mathematics, for instance. Thus, a very important criterion for evaluating the suitability of a doctoral program is

- Does the program have a substantial staff of well-educated, core computer scientists, or alternatively, are there concrete plans to hire such staff in the near future?

An additional factor to consider is the relationship between theory and practice in the skills of the faculty. While theory is an essential part of the mix, there is a tendency for theoreticians to gravitate to academia, and therefore, a school can find itself with a preponderance of faculty who are interested in the theory of the subject. We therefore prefer to see a program in which the faculty included both theoretically inclined people and those more interested in system building.

Faculty Quality

There are a number of criteria that distinguish an excellent faculty. Somewhat unreliable heuristics include counting publications in high-quality journals and conference proceedings, considering the quality of the institutions from which they graduated, or the professional offices that they hold or have held. Likewise, the faculty should have adequate research support. Support from NSF is probably an indication of good standing among their peers, but many other agencies, including the defense department agencies, DARPA, ONR, AFOSR, and ARO, have high standards in what they support.

Among the younger faculty, we would expect those who are eligible to have won NSF Young Investigator (NYI) awards from the National Science Foundation (formerly known as PYI — Presidential Young Investigator), or the rarer Packard fellowships, ONR Young Investigator awards, or PFF (Presidential Faculty Fellows) grants. Many of the senior faculty are now reaching the age where they might be elected to the National Academies of Science or Engineering, or have won the Turing award ("Nobel prize" for computer science), or other prestigious national or international award.

Student Quality

Computer science is blessed with a graduate student population that is very strong. A good heuristic measure is the scale used by ETS for the advanced tests such as computer science. The raw (not percentile) scores on the advanced test are calibrated so that the typical student taking the CS test will score the same on that test as on the average of the verbal and quantitative parts of the general GRE. Thus, we can look at the raw score scale of the CS GRE as a measure of the quality of students destined for CS. When the CS GRE was first offered, its scale was second only to Mathematics, and comparable to the scales for the physical and biological sciences. As the field grew and the CS GRE began to be used as an exit exam for seniors at some schools, the scale has dropped but is still above that of the engineering exam and above the social sciences and humanities.

It is not atypical for a strong school to fill its ranks from people who have GPA's near 4.0 and quantitative GRE's in the 98-99th percentile. Domestic students will also score very high in the verbal and analytic portions. Many schools accept students with backgrounds other than CS, so one tends to see more variation in the CS GRE scores. We should add that strong recommendations, especially those reflecting research experience as undergraduates, form an important admissions criteria.

Finally, a strong program usually has at least a few students who are winners of national fellowship competitions. These include NSF and Hertz, for example, although not all schools are eligible for the latter. There should also be a number of industrial fellowships, such as those sponsored by IBM, ATT, and a number of other corporations.

Student Origins

As in many disciplines, there is a strong international demand for education ~~in computer~~ science. Because the selectivity factor is so much greater, those foreign students who do make it into programs in North America are generally extremely talented, so one might be tempted to view a large foreign student population as indicating a good program. However, there is in truth more competition for domestic applicants because, for example, they typically do not have language problems that make them poor teaching assistants, and they are eligible for many fellowships not open to foreign students. Thus, the domestic applicants can "vote with their feet." We therefore view favorably a program that can attract high-quality domestic applicants, and a significant majority of foreign students is a warning that the program may be experiencing difficulty.

However, we recognize that there could be cases of a departmental or institutional decision to accept outstanding foreign students in preference to average domestic applicants. We do not wish to take a stand on the appropriateness of such a policy. Thus, we would also view favorably a student population dominated by foreign students of recognizably high quality, especially if the applicant pool gave evidence that the school was also sought after by some domestic applicants.

Where do the Students Go?

Generally, the best doctoral students have found positions at either top-level research laboratories or top academic institutions, although a fraction have gone from graduate school directly into a variety of small companies, especially start-ups that offer equity positions. A top research lab is characterized by offering its doctoral-level employees a great deal of freedom in setting their own research goals. We must be careful about circumscribing the set of "top" research labs, since there are many places where at least a few employees have the requisite freedom to set goals. As some examples, it is generally accepted that the IBM labs at Yorktown Hts. and Almaden fall into this category, as do the ATT Bell Labs at Murray Hill and the Bellcore lab at Morristown. Likewise, Xerox PARC and DEC SRC in Palo Alto qualify. A number of major Japanese electronics firms, such as NEC, Mitsubishi, and Matsushita (Panasonic) are in the process of starting laboratories for basic computer science research in the US as well.

It is easier to determine the schools that are ranked highly, since periodic surveys are taken. The Taulbee studies group schools by "first 12," "second 12," and so on.

When looking at the recent history of student placement, we would expect to find that many of the graduates have gone either to the sort of research lab mentioned, to a start-up, or to a school in the first few dozen. In the very recent past, positions of this type have been much harder to get, and that trend will probably continue. Thus, we do not perceive a decline in the percentage of students obtaining these "good" jobs as an indication of falling quality at the graduating institution.

Support for Research Students

A sound program will be able to raise enough external research funds to support most or all of the doctoral students who are performing research. Ideally, coupled with other sources of support such as teaching assistantships, all full-time doctoral students are supported.

Teaching Load

A faculty member who is conducting an extensive research program and supporting a number of graduate students through that research probably does not have time to teach more than one course at a time, especially if the courses are large-enrollment basic courses for undergraduates. Thus, a program where the teaching load is significantly higher for faculty who are heavily involved in Ph. D. training may have trouble providing adequate instruction in research for these students. While we would not dictate an overall policy regarding course loads, we do look at the de facto load on the people most closely connected to the doctoral program as one indicator of the quality of the Ph. D. program.

Intellectual Ambience

There should be a sense of community among the faculty and their doctoral students. The faculty should be accessible to the students, not spending excessive time at home or consulting. There should be regular seminars in subjects of active research, and students should have access to speakers and visitors. If appropriate to the locale, there should be active participation in the program from nearby industrial researchers, and regular contacts with this outside research community.

Library Facilities

We expect a strong program to have an extensive collection of journals in CS, including all journals of ACM, IEEE Computer Society, many from SIAM, and a number of important journals from commercial publishers. Book collections are equally important. There must be easy access to the collections, especially for students at distributed sites and part-time students.

Computing Facilities

The cost of workstations has dropped considerably in the past few years. At the same time, manufacturers have shown great enthusiasm for exposing students to their wares. Thus, it is usually feasible for a graduate program to arrange for a workstation per doctoral student, or something close to that, especially if they have an adequate budget for support staff to maintain the system software. It is normal today for these computers to be connected on a local-area network such as ethernet and for there to be access to laser printers and file servers on such a net. Internet access for electronic mail and other forms of exchange among computer scientists is also the norm.

We would expect a strong CS doctoral program to have such an environment, and we would be concerned about a program that made available to its students little equipment, or outmoded equipment. We would also be concerned about a program where modern equipment was available, but only through research groups, rather than as an entitlement of the Ph. D. program.

Recruitment of Women and Minorities

We look favorably upon a program that has made special efforts to attract and support women and underrepresented minorities (Blacks, Hispanics, and Native-Americans) to its faculty and student body. However, given the demographics of the field suggested earlier by Fig. 4, it is hard to fault a program that has not been successful at bringing such people to its faculty or doctoral student body. There is some evidence that the fraction of women among graduating Ph. D.'s is rising. A significant increase in the number of underrepresented minorities in doctoral programs or faculty will have to wait until there is growth of these populations at lower levels in the educational chain.

Computer science is a discipline critical for the world technology base. When we see a population significantly underrepresented among computer scientists, we assume that the world is missing a large number of talented people that, if properly motivated to enter the field, could become important contributors. While the solution to the problem of

Intellectual Ambience

There should be a sense of community among the faculty and their doctoral students. The faculty should be accessible to the students, not spending excessive time at home or consulting. There should be regular seminars in subjects of active research, and students should have access to speakers and visitors. If appropriate to the locale, there should be active participation in the program from nearby industrial researchers, and regular contacts with this outside research community.

Library Facilities

We expect a strong program to have an extensive collection of journals in CS, including all journals of ACM, IEEE Computer Society, many from SIAM, and a number of important journals from commercial publishers. Book collections are equally important. There must be easy access to the collections, especially for students at distributed sites and part-time students.

Computing Facilities

The cost of workstations has dropped considerably in the past few years. At the same time, manufacturers have shown great enthusiasm for exposing students to their wares. Thus, it is usually feasible for a graduate program to arrange for a workstation per doctoral student, or something close to that, especially if they have an adequate budget for support staff to maintain the system software. It is normal today for these computers to be connected on a local-area network such as ethernet and for there to be access to laser printers and file servers on such a net. Internet access for electronic mail and other forms of exchange among computer scientists is also the norm.

We would expect a strong CS doctoral program to have such an environment, and we would be concerned about a program that made available to its students little equipment, or outmoded equipment. We would also be concerned about a program where modern equipment was available, but only through research groups, rather than as an entitlement of the Ph. D. program.

Recruitment of Women and Minorities

We look favorably upon a program that has made special efforts to attract and support women and underrepresented minorities (Blacks, Hispanics, and Native-Americans) to its faculty and student body. However, given the demographics of the field suggested earlier by Fig. 4, it is hard to fault a program that has not been successful at bringing such people to its faculty or doctoral student body. There is some evidence that the fraction of women among graduating Ph. D.'s is rising. A significant increase in the number of underrepresented minorities in doctoral programs or faculty will have to wait until there is growth of these populations at lower levels in the educational chain.

Computer science is a discipline critical for the world technology base. When we see a population significantly underrepresented among computer scientists, we assume that the world is missing a large number of talented people that, if properly motivated to enter the field, could become important contributors. While the solution to the problem of

the missing computer scientists is not an easy one, and it cannot be solved primarily at the doctoral level. we would hope that every CS Ph. D. program would have made some attempt to contribute to the solution, even if the numerical results are not impressive.

University Commitment

There are several special factors concerning CS that put a burden on the university administration. An important indicator of the potential for the program is how well the university has managed to cope with these problems.

Space is a very common problem for CS departments. Since most CS departments have experienced considerable growth during the past decade, it is quite normal for the department to outgrow whatever space it had been given. A university with a strong commitment to CS must deal with the space problem effectively. As a laboratory science, there is need not only for offices, but often for rooms housing clusters of workstations, robotics labs, and possibly other kinds of specialized equipment. Ideally, there should be enough room that every Ph. D. student has a desk, whether or not he/she is working for a project. Finally, there should be sufficient extra space that there is room for an occasional visitor, since such outsiders often enrich the academic life of the students (and faculty) immensely.

In addition to examining whether space needs for the department have been met, we also need to look to the future. Are there concrete plans to solve whatever space problems now exist? Are there plans that will allow future growth of facilities as the faculty and/or student body grow?

A second area in which a university needs to show its commitment is in the matter of academic computing facilities. The CS department is dependent on academic computing facilities at all levels of instruction, from introductory programming to advanced courses such as operating systems. The arrangement for providing such facilities may vary from school to school. Some facilities may be managed by the central administration, while the CS department may have direct responsibility for the equipment in other cases. Regardless of the actual arrangement, there should be evidence that the university is providing adequate support for modern instructional equipment and infrastructure, including networking and professional staff.

A Deficient Program

Certainly, not all programs can meet the ideals suggested above, and we are not suggesting that programs not meeting all of these criteria are unacceptable. Yet the demographics of the field suggest that there is not the justification for weak programs that may have existed in the past. Therefore, we would question the need for programs that fall below the ideal in several areas. Some important warning flags are:

- Faculty is composed primarily of people with principal interests outside the mainstream of CS, or composed primarily of people with expertise exclusively in one area of CS.
- Faculty rarely receives recognition appropriate to their age group, e.g., national awards and editorships of major journals.

- External funding for research is low or nonexistent.
- Student body is composed primarily of foreign students of less than outstanding qualifications or of weak domestic students.
- Attrition rate is high, and frequently more than six years are required to complete the doctorate.
- Students often have quantitative GRE's less than 700 and rarely win national fellowships.
- The consumers of CS Ph. D.'s (academia and industry) appear unenthusiastic about most of the graduates of the program, as reflected in placement records.
- Facilities — libraries, computing power, networks — are inadequate.