Microelectronics is part of the answer to making information systems both affordable and sophisticated enough for widespread use, but the innovation challenge is not just technical. It is matching the broad spectrum of new technologies to opportunities for serving the information and education needs of contemporary society.

References and Notes

1. Definitions of VLSI abound, but *very* large scale integration is sometimes considered to start at about 5000 MOSFET (metal oxide semiconductor field-effect transistor) logic circuits, or more than 16,000 memory bits, per chip.
2. P. F. Olsen and R. J. Orrange, *IBM J. Res. Dev.* **25** (No. 5), 405 (1981).
3. Paper to be presented at the International Solid State Circuits Conference, in February 1982.
4. V. L. Rideout, *IEEE COMPCON 80 Technical Digest* (Institute of Electrical and Electronic Engineers, New York, February 1980), pp. 2–5.
5. _____, in *Digital Technology Status and Trends*, H. Pinke, Ed. (Oldenbourg, Munich, 1981), pp. 11–48.
6. A. L. Robinson, *Science* **208**, 480 (1980).
7. P. M. Russo, *IEEE Trans. Electron Devices* **ED-27**, 1332 (1980).
8. F. Jelinek, *Proc. IEEE* **64**, 532 (April 1976).
9. P. A. Strassmann, *Technol. Rev.* **82**, 54 (December/January 1980).
10. L. M. Branscomb, *Comput. Networks* **5**, 3 (1981).
11. N. Mokhoff, *IEEE Spectrum* **18**, 57 (April 1981).
12. L. M. Branscomb, *J. Commun.* **31**, 143 (winter 1981).

Computers: A Survey of Trends and Limitations

Joel S. Birnbaum

Summary. Enormous progress in electronic technology is accelerating the use of computers in everyday life. In this article trends in hardware, input-output technology, computer architecture, software, communications, and artificial intelligence are examined and complexity is identified as a limitation to further progress. Promising directions of research, which may extend the range of computer applications, are discussed.

The popular press and some computer scientists would have us believe that we are on the verge of the realization of an age-old dream: intelligent, obedient machines relieving people of much of the hazard and drudgery of life and extending mental capabilities and creativity through the filtered delivery of information from a global range of sources. The evidence presented for this is the growing use of inexpensive microprocessors in homes, laboratories, and factories, the availability of new communications technologies, and the extrapolation of the everyday uses of computers as calculating engines for scientists and as data processors for businessmen. Increasing computer literacy in the current generation of students is expected to multiply the rate of development of new applications. Yet many of the limitations to our progress today are precisely those that have prevented many of the sanguine projections of a decade ago from coming to pass. These are not fundamental limitations imposed by physical laws; rather, they result from our difficulty in dealing with complexity. Software development productivity has suffered most from human-introduced complications (see Table 1), compounded by approaches that are seldom disciplined or scientific (*1*).

In this article trends in computer technology, including hardware, interfaces, architecture and organization, software, communications, and artificial intelligence are examined, and major changes that seem likely in the design and application of computers are indicated. The improvements in both hardware and software technologies are now so great that basic reformulation of our design assumptions may be necessary, and greatly increased attention to tools and technology to combat complexity is called for.

The author is director of the Computer Research Center, Hewlett-Packard Laboratories, Palo Alto, California 94304.

Trends in Hardware: Logic, Memory, and Storage

The startling advances that have been made in logic, memory, and storage technologies are all possible because a binary digit has informational value independent of its physical size. Technologists have produced ever faster, cheaper, and less power-hungry logic and memory elements simply by shrinking them; much of the research in computer technology can be thought of as a quest for smallness. As objects change radically in size, scaling relations between surface area and volume become important, and much new science has been driven by a need to understand the nature of surfaces and thin films and the nature of the interactions at interfaces between dissimilar materials.

As critical dimensions approach 1 micrometer, the wavelength of light becomes a limiting factor to attainable lithographic resolution. Progress in electron beam– and x-ray–based techniques will circumvent this problem, and most technologists do not think lithography will prove to be a limitation for the next 10 years (*2*).

A more serious limitation is signal propagation speed, which is about 1.5 nanoseconds per foot in contemporary packages. To achieve 1 billion instructions per second in a conventionally organized machine, the basic instruction cycle of 1 nanosecond can have a maximum path length of only a few inches. Very fast conventional machines must therefore be extremely small, and so component interconnection assumes paramount importance. Densely packed devices running at high speeds require efficient cooling mechanisms. One of the boldest attempts to solve this problem is being pursued by researchers, primarily at IBM, who are investigating superconducting Josephson junctions that promise two to three orders of magnitude less power consumption than conventional silicon technologies. While the laboratory demonstrations of this technology are very impressive, the difficulties imposed by extraordinarily critical manufacturing tolerances and the need to

0036-8075/82/0212-0760$01.00/0

maintain these tolerances under repetitive cycling between liquid helium and room temperatures for servicing in user environments seem to indicate that practical applications are still many years off (*3*). Cooling of conventional silicon devices to liquid-nitrogen temperatures is also being explored, and such devices will probably be used in products within the decade.

Modern computers contain meters of metallic connections on and among the silicon chips, and so design automation systems of great sophistication are evolving to handle the complicated interconnection and placement problems. Standardized gate arrays, called masterslices, can be fabricated with high yield and good performance characteristics, and reasonably automated tools are being developed to assist with the customization of these arrays to accomplish specific functions. It appears probable that in most future computers this approach will be employed for all but those components where the ultimate in density or performance (or both) is required, or where volume production justifies the densest configuration (as in microprocessors). In the latter case, custom design techniques must be used, but even here a degree of automation is becoming realistic as hierarchies of predefined structures can be coordinated with computer programs employing high-level graphic interaction. Automated test generation is the subject of much study.

Beyond the interconnections on and among the chips, intercard, -board, and -frame wiring also requires careful planning, and multilevel packaging schemes can be expected to proliferate, particularly in medium- to high-performance machines (Fig. 1).

Packaging and design automation are two factors pacing hardware progress. Many interesting approaches to these related areas are emerging, and we can be most optimistic about the outcome (*4*, *5*).

Mechanically based magnetic recording is still the principal low cost, large volume, fast access storage technology, and improvements in the density and data rate parameters of magnetic disks, drums, and tapes are projected to continue through the 1980's. Two trends are evident as the cost of analog and digital electronics decreases: (i) more use of powerful error-correcting code circuitry, permitting information to be stored more densely and read more quickly, and (ii) attempts to store more than one bit of information per magnetic domain by using linear multilevel recording and signal processing techniques. However, the limitation in storage for an increasing number of applications is neither recording density nor data rate but access time. It appears likely that improvements will come from increasing the number of independently actuated heads per surface, requiring research in making the heads and their actuators smaller and cheaper.

Table 1. Summary of data processing industry trends [(*1*), p. 171] (normalized to 1955).

Indicator	1955	1965	1975	1985
Industry growth	1	20	80	320
Performance/cost	1	10^2	10^4	10^6
Programmer productivity	1	2.0	2.7	3.6

Write-once optical disks are a subject of intense research; they offer the potential of vast capacity (trillions of bits) at a cost less than the rental of the volume of office space necessary to store paper containing an equivalent amount of information. Archival systems that save all information and can retrieve it in a reasonable amount of time could become a reality soon, and reconsideration of prejudices regarding the permanent retention of information may be in order.

In summary, no fundamental limitations to progress in logic, memory, and storage are evident, and advances in these areas are expected by technologists to continue at the present rate for the next decade. Research challenges will be keenest in the packaging and design automation areas, where little understood pattern recognition and judgmental skills of humans must be augmented or replaced by algorithms.

Trends in Input-Output Technology

Contrary to intuition, it is perhaps in the area of access to computers that the quest for smallness may ultimately pay the greatest dividends. In fact, vast reductions in cost and improvements in performance already enable humans to communicate through computers in a more natural manner. Speech recognition products are already in existence for limited vocabularies of isolated utterances, as are fairly realistic speech synthesizers. Recognition of certain forms of handwritten or printed input (either tablet- or scanner-generated) and a panoply of printing devices that provide low-cost, high-quality publication of text and images are now commonplace. Interactive graphics devices of high resolution, some with color, will soon be a staple on most computers.

It is now cost-effective for a wide range of sensors and effectors to be connected to computers and, conversely, for instruments to contain computers. The distinction between instruments and

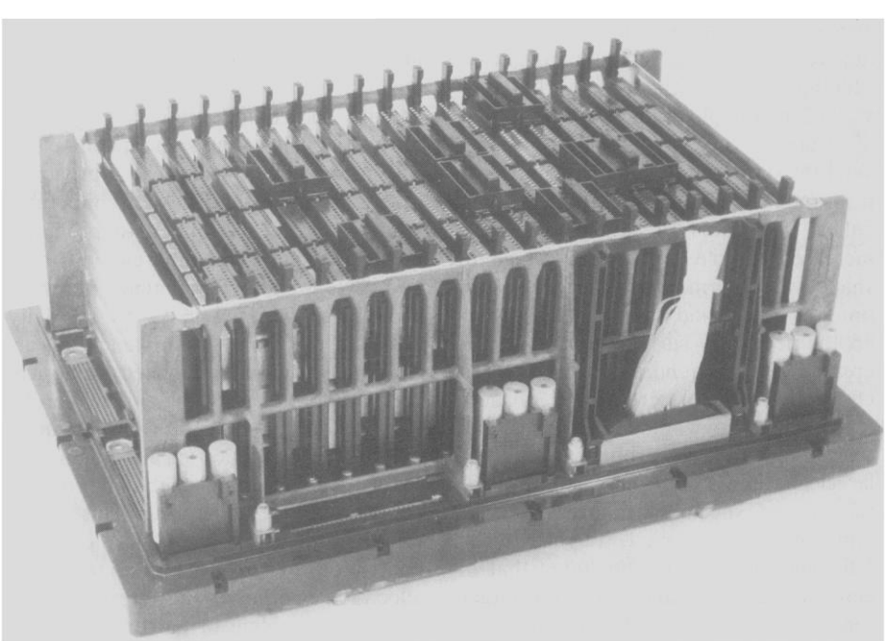

Fig. 1. Card-on-board packaging used in IBM 4300 series processors. The 100,000 circuits inside IBM's 4331 and 4341 computers have three levels of interconnection: LSI chips in multilayer ceramic modules are soldered to subsidiary multilayer printed-circuit cards that in turn plug into a large multilayer printed-circuit board.

computer systems will become blurred in the future as the acquisition of sensory information and its analysis become more closely integrated functions. We can expect that automation of homes, laboratories, and especially factories, driven by economic forces, will accelerate this trend. The myriad electrical and logical interfaces of these devices introduce much complexity in their connection to computer systems. Trends toward standardization, particularly on local area networks, offer promise of relief, but it will take another 5 years at least for the situation to stabilize.

The interfaces between people and computers today tend to force people to adapt to machine idiosyncrasies. A goal we should work toward is allowing individual users to customize the interface to their own styles of use.

Trends in Computer Architecture and Organization

Improvements in hardware technology make the distinction between microcomputers, minicomputers, and maxicomputers no longer obvious. Today's minicomputers exceed in cost-performance characteristics and in range of application many of the large computers of the previous generations; neither size nor price nor application is sufficient to characterize a computer.

The British biologist J. B. S. Haldane observed that nature is not capricious about either the size or complexity of creatures (*6*). There is a proper size for every type of animal, and scaling laws dictate that a large change in size is accompanied by a change in form. Consider, for example, a microscopic worm, absorbing oxygen by diffusion through a smooth skin and extracting nutrients from food by osmosis through a straight gullet. If each of its dimensions were increased tenfold without a change in shape, the worm would require 1000 times as much food and oxygen; however, the necessary surface areas have increased only a hundredfold, and so a change in form to increase the surface-to-volume ratio would be needed. The oxygen- and food-absorbing capacities of larger creatures are increased by the addition of special structures such as lungs or gills and coiled, rough-textured intestines. It may be concluded that in general higher animals are not larger than lower animals because they are more complex, but are more complex because they are larger.

Similarly, in the comparative anatomy of computers, physical size (often directly related to price) is usually the determinant of architectural complexity. Expensive resources must be kept busy, and so a large machine usually has special appendages to keep the rate of flow of information into and out of the machine commensurate with its ability to process the information. Independent, asynchronous input-output processors, arithmetic units that can perform multiple instructions in parallel, and multilevel storage hierarchies are examples of architectural features that minimize the idle time of the central processor. Smaller machines, such as microprocessors, which typically perform single functions serially, can have far simpler control structures and need not be as optimized with regard to idle cycles. As personal computing becomes more economically feasible, the question of the proper structure of these simpler machines and their interconnection to other, perhaps more complex, machines becomes important. We must seriously question the current trend to evolve microprocessors to forms like minicomputers; they are intrinsically different, and perhaps should remain simpler. Nature's solutions often introduce complexity, but only so that function can follow form. Man often complicates his complex creations arbitrarily.

Just as computer technologists have improved performance through miniaturization, computer architects concerned with high performance have pursued concurrency to achieve speeds beyond those occurring naturally from advances in the technology. This parallelism has, in some instances, taken the form of separate, tightly coupled machines, and in others data and control paths have been arranged so that a significant amount of computation can proceed simultaneously in the same machine (*7*). The most interesting of these approaches is the so-called data flow architecture, although problems related to programming have delayed commercial introduction. The trend to enhancing performance by off-loading tasks to specially designed machines (for instance, an array processor for high-speed numerical computation, or a sorting engine) can be expected to continue. More problematic is the interconnection of many identical small machines for high-performance general-purpose computing. This seems to substitute a hardware problem (bus contention) for a software one (resource allocation overhead) and introduces the additional problem of partitioning tasks and integrating results. It is, however, an effective solution to problems with a high degree of natural concurrency, such as many areas of signal processing.

The trend to multilevel memory hierarchies will continue, as storage bandwidth remains a key limitation to high performance. Additional levels in the memory hierarchy may emerge, as may storage subsystems with local processing capability for the logical and physical management of information sharing, backup, recovery in the event of failure, staging of information across levels of the hierarchy, and handling of protocols for multiple-processor configurations. The trend away from the uniform handling of instructions and data (the von Neumann model) will lead to increased reliance on sophisticated language processors.

At least two emerging architectural styles will compete for the low- and intermediate-performance market in the next decade: machines that use the hardware to raise the level of abstraction at which the computational entities are viewed, and so-called reduced instruction set computers (RISC), which trade hardware for software complexity. The former, as exemplified in such recent systems as the Intel iAPX 432 and IBM System/38 (*8, 9*), are claimed to help application development productivity by putting some of the burden usually associated with high-level language compilers into the hardware. Cost and performance penalties are still to be evaluated, as are the productivity gains. RISC machines are designed to exploit the observation that only a small percentage of the simplest instructions account for most of the instructions executed (*10*). If these primitive instructions are carefully chosen to be the target of an optimizing compiler, then the complex instructions can be effectively composed from the basic set. This results in a machine that is, in effect, its own microcomputer, with cost advantages due to the simplicity of the controls and performance advantages from rapid hardwired execution of the primitives. However, programming and debugging such machines, particularly in a network environment, has not yet been commercially tested. Which architectural style prevails is likely to be determined by which can provide the more trustworthy software.

The commercial success of Tandem Computers' Non-Stop system (Fig. 2) (*11*) has demonstrated convincingly that the computer-using public considers high system availability as one of the major requirements for expansion of applications and is willing to pay for the redundancy to achieve it. Tandem's dual-bus design permits graceful degradation in the event of processor and storage system failure. It seems probable that this

and other types of fault-tolerant architectures will proliferate, with manufacturers seeking to provide the necessary redundancy or reliability with minimal cost and complexity burden. The widespread appearance of computer networks will emphasize this requirement.

The greatest problem facing computer manufacturers and users alike is that of graceful migration from current systems to newer architectures and organizations without sacrificing the huge investments in application code. The trend to restructured systems that retain compatibility with old programs will continue to constrain product offerings, although the decreasing cost of emulators and coprocessors to run the old code should help.

Trends in Software

Despite the brave projections of computer scientists in the 1960's and 1970's, the productivity of programmers has increased slowly in the last 20 years. In the early days of computing, the ratio of hardware costs to software costs was about four to one; today it is probably closer to one to four or more (*1*). The end user as application developer is a rarity, and systems programmers, once projected to become obsolescent, are still much in evidence. Many design engineers are now microcoders instead, and it is not at all clear that system reliability has profited from the transformation.

One reason that progress in software productivity has been slow is that programming has not evolved beyond being a craft (that is, an industry in which things are done as they have been done before rather than as some theory would indicate). Programming is not based on a science, and programs are not bound by natural laws in their manipulation of data. The programmer may cause practically any arbitrary sequence of logical and arithmetic manipulations to be performed on the data, as long as the results remain within the range of values acceptable to the hardware. This is not to imply that programming has no intellectual or creative content, but it is important to distinguish an art from a science.

Unlike engineering, programming as a technology is characterized by the limited reusability of solutions. Integration of independently created data structures is seldom possible; most programs are not designed with the idea of maintenance and modifiability in mind, and the evolution of large systems is often chaotic. Much has been written about structured programming and the imposition of engineering-like discipline on the programming process, but the evidence to date remains, to many, unconvincing. There are only limited data relating to software management and production; we must invest in the technology to do controlled experiments to quantify the software development process if these approaches are to be fruitful.

The computer is probably man's intellectually richest invention, making programming the most complex craft ever practiced. This complexity is the key barrier to the realization of the promise of universal computation. For example, after 20 years of struggling with this problem, systematic design trade-offs between the complicating of code and its eventual maintainability are not possible. We are just learning to model programming errors. We strive for robust programs able to survive flaws in much the way that a compartmentalized ship can survive a puncture in its hull, but we do not have a model for redundancy in software.

Unlike the situation of the early days of computing, when the application programmer's interface with the machine was close to the raw hardware itself, today's programmer must deal with implementations of interfaces to an extraordinarily complex system control program (the collection of utility programs and services that allocates the resources of the machine). A system control program is usually created by a large number of programmers over a long period. It is typically an unstable mélange, and it is not surprising that perhaps two-thirds of the programmers employed today are involved in the maintenance and modification of existing code instead of the creation of new applications. An industry trend today (which should lead to continual improvement) is the hardening of the system control program interfaces by individual manufacturers, since the cost/benefit ratio of developing new, proprietary system control programs is not attractive. De facto industry standard operating systems may emerge that are flexible enough to add function in a systematic and modular way (the rapid emergence of UNIX, a single user operating system, is cause for hope in this direction).

There are other promising directions. Very high level languages embedded in an operating environment that frees the applications developer or end user from concern with the details of program construction have had several commercial realizations. A recent example of such a system is VISICALC (*12*), in which a

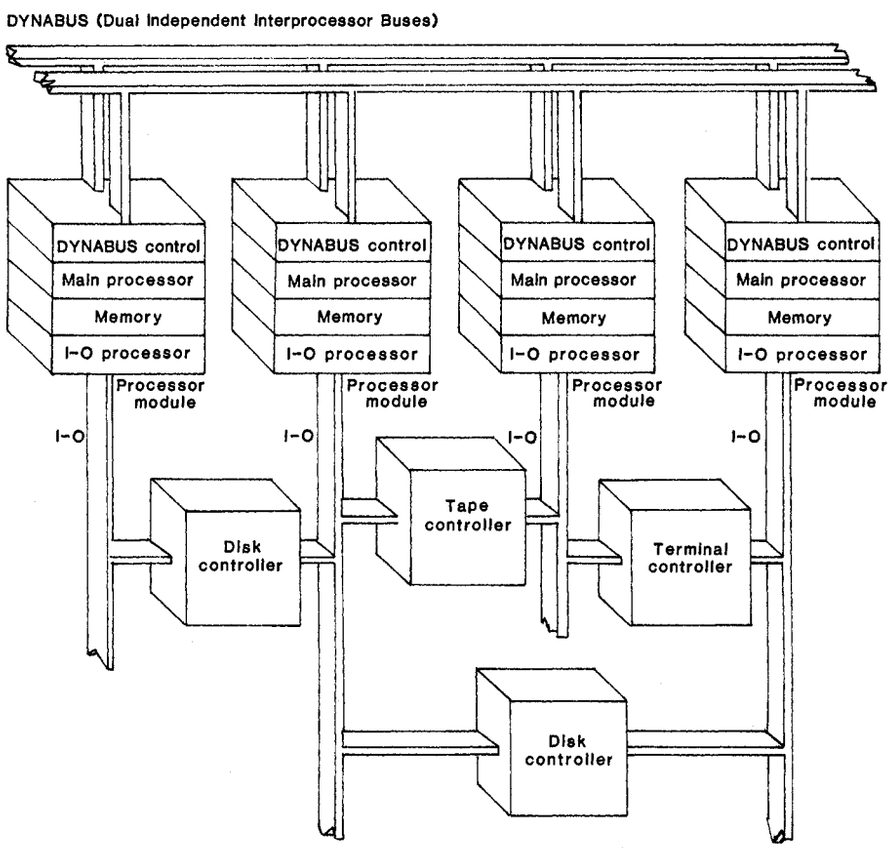

Fig. 2. Block diagram of the Tandem Non-Stop computer system showing the redundant and modular nature of the system's organization.

graphic representation of a work sheet can be tailored by the user for his own particular needs, and the detailed operation of the program in order to satisfy the constraints imposed by the problem is not the user's concern. The user can ask "what if" questions interactively simply by changing an entry; the effect of that change then propagates across the system automatically.

QUERY-BY-EXAMPLE is a data management system that permits sophisticated queries and database operations to be performed simply by filling in requests in tables in a way that mimics a person's manual pencil and paper solution (much in the manner of filling in the pieces of a puzzle in arbitrary order). An important advantage is that the user can structure requests not foreseen by the programmers (*13*). This may be the harbinger of nonprocedural end user systems that are natural to use and that can be learned quickly and incrementally.

Basic changes in the nature of programming are necessary to make a real breakthrough in software development productivity. Winograd (*14*) suggested that the complexity of contemporary tasks requires higher level programming systems that concentrate on the properties of the packages and objects from which application programs are built, and not on the detailed specification of algorithms. The solution to raising the level of abstraction with which we view complex systems will ultimately lie not in greater discipline but in more appropriate tools.

Trends in Communications

For many, the most exciting new applications of computers result from the coalescence of computing and communications technologies. The situation today can be likened to the early history of the automobile: an effective hierarchy of paved roads and lower costs due to mass production were both necessary to establish the automobile as an effective personal and commercial vehicle for nonlocal applications. Many companies are today vying to create networks of communications highways for data and voice and, perhaps, image and video information as well. Unfortunately, this is a complicated and highly competitive business fraught with multiple "standards." The resulting complexity is threatening to delay significantly the now commonplace "X of the future" scenarios (where X is home, office, school, factory, or some combination of these).

All of these technologically plausible visions feature the ability to send and receive "documents" that can be mixtures of several media and that draw on global information repositories. Often suggested benefits include substitution of electronic information flow for human transportation and of home offices for skyscrapers.

One limitation, again, is software complexity, made even more difficult by a need for cooperation among competitive agents. The information culled from various sources is stored in different formats, addressed by different access methods, and transmitted according to different protocols. Needed are a uniform mechanism for self-description and a message-based architecture to minimize the overhead required today at each stage of the process. The albatross of compatibility with the past is present here as well.

From a communications viewpoint, computers are simply devices for storage, switching, and mathematical transformation of the information being transported by the network. We must question some fundamental parameters: Just how much bandwidth is required? (That is, what is the rate at which humans can assimilate information?) What degree of connectivity is required? We must be careful to distinguish entertainment, largely a one-way high-bandwidth broadcast phenomenon, from transmission of speech, text, data, and images for human decision-making activities. The telephone network already in place, augmented with fiber optics, microwave, and satellite technology for high traffic links, may be sufficient for much of the latter task. The store-and-forward capability of such networks can free us from the tyranny of time in much the way that the telephone helps us to conquer space. Today's emerging voice mail systems are a new form of communication (consider the deliberately delayed delivery of bad news, or listen-once-and-destroy modes), as are network-based conferences that do not take place in real time. Our intuition will be a poor guide for design and use of these systems, particularly as they evolve to incorporate multiple-media documents.

Local and global networks will promote the design of machines for highly optimized shared functions, such as the recognition of speech, the encryption and decryption of information, and especially the efficient storage and retrieval of information. The latter is a difficult and topical problem (*15*), for if we can produce reliable, cost-effective, and secure distributed storage management, we may achieve economies of scale for personal work stations and be able to off-load the complexities of database programming to a considerable extent.

Trends in Artificial Intelligence

Nothing has so raised the hopes and fears of people as the notion of reasoning machines able to make judgments based on preprogrammed knowledge. In fact, the accomplishments of artificial intelligence (AI), as this field is loosely called, are just beginning to see practical application, long after overly optimistic researchers had predicted. A central idea of early work in AI was the substitution of a process capable of adaptive learning for another too complex or too little understood to be represented analytically. Most workers in the field have now abandoned this approach in favor of systems that can represent the sources and structure of knowledge and bring expert heuristic information to bear on problems in an effective way (*16*).

A decade ago few would have believed that there would be a $100 machine that plays chess better than 90 to 95 percent of the population, or a set of programs that interprets electrocardiograms better than most doctors (and even better than some cardiologists) (*17*), or a program that successfully prospects for minerals (*18*).

List-processing systems, so effective for manipulating symbolic information, are now a commercial reality and may offer significant advantages for tasks like algebraic problem-solving and interactive computer-aided design. Artificial intelligence is beginning to spawn technology, and many think the application of these techniques will blossom in the next 10 years. The technology seems most likely to affect four areas: application programming, recreation, education, and industrial automation.

The tool kit of the applications programmer may be augmented in the areas of interfaces between humans and machines and the ability to supplement algorithmic problem-solving with the commonsense knowledge of an expert in the field. In the former area, limited but useful forms of natural language understanding should permit reasonable dialogues in a restricted domain of discourse. Adaptive techniques will model modes of user interaction, tailoring default options and command sequences to individual user styles. The ability to capture expert information and apply it systematically to diagnostic problem-solving may produce a generation of "smart"

instruments that present less data and more information to users.

The trend toward intelligent toys like chess machines can be expected to escalate dramatically. As the new generation of computer-literate children grows up during the 1980's, AI-based games like "Adventure" will proliferate on inexpensive personal computers. These games already incorporate limited natural language understanding, commonsense knowledge bases, and nontrivial simulations of human reasoning. Such toys may evolve with a new form of computer-aided education where the concept of an educational assistant is substituted for that of a tutor used in the earlier, not very successful attempts at computer-aided instruction. For instance, these assistants will not teach spelling or algebra but will deliver a resource to the student, just as calculators assist with arithmetic today. The "knowledge" calculator would be the entry to large libraries during and after professional training and could be a standard item included in college tuition.

The escalating trend toward factory automation, driven by economic pressures, can be expected to incorporate many of the developments of AI. Simple pick-and-place industrial robots of the sort now used for painting, welding, and simple assembly are being replaced by computer-based manipulators with sensors and effectors that can perform much more sophisticated tasks. Real-world modeling, computer interpretation of sensory data, and higher level procedure and parts specification will all be needed if machines are to deal with incorrect parts, improper sequences of assembly operations, and cost-effective retooling. Networks will again be important, as the computer-aided design and manufacturing databases are created and integrated with production management and office information systems.

Conclusion

Limitations to further progress in computing and communications are not fundamentally related to physical constraints. Hardware and software technologies have now advanced to the stage where complications introduced by design assumptions of the last 30 years are the principal barrier. Many of these issues will be reconsidered from first principles in the next decade. It will be an exciting time.

References and Notes

1. T. A. Dolotta, M. I. Bernstein, R. S. Dickson, Jr., N. A. France, B. A. Rosenblatt, D. M. Smith, T. B. Steel, Jr., *Data Processing in 1980–1985: A Study of Potential Limitations to Progress* (Wiley-Interscience, New York, 1976), p. 87.
2. R. N. Noyce, in *The Computer Age: A Twenty-Year View*, M. L. Dertonzos and J. Moses, Eds. (MIT Press, Cambridge, Mass., 1979), pp. 322–337.
3. For a full review of the status of this effort, see *IBM J. Res. Dev.* **24** (No. 2) (1980), special issue.
4. G. G. Werbizsky, P. Winkler, F. W. Haining, *Electronics* **52** (No. 2), 109 (2 August 1979).
5. *Eighteenth Design Automation Conference*, ACM SIGDA and IEEE Computer Society (Institute of Electrical and Electronic Engineers, New York, 1981).
6. J. B. S. Haldane, in *The World of Mathematics*, J. R. Newman, Ed. (Simon & Schuster, New York, 1956), p. 952; D. W. Thompson, in *ibid.*, p. 1001.
7. For several overviews of a number of parallel machine organizations, associative processor architectures, pipelines, and multiprocessing and similar techniques, see *Comput. Surv.* **9** (No. 1) (1977).
8. *Introduction to the iAPX 432 Architecture* (Manual 171821-001, Intel Corporation, Santa Clara, Calif. 1981).
9. M. E. Houdek, F. G. Soltis, R. L. Hoffman, in *Eighth Annual Symposium on Computer Architecture* (Institute of Electrical and Electronic Engineers, New York, 1981), pp. 341–348.
10. D. A. Patterson and C. H. Sequin, in *ibid.*, p. 443; D. A. Patterson and D. R. Ditzel, *Comput. Archit. News* **8** (No. 6), 25 (1980).
11. For descriptions, see *Mod. Data* **9** (No. 2), 57 (1976); "Multiprocessor system claims to be fail-safe," *Electron. Des.* **24** (No. 1), 21 (1976); *Comput. World* **10**, 51 (1975).
12. Trademark of Software Arts, Inc., Arlington, Mass.
13. M. M. Zloof, *IBM Syst. J.* **16**, 324 (1977).
14. T. Winograd, *Commun. ACM* **22**, 391 (1979).
15. For recent surveys, see W. H. Kohler, *Comput. Surv.* **13** (No. 3), 149 (1981); P. A. Bernstein and N. Goodman, *ibid.*, p. 185.
16. *Fifth International Joint Conference on Artificial Intelligence* (Massachusetts Institute of Technology, Cambridge, 1977).
17. P. M. Rautaharju *et al.*, in *Tenth Bethesda Conference, Optimal Electrocardiography*, J. A. Abildskov and L. S. Dreifus, Eds. (Department of Health, Education, and Welfare, Hyattsville, Md., 1977), pp. 48–60; *Am. J. Cardiol.* **41**, 158 (1978).
18. R. O. Duda, J. Geschnig, P. E. Hart, in *Expert Systems in the Micro-Electronic Age*, D. Michie, Ed. (Edinburgh Univ. Press, Edinburgh, 1977), pp. 153–167.