computers and people

formerly *Computers and Automation*

March — April, 1987

Vol. 36, Nos. 3-4

LEAVES & BIRDS & SUN DESIGN *by Virginia Hines*

The Computer Almanac and Computer Book of Lists — Instalment 52

Neil Macdonald, Assistant Editor

36 TOPICS IN A COURSE "ARTIFICIAL INTELLIGENCE OVERVIEW" (List 870301)

1. Introduction
   - Definition of Artificial Intelligence (AI)
   - History
   - Current Activity
   - The Basic Paradigm
   - Commercial Possibilities
   - The Japanese Fifth Generation Project
   - American National Responses
2. Who's Who and Where's Where?
   - People
   - Universities
   - Countries
   - Companies
3. Expert Systems
   - What They Can Do
   - Some Representative Systems
4. Knowledge Engineering
   - Knowledge Acquisition
   - The Programming
   - Knowledge Representation
   - How a system can explain its reasoning
   - How long to build a system?
   - Choosing the right problem
5. Natural Language
   - History of language work
   - ELIZA and keyword matching
   - Approaches via syntax (grammar)
   - Approaches via semantics (meaning)
   - Front ends to databases
6. Vision and Robots
7. Machine Learning
8. Technical Basis for an AI Project
   - Staff
   - Hardware
   - Languages
   - Tools for making expert systems
   - Languages and tools for the IBM personal computer
9. Issues in Commercializing AI
   - Technical
   - Managerial
   - Commercial
   - Social

(Source: Catalog of Professional Development Seminars, Feb.-July, 1987, issued by Control Data Corp., Inst. for Advanced Technology, 1450 Energy Park Dr., St. Paul, MN 55108)

34 TOPICS IN A COURSE "KNOWLEDGE ENGINEERING FOR EXPERT SYSTEMS (List 870302)

1. Introduction
   - What is artificial intelligence?
   - What are expert systems?
   - Examples
   - What is knowledge engineering?
   - The role of the knowledge engineer
   - Exploratory programming
2. Development of Expert Systems
   - Problem identification
   - Conceptualization
   - Building a prototype
   - Checking it
   - Debugging it
3. Knowledge Acquisition
   - Nature of the problem
   - Techniques for organizing the solution process
   - Sources of knowledge
   - Interviewing
   - Eliciting information
   - Recording information
   - Developing artificial tasks
   - Protocol analysis
   - Automated knowledge acquisition
4. Conceptual Modeling
   - Models of human knowledge
   - Models of task behavior
   - Learning of concepts
   - Using the models to guide knowledge engineering
5. Rule-based Programming
   - Writing the first rules
   - Using hierarchy
   - Picturing your rule base
   - Going wrong with rules
6. Advanced Knowledge Programming
   - Knowledge representation
   - Inference

(please turn to page 26)

Editorial

Wisdom and Artificial Intelligence

Edmund C. Berkeley, Editor

What is wisdom?

One of the best short definitions of wisdom is expressed in a remark by the author W. Somerset Maugham in his novel "Catalina". The character that possesses the wisdom is not Catalina but the Prioress, who was

> "impatient to think that for such a trifling reason Catalina should be willing to forego the great advantages which the religious life offered her. But a wise person takes things as they are and, knowing the conditions, proceeds to deal with them in such a manner as to achieve the desired result."

It is difficult to give a better short definition of wisdom than Maugham's last sentence.

What is artificial intelligence?

The term "artificial intelligence" has been used often since the 1950s to designate behavior by a machine (or a computer) which would be considered intelligent if it were behavior by a human being. The term has been given many more definitions by many computer scientists. In fact the meaning changes over the years, because computer programs that express intelligent behavior become more and more intelligent, and resemble more and more the intelligent behavior of humans.

This fact emphasizes the fascination of intelligence exercised and shown by a machine (or a computer), and also the main direction of development in the computer field: more and more intelligent behavior.

What are the tasks that artificial intelligence now accomplishes usefully?

Among these are:

- the representation of expertness (as in developing expert systems)
- inference (as in drawing conclusions from large amounts of data)
- understanding (as in deducing answers from files of data)
- game playing (as in chess and checkers)
- translating natural language (from one language to another)
- proving theorems (as from one set of statements to another)

Some more difficult and important tasks are often included.

Many other kinds of difficult and important tasks used to be included in the list of applications of artificial intelligence, but are no longer usually listed, because much of the perplexity about them has gone. It is reasonable to expect that, as years pass, we shall gain more knowledge and experience with programming perplexing tasks, and artificial intelligence will gradually disappear.

So the programming of wisdom in general can become sensible, and perhaps easy?

Yes. In fact, the answer to this question is already clear. For example, there is no doubt at all that expert systems in certain tasks (such as a computer playing chess with the ability of a grandmaster) can be programmed. In 1987 already the playing of chess by a computer is better than the playing of 95% of the members of chess clubs.

Of course, if a team of clever programmers cannot devise an expert computer system, the construction of that system may be delayed for years. But, if the appropriate wisdom can be deduced or inferred, and if the appropriate observations of the real world can be made, and if the programmers do not make mistakes, then the programming of wisdom can be accomplished.

Ω

Vol. 36, Nos. 3-4
March-April, 1987

computers and people

formerly *Computers and Automation*

"Computers and People" (ISSN 0361-1442), formerly "Computers and Automation," is published every two months at 815 Washington St., Newtonville, MA 02160 U.S.A., by Berkeley Enterprises, Inc. Printed in U.S.A. Second-class postage paid at Boston, MA and additional mailing points.

Subscription rates, effective Sept. 1, 1986: U.S.A., $22.00 for one year, $43.00 for two years; elsewhere, add $7.00 per year.

NOTE: The above rates do not include our publication, the "Computer Directory and Buyers' Guide." To receive this, please add $20.00 per year to your subscription rate in the U.S.A., and $23.00 per year elsewhere.

NOTE: No organization in Switzerland or Morocco is authorized or permitted by us to solicit for, or receive payment for, "Computers and People" or the "Computer Directory and Buyers' Guide." All subscriptions to and payments for these publications should be sent directly to Berkeley Enterprises, Inc.

Please address all mail to: Berkeley Enterprises, Inc., 815 Washington St., Newtonville, MA 02160, U.S.A.

Postmaster: Please send all forms 3579 to Berkeley Enterprises, Inc., 815 Washington St., Newtonville, MA 02160, U.S.A.

Change of address: If your address changes, please send us both your new address and your old address (as it appears on the magazine address imprint), and allow four weeks for the change to be made.

The magazine of the design, applications, and implications of information processing systems – and the pursuit of truth in input, output, and processing, for the benefit of people.

Announcement

"The Computer Directory and Buyers' Guide"

The names, addresses and descriptions of over 3500 computer field organizations are still being updated in our computer data base for the next Directory edition. Production of the photooffset master for printing has again been unfortunately delayed. We hope that we will have this, the 28th edition, ready for mailing to subscribers early in 1987.

Meanwhile, any current subscriber to "Computers and People" who also subscribes to the "Computer Directory and Buyers' Guide," and who does not already have the 1984-85 edition, may on request to us, receive a copy of that issue, so long as the overrun lasts.

Front Cover Picture

The front cover shows a sample of art by Virginia Hines, a student in a computer-assisted art class at Calif. State Univ. - Chico, Chico, CA. This is a black and white illustration of an original color work made by using a group of programming routines. The system used was an HP3000 mini-computer connected to a Tektronix display terminal, with a choice of CRT photography or a colored printer for output.

Computer Field → Zero

There will be zero computer field and zero people if the nuclear holocaust and nuclear winter occur. Every city in the United States and the Soviet Union is a multiply computerized target. Radiation, firestorms, soot, darkness, freezing, starvation, megadeaths, lie ahead.

Thought, discussion, and action to prevent this earth-transforming disaster is imperative. Learning to live together is the biggest variable for a computer field future.

Signals in Table of Contents

[A]	–	Article
[C]	–	Monthly Column
[E]	–	Editorial
[EN]	–	Editorial Note
[O]	–	Opinion
[FC]	–	Front Cover
[N]	–	Newsletter
[R]	–	Reference

Type of Subscription

*D ON YOUR ADDRESS IMPRINT MEANS THAT YOUR SUBSCRIPTION INCLUDES THE COMPUTER DIRECTORY AND BUYERS' GUIDE. *N MEANS THAT YOUR PRESENT SUBSCRIPTION DOES NOT INCLUDE THE COMPUTER DIRECTORY.

Safety in Computer Programs

Prof. C.A.R. Hoare
Programming Research Computation
Oxford University
Oxford, England

"You can establish the correctness of computer programs by the normal mathematical techniques of modeling, calculation, and proof ... but this method could never be effective in practice unless it is accompanied by appropriate attitudes and managerial techniques."

Programs That Control Computers

Digital computers must be the most reliable mechanisms built by the human race. Millions of computers throughout the world, and thousands in space, execute billions of instructions per second for billions of seconds without a single error in any of the millions of bits that comprise each computer. Yet few of us would trust our lives to a computer.

The fault lies not in the computer's hardware but in the programs which control it. Programs faithfully reproduce the errors, oversights, inadequacies and misunderstandings of the programmers who compose them. There are some large and widely used programs, operating systems and compilers in which hundreds of new errors are discovered each year. Even when programmers correct errors, the rate at which users continue to discover new ones remains constant over several decades. Indeed, some suspect that each correction introduces more than one new error. And only a few of the errors in these programs will ever be discovered before the programs are superseded by new products. These new products are, of course, equally unreliable.

Subtle Errors

Most of the errors that are found in general computer programs are extremely subtle: their effects are not serious, and it is easy to avoid them until the software's supplier gets round to correcting them. But computers are beginning to play an increasing role in "life-critical applications", situations where the correction of errors on discovery is not an acceptable option -- for example, in control of industrial processes, nuclear reactors, weapons systems, oil rigs, aero engines and railway signalling. The engineers in charge of such projects are naturally worried about the correctness of the programs performing these tasks, and they have suggested several expedients for tackling the problem. Let me give some examples of four proposed methods.

Examples of Corrections

The first method is the simplest. I illustrate it with a story. When Brunel's ship the SS Great Britain was launched into the River Thames, it made such a splash that several spectators on the opposite bank were drowned. Nowadays, engineers reduce the force of entry into the water by rope tethers which are designed to break at carefully calculated intervals.

When the first computer came into operation in the Mathematish Centrum in Amsterdam, one of the first tasks was to calculate the appropriate intervals and breaking strains of these tethers. In order to ensure the correctness of the program which did the calculations, the programmers were invited to watch the launching from the first row of the ceremonial viewing stand set up on the opposite bank. They accepted and they survived.

A similar solution has been proposed for programs that control the propeller and steering of a ship which has to keep station in rough seas close to the leg of an oil drilling rig. The action of the wind and waves is so sudden that no human helmsman could avoid collision, and the task must be delegated to a computer program. But if we require the programmer to demonstrate the reliability of his program by joining the crew of the ship, a question arises when he resigns his highly paid post. Is this because of boredom, seasickness or fear of something worse?

Inspection of Programs

When the early American satellites were first controlled by on-board computers, outside contractors wrote the programs. On delivery, checkers visually inspected the absolute binary code of the programs: rows and rows of raw binary digits. They could not use any higher level programming language, since assemblers and compilers are large programs and, therefore, even less trustworthy than the programs that they compile.

To assist in the "eyeballing", NASA constructed a massive suite of programs -- for example, to reconstruct the assembly code from the delivered binary code, to draw flowcharts and to analyse all the control paths. The human checkers then annotated these charts with assertions about scaling factors of the arithmetic operations; with the help of machines, they then checked that the scaling factor would not vary each time that the program went round a loop.

Checking Binary Code Against Nothing

The fundamental flaw in this approach is that, when checking something, you should always check it against something else which you either know is reliable, or which someone has similarly checked. To check the binary code against absolutely nothing except itself is a fearsome task, and requires inspired guesswork in order to reconstruct the documentation, designs and specifications. No wonder checking is even more expensive than the original programming, which progresses in the more natural direction, from abstract to concrete, from specification through design, to the voluminous detail of the code.

This kind of machine-assisted analysis is still very popular in the checking of safety-critical software. The quoted reason is far from reassuring: many of the programs are written without any specification at all; so the only thing there to check is the ultimate code. The basic mistake is that the checking is done far too late: it is a fundamental principle of quality control that what you should check is not the product but the methods by which it is produced. It is only by improving methods that it becomes possible to achieve reliability.

Checking in a Simulated Environment

We can often test programs that control critical processes by running them initially in a simulated environment -- for example, inside of a fast mainframe computer. Suppose that the simulation runs many times faster than real time. Thus in one year it may be possible to simulate say a thousand years' operation of the process and check all the answers that the program gives. If only a few errors are detected, it is then unlikely that any such error will occur within the first 10 years of live running -- which may be the length of the program's useful lifetime. This appealing method suffers from several devastating drawbacks. The first and least of them is that the delay before a working program is installed is usually unacceptable.

The second flaw is philosophical: it is morally very difficult to risk people's lives on a program that has known bugs. Yet to "correct" the known bugs would not only be wholly ineffective; it could be disastrous since it could introduce a completely unknown batch of new bugs. To counter this possibility, all the testing would have to start again from the beginning. The third is a practical flaw. What happens if 10 errors are detected in the thousand-year test? This result gives a quite unacceptable risk that an error will occur in the 10 years of actual use. The only remedy is to rewrite the whole program and start the test again. By that time, the project will have lost its value and relevance.

The fourth is a logical flaw: the method depends on the correctness of the simulated environment and on that of the checking program. Yet if the checking program is correct, why not use it as part or whole of the program which controls the real process?

The fifth drawback fortunately makes the previous four irrelevant: it is only in the very simple and increasingly rare applications that it is possible to run a simulation, even on the fastest supercomputers, at a rate faster than real time. So this method is applicable only to programs with a design life measured in minutes or hours; it is therefore not appropriate for most civilian applications.

Voting

In many life-critical applications, the problem of the reliability of hardware requires there to be three or more identical computers, with a voting circuit at their output which ensures that every action has the agreement of at least two of them. The likelihood that two or more computers would go wrong simultaneously is very much smaller than the risk that just one would. Since the hardware is available, it is possible to apply the same technique to software. You get three or more independent teams of programmers to write three or more independent programs, and load a different one into each computer. If one of the programs goes wrong occasionally, the other programs, which are unlikely to go wrong on the same occasion, outvote it.

In hardware, such a method will deal with transient errors, such as might arise from an occasional cosmic ray impinging on the silicon chip. It does not deal with persistent faults, which must be cured by manual (or automatic) replacement of components. Unfortunately, in software there is no reason to suppose that errors are transient; a single erroneous subscript can cause the program to be overwritten, so that it never works again. To guard against this possibility, it is necessary to design the hardware to clear the store and reload the program between each cycle of operation. So this technique applies only to programs whose operation is a series of independent cycles, with no long-term storage. Such a program cannot accumulate past readings, to integrate or to smooth them. Thus the technique is applicable only to simple control processes.

A second weakness in this method is that there is no reason to suppose that errors in programs produced by independent programmers will be independent. Quite the reverse. Programmers are often educated in the same "culture"; they find the same things difficult, and they are subject to the same kinds of misunderstandings and oversights -- for example, forgetting to test for an extreme case, or omitting to provide for zero iterations of a loop.

Hardware Involved With Software

But there is one new circumstance that will make diversity impractical on large systems. In real-time applications, the response of a computer depends on details of the timing of the signals which it responds to -- for example, the arrival of an interrupt. Thus two correct programs which receive signals at slightly different times can give different results, both of them correct. Unfortunately, a simple voting circuit in the hardware cannot know this, and it will invoke unnecessary alarms. In spite of vigorous efforts to prevent it, this is what actually occurred on the first attempted launch of the American space shuttle. The Columbia was a victim of the first highly spectacular public failure of a computer program. The cause of the failure was the very technique designed to ensure reliability.

There is more insidious danger in using diversity for long-running programs. When a software error occurs, the hardware-comparison circuit will signal an alarm; but the operators will attribute this alert to a software error, and will ignore it. As a result, when it really is the hardware that goes wrong, it will be impossible for the operators to distinguish it from an error in the software, and they will continue to ignore it. Thus intermittent and faulty hardware will not be replaced in good time. This infection of hardware by the unreliability of the software has actually been observed on a computer for which the ALGOL (ALGOrithmic Language, an arithmetical language by which numerical procedures may be precisely presented to a computer in a standard form) compiler used the hardware parity violation circuits to detect the programming error of an uninitialized variable. In a short while, as a result of lack of maintenance, the hardware of the main store of every computer which used that compiler became unreliable.

Mathematics Holds a Solution

These are the four methods that people have proposed and used in practice for the construction of safety-critical software. As far as I know, they have been almost completely successful so far, and no large-scale loss of human life has ever been attributed to a programming error. Nevertheless, each method suffers from several drawbacks that may become more serious with the imminent increase in the scale and sophistication of systems whose safety is monitored by computer programs. I therefore suggest that we should explore an additional method, which promises to increase the reliability of programs. The same method has assisted the reliability of designs in other branches of engineering, namely the use of mathematics to calculate the parameters and check the soundness of a design before passing it for construction and installation.

Alan Turing first made this suggestion some 40 years ago; it was put into practice. on occasion, by the other great pioneer of computing, John von Neumann. Shigeru Igarashi and Bob Floyd revived the idea some 20 years ago, providing the groundwork for a wide and deep research movement aimed at developing the relevant mathematical techniques. Wirth, Dijkstra, Jones, Gries and many others (including me) have made significant contributions. Yet, as far as I know, no one has ever checked a single safety-critical program using the available mathematical methods. What is more, I have met several programmers and managers at various levels of a safety-critical project who have never even heard of the possibility that you can establish the total correctness of computer programs by the normal mathematical techniques of modelling, calculation and proof.

Such total ignorance would seem wilful, and perhaps it is. People working on safety-critical projects carry a heavy responsibility. If they ever get to hear of a method which might lead to an improvement in reliability, they are obliged to investigate it in depth. This would give them no time to complete their current projects on schedule and within budget. I think that this is the reason why no industry and no profession has ever voluntarily and spontaneously developed or adopted an effective and relevant code of safe practice. Even voluntary codes are established only in the face of some kind of external pressure or threat, arising from public disquiet, fostered by journals and newspapers and taken up by politicians.

A Mathematical Proof That a Program Works

A mathematical proof is, technically, a completely reliable method of ensuring the correctness of programs, but this method could never be effective in practice unless it is accompanied by the appropriate attitudes and managerial techniques. These techniques are in fact based on the same ideas that have been used effectively in the past.

It is not practical or desirable to punish errors in programming by instant death. Nevertheless, programmers must stop regarding error as an inevitable feature of their daily lives. Like surgeons or airline pilots, they must feel a personal commitment to adopt techniques that eliminate error and to feel the appropriate shame and resolution to improve when they fail. In a safety-critical project, every failure should be investigated by an impartial enquiry, with powers to name the programmer responsible, and forbid that person any further employment on safety-critical work. In cases of proven negligence, criminal sanctions should not be ruled out. In other engineering disciplines, these measures have led to marked improvement in personal and professional responsibility, and in public safety. There is no reason why programmers should be granted further immunity.

Mathematical calculations and proofs are in many ways very like programs. They are long and intricate texts in which the slightest blunder leads to invalidity. In principle, a programmer could learn to write completely formal proofs, and a computer could be programmed to check these proofs. This principle has inspired some excellent research and development of proof-checking programs. But the labor of constructing proofs with sufficient formality for a machine to check them has turned out to be excessive.

Checking a Mathematical Proof

For the time being, the most effective method of checking proofs is to submit them to the gaze of another programmer or mathematician. The checker then joins the programmer in taking responsibility for the correctness of the program. The principle that the work of an engineer should be inspected and signed off by another more experienced and competent engineer lies at the heart of the codes of safe practice in all branches of engineering.

The checking of proofs has much in common with the eyeballing of code, but it is in principle more effective, since it is easier to detect a hole in a well-presented proof than it is to find an oversight in a program. This is because a proof checker only needs to check the validity of each line of the proof, comparing it only with one or two previous lines. For a program, the checker has to check each line in the context of every other line of code in the program -- a task which is quite impossible for large programs.

Testing

However carefully an engineer has specified, designed and implemented a product, it would be extremely foolish to put the item into service without subjecting it to thorough tests that are as realistic as possible. Such tests are necessary to check the adequacy of the original specification, and the general assumptions on which it is based. We need these tests in order to check our understanding of the relations between the hardware, the software and their working environment. Evaluation of this kind also checks the adequacy of the methods by which the system was specified, designed and constructed.

The vast majority of all tests should succeed -- otherwise there is something so seriously wrong with the product that it would be best to throw it away. But inevitably there will be an occasional failure even in programs which were thought to be proved correct. On these occasions the proper response is first to find the cause of the error: for example, carelessness, misunderstanding or use of inadequate methods. Second, you must assess whether that same cause might have led to other errors in other parts of the program. All such doubtful areas must be checked again. Only then should the detected error be corrected, and the correction itself undergo more rigorous checks against the possibility that it will introduce further errors. These common-sense principles are standard practices in many branches of engineering, yet their application to programming has been slow to gain recognition.

Machine Language Testing

Most computer programs today are written in high-level programming languages which have to be translated into the language of the machine before they can be executed. The program which does the translation is itself large, complex and subject to error. This has inhibited the use of high-level languages for safety-critical programming. This is a mistake. Some compilers for simple high-level languages have proved reliable in widespread use and the chance that they will make a mistake in compiling a particular safety-critical program is very small. The chance that such a mistake would escape detection in routine testing is far smaller still. In the last resort, a visual check of the machine code against the original high-level program is still a possibility -- easier and safer than writing the original program.

The principle of diversity is fundamental to improving the reliability of programs. The people who check proofs should be independent of those who construct them. Those who design test regimes should be independent of those who design the objects undergoing the tests.

Finally, for ultimate confidence, it would be ideal to have two independent proofs, preferably using different methods of proof. It is by independent experiment that the laws of physics are confirmed; and even mathematicians are happier with fundamental theorems that have been proved more than once.

The principle of diversity when applied to proof will be more effective than when applied to programs, and will probably be no more expensive. It does not suffer from any of the problems or dangers which arise from the use of hardware error checks as a protection against software bugs.

Mathematical Aspects of Programming

Mathematics is a traditionally unpopular subject, among programmers as in the general population. Even programmers with a university degree in pure and applied mathematics have no idea how to apply their mathematical skills to the practice of their profession. Fortunately, the mathematical aspects of programming are now beginning to find their way into university curricula. I look forward to the day when these topics find their way into specialist teaching in secondary schools. My prediction is that the mathematical study of programming will contribute to wider appreciation, understanding and love of mathematics; for mathematics is constantly rejuvenated by discovery of new applications. By that time, the professional use of mathematics in safety-critical programming will ensure that computer programs are the most reliable component of any system in which they are embedded.

The Reliability of Programming

To achieve this desirable goal, I believe it is necessary to raise the alarm publicly about the inadequacy of some of the methods that are currently proposed and in use. But it is important not to exaggerate the risk or to provoke over-reaction. Critical processes controlled by computer programs are probably far more reliable than those controlled by older analogue devices and certainly more reliable than manual methods. This is mainly due to the greatly increased reliability of the hardware of computers. Indeed, at present, the fear of errors in programming is the main reason for delay in the introduction of computerized control in safety-critical applications. This delay itself represents a risk to public safety. The introduction of mathematical methods to prove programs correct will therefore bring double benefit -- it will improve the reliability of existing applications, and it will give people confidence to extend these benefits to new applications.

References

Roland C. Backhouse, "Program Construction and Verification," Prentice Hall (1986). An introductory text for assertional methods of program development and proof. These methods form the basis of a standard of rigor and accuracy for safety-critical programs.

"Software, Vital Key to UK Competitiveness," Report of ACARD, HMSO, 1986, £6.00. This report emphasizes the importance of improved methods in the construction of software for all purposes. It suggests that mathematical methods introduced for safety-critical programs will bring improvements throughout the software industry.

Ω

Opinion

Nuclear Weapons — And the Simplest Truths

Dr. Joseph Weizenbaum
Laboratory for Artificial Intelligence
Mass. Institute of Technology
Cambridge, MA 02139

"The highest duty of intellectuals in these times is to speak the simplest truths in the simplest possible words." *– George Orwell*

Apparent Normality

Whenever I am in West Germany, I am amazed by the apparent normality of everyday life. As only an occasional visitor to Germany I see strange things that must by now appear routine, even natural to Germans. For example, holes in the streets that are intended to be filled with nuclear land mines or the closeness of every German citizen to nuclear weapons storage facilities. I notice, in other words, the Germans' physical, but even more their psychological, proximity to the final catastrophe.

We in America are no more distant from the catastrophe than the Germans. In case of war, regardless of whether unintentionally initiated by technology allegedly designed to avert war, or by so-called statesmen or women who thought it their duty to push the button, Germans may die ten minutes earlier than we in fortress America, but we shall all die.

Holes for Nuclear Land Mines

We have no holes in our streets for nuclear land mines. We see our missile silos only now and then, that is, only whenever it pleases someone to show them to us on television. No matter how passionately our government tries to convince us that the nasty Soviets are effectively as near to us as to the Europeans, that they threaten us from, for example, Cuba or Nicaragua, Americans are, on the whole, unconvinced and therefore untroubled by such efforts. So it isn't surprising that the average American worries so little about the danger that confronts us. In fact, it would be astounding if he were even particularly aware of it. The American experience of war allows an "it can't happen here" attitude to grow, rather than a concrete fear of what appears to be far removed from the immediate concerns of daily life.

I am aware that it is emotionally impossible for people to live for very long in the face of immediate threats to their very existence without bringing to bear psychological mechanisms that serve to exclude those dangers from their consciousness. But when repression necessitates systematically misdirected efforts or excludes potentially life-saving behavior, then it is time to replace it by a deep look into the threat itself.

This time has come for computer professionals. We now have the power to alter the state of the world fundamentally and in a way conducive to life.

Computer Professionals Cooperate for Nuclear Land Mines

It is a prosaic truth that none of the weapon systems which today threaten murder on a genocidal scale, and whose design, manufacture and sale condemns countless people, especially children, to poverty and starvation -- that none of these devices could be developed without the earnest, even enthusiastic, cooperation of computer professionals. It cannot go on without us! Without us the arms race, especially the qualitative arms race, could not advance another step.

Does this plain, simple and obvious fact say anything to us as computer professionals? I think so:

First, those among us who, perhaps without being aware of it, exercise our talents in the service of death rather than that of life have little right to curse politicians, statesmen and women for not bringing us peace. Without our devoted help they could no longer endanger the peoples of our earth. All of us must therefore consider whether our daily work contributes to the insanity of further armament or to genuine possibilities for peace.

In this context, artificial intelligence (AI) comes expecially to mind. Many of the technical tasks and problems in this subdiscipline of computer science stimulate the imagination and creativity of technically oriented workers particularly strongly. Making a thinking being out of the computer, giving the computer the ability to understand spoken language, making it possible for the computer to see -- goals like these offer nearly irresistible temptations to those among us who have not fully sublimated our playful sandbox fantasies or who mean to satisfy our delusions of omnipotence on the computer stage, i.e., in terms of computer systems. Such tasks are extraordinarily demanding and interesting. Robert Oppenheimer called them "sweet." Besides, research projects in these areas are generously funded. The required moneys usually come out of the coffers of the military -- at least in America.

Tempting, Seductive Fable

It is enormously tempting and, especially in artificial intelligence work, seductively simple, to lose or hide oneself in details, in subproblems and their subproblems, and so on. The actual problems on which one works -- and which are so generously supported -- are disguised and transformed until their representations are mere fables, harmless, innocent, lovely fairy tales.

For example, a doctoral student characterized his projected dissertation task as follows:

A child, perhaps six or seven years old, sits in front of a computer display on which one can see a kitten and a bear -- all this in full color of course. The kitten is playing with a ball. The child speaks to the computer system: "The bear should say 'thank you' when someone gives him something." The system responds in a synthetic but nevertheless pleasing voice: "Thank you, I understand." Then the child again: "Kitty, give your ball to your friend." Immediately we see the kitten on the computer display throw the ball to the bear. Then we hear the bear say: "Thank you my dear kitten."

This is the kernel of what the system, whose development is to constitute the student's doctoral work, is to accomplish. Seen from a technical point of view, the system is to understand spoken instructions -- that alone is not simple -- and translate them into a computer program which it is then to integrate seamlessly into its own computational structure. Not at all trivial, and beyond that, quite touching.

Dangerous, Horrible Reality

Now a translation to reality. A fighter pilot is addressed by his pilot's associate system: "Sir, I see an enemy tank column below. Your orders please." The pilot: "When you see something like that, don't bother me, destroy the bastards and record the action. That's all." The system answers: "Yes sir!" and the plane's rockets fly earthward.

This pilot's associate system is one of three weapons systems which are expressly described, mainly as a problem for artificial intelligence, in the Strategic Computing Initiative, a new major research and development program of the American military. Over six hundred million dollars are to be spent on this program in the next four or five years.

Hindering Thought and Quieting Conscience?

It isn't my intention to assail or revile military systems. I intend this example from the actual practice of academic artificial intelligence research in America to illustrate the euphemistic linguistic dissimulation, the effect of which is to hinder thought and, ultimately, to still conscience.

I don't quite know whether it is especially computer science or its subdiscipline artificial intelligence that has such an enormous affection for euphemism. We speak so spectacularly and so readily of computer systems that understand, that see, decide, make judgments, and so on, without ourselves recognizing our own superficiality and immeasurable naivete with respect to these concepts. And, in the process of so speaking, we anesthetize our ability to evaluate the quality of our work and, what is more important, to identify and become conscious of its end use.

The student I mentioned above imagines his work to be about computer games for children, involving perhaps toy kittens, bears and balls. Its actual end use will likely mean that some day a young man, quite like the student himself, who has parents and possibly a girl friend, will be set afire by an exploding missile which was sent his way by a pilot's associate system shaped by the student's research. The psychological distance between the student's conception of his work and its actual implications is astronomic. It is precisely this enormous distance which makes it possible not to know and not to ask if one is doing sensible work or contributing to the greater efficiency of murderous devices.

One can't escape this state without asking again and again: "What do I actually do? What is the final application and use of the products of my work?" and, ultimately, "Am I content or ashamed to have contributed to this use?"

Once we have abandoned the prettifying of our language, we should begin to speak realistically and in earnest about our work as computer professionals. We should, for example, ask questions with respect to attempts to make it possible for computer systems to see. Progress in this domain will, with absolute certainty, be used to steer missiles like the cruise and the Pershing ever more precisely to their targets. And at their targets, mass murder will be committed.

The Computer "Only a Tool"?

Such statements are often countered with the assertion that the computer is merely a tool. As such it can be used for good or for evil. In and of itself, it is value free. Furthermore, scientists and technicians cannot know how the products of their work will be applied, whether they will find a good or an evil use. Hence scientists and technicians cannot be held responsible for the final application of their work.

Many scientists adopt the argument just stated as their own. They say that the systems on which they work can take men to the moon and bring them back just as these same systems can guarantee that missiles aimed at Moscow will actually hit Moscow when fired. They cannot know in advance, they say, which of these two or still other goals their work will serve in the end. How then can they be held responsible for whatever consequences their work may entail? So it is, on the whole, with computer professionals. The doctoral student I mentioned, who wishses to be able to converse with his computer display, does in fact believe that future applications of his work will be exclusively in innocent applications such as children's games. Perhaps his research is not sponsored by the Pentagon's Strategic Computing Initiative, perhaps he never even heard of the SCI. How then can he be assigned any responsibility for anti-human use to which his results might be put?

Becoming Militarized?

Here we come to the essence of the matter: today we know with virtual certainty that every scientific and technical result will, if at all possible, be put to use in military systems. The computer, together with the history of its development, is perhaps the key example. In these circumstances, scientific and technical workers cannot escape their responsibility to inquire about the end use of their work. They must then decide, once they know to what end it will be used, whether or not they would serve these ends with their own hands, that is, with the psychological distance between themselves and the final consequences of their work reduced to zero.

I think it important to say that I don't believe the military, in and of itself, to be an evil. Nor would I assert that the fact that a specific technology adopted by the military is, on that ground alone, an evil. In the present state of the evolution of the sovereign nation-state, each state needs a military just as every city needs a fire department. (On the other hand, no one pleads for a fire station on every corner, and no one wishes for a city fire department that makes a side business out of committing prophylactic arson in the villages adjacent to the city.)

But we see our entire world, particularly its universities and science and engineering facilities, being increasingly and ever more profoundly militarized every day. "Little" wars burn in almost every part of the earth. (They serve in part to test the high-tech weapons of the "more advanced nations.") More than half of all the earth's scientists and engineers work more or less directly in military institutions or in institutions supported in the main by the military.

Effective Opposition

Probably the most pandemic mental illness of our time is the almost universally held belief that the individual is powerless. This (self-fulfilling) delusion will surely be offered as a counter argument to my thesis. I demand, do I not, that a whole profession refuse to participate in the murderous insanity of our time. "That cannot be effective," I can already hear it said. "Yes, if actually no one worked on such things...but that is plainly impossible. After all, if I don't do it, someone else will."

(please turn to page 26)

Astronomy, Instruments, and Computers —Part 2

Sandra Blakeslee
23299 Red Rock Rd.
Topanga, CA 90290

It is true ... that telescopes will become front-end peripherals for computers and electronic instruments."

(Part 1 appeared in the January-February, 1987 issue of "Computers and People".)

Designs for New Giant Telescopes

Among American astronomers, ideas for new giant telescopes, beyond the MMT (Multiple Mirror Telescope), began to gel about 1975. They knew that obtaining clues to the big cosmological questions involving the chemistry of the universe and the birth and death of stars and galaxies simply required more photons than they could gather. An arbitrary goal of a 25-meter telescope was set.

The basic purpose of an optical telescope is to collect light, and one idea advanced was to develop an array of light-gathering telescopes positioned either in a line or in a circle. Single telescopes would work individually or by combining their light into one image. This approach is extremely difficult because optical combining problems are formidable.

Another idea for a new-technology telescope involves siderostats -- flat mirrors that collect light and direct it to a curved mirror, which focuses the light. The problem with this approach is that very large collectors and focusers are needed. The design is impractical and essentially increases classical mirror problems.

Another idea is called the bowl concept. It is a huge spherical mirror (telescope mirors are usually parabolic) set in a bowl-like depression in the earth. The ground would effectively brace such a mirror and simplify the problem of mechanical supports. The problem is that such a mirror would have a very small field of view, and images would have to be corrected for spherical aberrations. It would have to be huge. Another concept, called the rotating shoe, involves placing a section of the bowl on a rotating bearing, which would reduce the required size but would not correct the other problems.

Yet another approach, now actively embraced by German astronomers, is to build very large mirrors of metal. Metal can be formed into very large mirror shapes and can be coated to reflect light efficiently. Drawbacks are that metal has a way of warping when thermally stressed. Also, over a long period of time, the mirror blank itself can undergo dimensional change called creep, which is a slow relaxation of the material to another stress level. Ways to combat these problems would be to control the temperatures around the mirror and to make periodic adjustments to its shape.

Two Sensible Designs for Gathering More Light

Larry Barr, program manager for new-technology telescopes at NOAO (National Optical Astronomy Observatories), says that by 1980 there were two competing designs that made the most sense. "When you boil it down, there are only two ways to get more light-gathering aperture. One is to build enough telescopes with one-piece mirrors and combine their light beams; another is to build a very large mirror from a lot of small ones put together like a mosaic. There are no magic formulas."

In 1980, through NOAO, which coordinates government-funded observatories for the National Science Foundation, the astronomical community began comparing the two approaches, to pick one for what became known as the NNTT, the National New Technology Telescope. The NNTT would be a 15-meter telescope to be built in the 1990s, potentially in Arizona or Hawaii.

In July 1984 a committee of astronomers picked a large multiple-mirror design being pursued by Roger Angel and Neville Woolf at the University of Arizona over a segmented-mirror design by Jerry Nelson and a University of California group at the Lawrence Berkeley Laboratory. The decision, participants say, was extremely close and difficult. The Angel-Woolf design ultimately won because it is believed to have a little more flexibility.

Big, Fast Mirrors as Components

Angel is building big, short-focal-length -- fast -- mirrors. He has developed an ingenious method of spincasting parabolic slabs of glass of very short (f/1) focal length. He proposes to build larger and larger versions until he successfully casts an 8-meter component mirror. Several such giant mirrors would eventually be placed in a multiple mounting. The Carnegie Institution recently committed $20 million to Angel's group to cast a single 8-meter mirror. The University of Arizona and Ohio University are seeking partners to help buy one or two of the 8-meter mirrors.

Angel says he was inspired by the MMT. The MMT mirrors, he says, are composed of pieces of quartz glass fused into a honeycomb shape. "We started with that idea," he says. But he wanted to have a single piece of glass in a larger size. It was natural to cast the mirror as a single piece. "When you think about it," he says, "once you've done all the fiddly work of making the molds, there is not a whole lot more."

The driving force behind new telescope designs, says Bob Shannon, professor of optical science at the University of Arizona, is to make telescopes more economically. That, he says, means mirrors with shorter focal lengths. The Hale telescope on Mount Palomar has a 3.3 focal ratio of f/3.30. The focal length of its primary mirrors is 3.3 times the diameter of the mirror. The generation of 150-inch telescopes built in the 1960s have focal ratios of f/2.8. However, Shannon says, "the difficulty of making the mirror of a given diameter goes in about the third power inversely of the focal length. The amount of glass that has to be carefully removed to make an aspheric mirror of f/2.8 instead of f/3.5 is about twice as much."

Working with Pyrex

"Our goal is not to make cheap mirrors but to make the best mirrors we know how," Roger Angel says. "It so happens the material we can use, Pyrex glass, is inexpensive." If a mirror should break, he says, it would not be an unconscionable disaster.

For decades astronomers have been making mirrors out of low-expansion materials such as a ceramic called Cer-Vit. The Pyrex glasses, once a standby, had a bad name because mirrors made from them had a relatively high thermal expansion coefficient. In large slabs, Pyrex (or any glass) does not come to thermal equilibrium with surrounding air. Differences in temperature throughout its bulk can distort an image in the mirror's surface.

The Hale telescope on Mount Palomar and the 3-meter Lick Observatory instrument are the only major instruments made of Pyrex. The thermal problems of these large mirrors were reduced by making the back of the mirror a rib structure. This minimized temperature variations within the glass and also reduced weight. The observatory itself is air conditioned during the day to maintain nighttime temperatures as much as possible.

Honeycombing may be more effective. When Pyrex is used to make a honeycomb, Angel says, there are two advantages. First, the result is a lightweight structure. Second, the lightness minimizes thermal problems. "It means," Angel says, "we can use Pyrex once again. It is a great material because it is so easily cast."

Making an 8-Meter Mirror

Angel's group decided on an 8-meter mirror because, as he says, "it is scientifically big enough to do better astronomy." So far, he has successfully made mirrors up to 1.8 meters in diameter.

To make a mirror, Angel first places blocks of glass in a circular vat that contains rigid, honeycomb-shaped molds made of a styrofoam-like, ceramic, refractory fiber material that does not melt under high heat. The molds are then placed in an oven and rotated. The glass melts through the cracks around the solid honeycombs, which are not affected by the heat. The result is that a smooth layer of glass forms above the honeycomb blocks and, because of centrifugal forces on the free liquid surface, assumes the shape of a parabola. This curved top layer is destined to become the surface of the mirror. Behind that thin layer, however, is a honeycombed ribbing of glass. The styrofoam-like material is later chipped away. The mirror is lightweight and strong. The spinning rate determines the curvature of the parabola, which determines the focal length of the telescope.

Shorter focal lengths mean that stubbier, more economical telescopes can be built.

There is only half as much glass in a spin-cast mirror as in an equivalent flat mirror blank, which means much faster cooling. A solid 8-meter mirror would take a year to cool down from 2,000 degrees to room temperature. Once annealed, the amount of glass scraped out of the blank to make a parabola is about equal to the weight of the rest of the mirror. A spin-cast mirror the same size takes only weeks in order to anneal and less of its glass is wasted in the polishing stage.

Angel's mirrors are so carefully made that they come out of the oven to within millimeters of their final surface configuration. To grind and polish them, he is working on a new method, using a computer-driven machine tool.

Computers to Coordinate Polishing

In conventional telescope mirror polishing, a lap with abrasive material is slowly dragged over the mirror blank to tease out the right parabolic shape. It is a tedious procedure that is fraught with potential for errors.

Angel plans to use a computer and a series of motor-driven actuators on the lap to grind the desired parabola on virtually the first cut. (Conventional polishing is a gradual conversion process; the new motor-driven method makes conversion to the final shape very rapid.) As the lap moves across the glass, instructions -- at the rate of one megabit per second -- will be sent to the lap telling it to be the right shape on the glass each millisecond. The lap will essentially follow the shape of the desired parabola in its computer-driven memory. The numerically controlled lap should be able to finish a large mirror in one year instead of the ten years (World War II added delays) it took to polish the Hale telescope primary.

If Angel is successful, four of his big 8-meter mirrors will be put into the NNTT on a multiple-mirror common mount similar to that of the MMT. The mirrors, with their low thermal mass, should not warp with changes in air temperature.

Finally, Angel's mirrors will be coated with aluminum or possibly new layered coatings -- such as a copper-silver-aluminum mixture -- that, in tests, have been shown to enhance reflectivity and resist tarnishing.

Problems With Image Alignment

While similar to the MMT, the NNTT will have significant differences, Larry Barr explains. First, it will be more accurately phased; that is, the light combined at the focal point will make an image unaffected by the fact that it was made at four different telescopes. Phasing is thus an image-alignment system. The MMT, Barr says, "was never designed to be phased, so it is a credit to the MMT people that they can phase the thing in a limited sort of way."

Phasing is difficult, says Barr. Light coming from a star arrives at the earth's atmosphere essentially in a flat plane. But as the flat wavefront goes through the atmosphere, it is distorted. It gets out of phase. The astronomer captures a little piece of that wavefront and must then try to put it back together.

To do so, the astronomer must first deal with a diffraction pattern. Any telescope samples only a little piece of that incoming wavefront. The image that is formed has the shape and characteristics of the device that formed the image. Different mirrors will produce different diffraction patterns. "You are stuck with the diffraction pattern," Barr says, "and it's as good as you can ever do with a telescope. Our goal is to combine the light from four telescopes in such a way that they're perfect except for the diffraction patterns formed by these finite-sized telescopes."

The four images are to be aligned with the help of lasers. First, a laser beam will be projected through each telescope out into space. A little part of the beam will be intercepted by a bridge telescope. There are four such bridges in the NNTT design, and each will sit over two primary mirrors, looking down on them and overlapping them slightly. The bridge telescopes will capture portions of the laser light and send the light down to a common focus, where an image will be formed. When the optical paths of the laser light samples are equal as reflected in the image, it will mean that the telescopes are aligned. A detector at the image will provide information for controlling the secondary mirrors to make corrections when needed.

The MMT used the same idea for a while, Barr says, but problems were encountered. with disturbances in the laser beams. The beams would wander around as the seeing conditions around the telescope changed. All kinds of unforeseen and silly little mundane problems -- such as insects confusing the computer sensors -- made the system anything but automatic, recalls MMT director Fred Chaffee. Ultimately, MMT scientists solved the alignment problem with a different technique. A video camera looks at the image in each telescope, digitizes it, and then feeds it to a computer. The computer also has a video image of the object and combines all the images into one.

Barr says the NNTT will avoid such problems by not depending on the laser's position for any instant in time. Rather, the NNTT scientists will average the laser positions. "We will still need to know if the spot picked on the mirror is representative of the rest of the mirror," says Barr, "but it's a matter of calibration. By trial and error, we expect to find the best position."

The NNTT concept is now in the second year of a three-year program to complete the engineering of the selected design "to a point where we can put together an intelligent proposal to NSF," Barr says. Astronomers are hopeful it will be funded.

Private Funding for the Segmented-Mirror Telescope (SMT)

In the meantime, the second major design for future telescopes has not been shelved. In fact, it has received private funding in California and is certain to be built before the NNTT.

It is the segmented-mirror telescope, the SMT, design-engineered by the group at Lawrence Berkeley Laboratory headed by Jerry Nelson. Nelson says he picked a 10-meter aperture because with it "you can do something qualitatively new in astronomy."

Nelson simply "cut" (in a figurative sense) a 10-ten meter primary into 36 hexagonal, 1.8-meter-diameter pieces -- like tile in a huge mosaic. The trick, he says, was to fashion each piece into a segment of a parabolic surface and make the pieces operate as a single mirror.

Nelson and Terry Mast first designed an active control system that places three actuators, for three degrees of freedom, behind each segment. To know where each mirror is at any time, they settled on an edge-sensing system. A displacement sensor bridges neighboring segments. By design, the sensor, which is a capacitor, can detect only relative motions that are height differences between two segments. When the sensors are correctly placed, there is enough information generated to tell where all the mirrors are at any time. The sensors, designed by George Gabor, are bolted to a universal mounting plate at the back of each segment. Each actuator is a motor-driven screw.

Nelson, like Roger Angel, has developed a new method for polishing his new-technology telescope. Each segment needs to be a different part of a parabola and they are off axis; that is, they are not surfaces of revolution about their own centers.

Stressed Mirror Polishing

This invention, by Jacob Lubliner of the University of California at Berkeley and Nelson, is called stressed mirror polishing and uses the fact that when an optician rubs two pieces of glass together, one (unless corrections are made in the rubbing) will take on a perfectly spherical shape. It is the easiest way to polish, and Nelson decided to take advantage of its time-saving simplicity. In his method, a mirror segment is hung with 48 weights along its edges to deform its surface to the desired asymetry. It is then polished to a sphere. When the weights are removed, the segment relaxes -- like potato chips, explains Nelson -- into the correct shape. The worst-case segments have peak-to-valley deformations of 200 microns across their surfaces. If the segments are not perfect for any reason, Nelson could put a warping harness on the back to make final shape corrections.

Nelson's segmented-mirror telescope is going to be built on Mauna Kea in Hawaii with a $70 million grant from the Keck Foundation. After construction, the University of California will fund day-to-day operation of the Keck Telescope, and astronomers from both institutions will share time on it.

Telescopes as Front-End Computer Peripherals

The future of astronomy will include a heavy flood of data and problems with archiving. "In the old days everything was on photographic plates," says NOAO's Jacques Beckers. "Now it's all on computer tape. We will have to put new information on laser discs." It is true, he says, that telescopes will become front-end peripherals for computers and electronic instruments.

Future astronomers, with the giant new-technology telescopes, will see further back

(please turn to page 26)

Questions, Intelligence and Intelligent Behavior —Part 1

Martin A. Fischler
Oscar Firschein
Artificial Intelligence Center
Stanford Research Institute
Menlo Park, CA 94025

"Many people believe that the ability to communicate freely using some form of natural language is an essential attribute of an intelligent entity."

Our purpose in this article is to address three broad questions about the nature of intelligence:

- What is intelligence, and to what extent is it a unique attribute of the human species?

- How can intelligence be measured or evaluated?

- What is the nature of the mechanisms that are capable of intelligent behavior? In particular, can a machine be designed to display intelligent behavior?

The Recognizing of Intelligence

Intelligence is easier to recognize than to define or measure. While the word "intelligence" is used in ordinary conversation, and has a dictionary definition, it has no agreed-upon scientific meaning, and no quantitative natural laws relating to intelligence have as yet been discovered. In view of this situation, the concept of intelligence is subject to change as our understanding of human intelligence increases. Further, without a scientific definition, much of the social debate over matters relating to intelligence (e.g., contentions about racial differences with respect to intelligence) cannot be rationally resolved.

A dictionary definition of intelligence includes statements such as (1) the ability to meet (novel) situations successfully by proper behavior adjustments; or (2) the ability to perceive the interrelationships of presented facts in such a way as to guide action toward a desired goal. We can associate the word "learning" with the first statement, and goal-oriented behavior, problem solving, and understanding with the second. Some additional attributes of intelligence include reasoning, common sense, planning, perception, creativity, and memory retention and recall.

The Components of Intelligent Behavior

Theories of intelligence are primarily concerned with identifying the major independent components of intelligent behavior, and determining the importance of, and interactions between mechanism, process, knowledge, representation, and goals. In particular, such theories address the following issues:

- Performance theories: How can one test for the presence or degree of intelligence? What are the essential functional components of a system capable of exhibiting intelligent behavior?

- Structural/function theories: What are the mechanisms by which intelligence is achieved?

- Contextual theories: What is the relationship between intelligent behavior and the environment with which an organism must contend?

- Existence theories: What are the necessary and/or sufficient conditions for intelligent behavior to be possible?

Theories are statements, circumscribed by definitions, about objects and their relationships that are implicit in a body of knowledge. Thus, definitions and theories of intelligence cannot be separated. Quantitative definitions of intelligence range from implicitly defining intelligence as that human attribute which is measured by IQ tests, to assuming that the total information processing capacity of the brain is measured by its size (beyond that needed to support normal body functions).

However, the dimension along which definitions of intelligence differ most is the structural (internal) versus the contextual (external). At the structural extreme, intelligence is viewed as the competence of the human (or animal) nervous system to reason. At the contextual extreme, intelligence is viewed as the ability of an organism to adapt to its physical and social environment. In the latter case, goals, expectations, stored knowledge, and prior experience are as important and relevant as the internal reasoning machinery.

Natural or Abstract?

Theories of intelligence are largely dependent on whether we define intelligence to be a natural phenomenon appearing in living organisms (especially man), or whether we define it to be an abstract facility with certain specified properties. If intelligence is viewed as an outgrowth of specific biological structures, then it is reasonable to ask whether a single or coherent mechanism produces intelligent behavior, or whether intelligence is the result of a number of relatively independent processes. From a practical standpoint, we might also ask what kinds of measurements are needed to predict human performance in specified tasks requiring intelligence.

For example, if intelligence is a highly integrated process, then it is quite possible that a single number, such as an IQ test score, could be a good predictor of a human's ability to perform in almost any intellectual task domain. To the extent that intelligence arises from a loosely integrated combination of different mechanisms, prediction of human performance would depend on tests much more closely related to the specific task of interest.

Composite or Singular?

Most psychological theories of intelligence, and intelligence tests that implicitly arise from these theories, assume that intelligence is a composite of a relatively small number of component factors, possibly dominated by a single integrating factor. These theories can be called "performance theories," since they are based on measurements of performance and make assertions about relationships and correlations between different tests of performance. Such theories are largely empirical and, while they have significant practical utility, they offer very little insight into the nature of intelligence. As noted by B.T. Butcher in "Human Intelligence":

> The study of human intelligence has yielded a large accumulation of knowledge about individual differences, but very little about the basic laws of cognitive functioning....For a concept to be valuable it should have more than purely statistical support, and be more than a blind abstraction from a set of correlated performances.

Most of our concern will be with what might be called structural or function theories of intelligence. These are theories that propose certain physical or formal structures as the basis for intelligent behavior, and then examine the functionality that results. For example, if we assume that intelligence is a result of formal logical inference, then we might ask if there are human capabilities that could be shown to be unachievable in the formal system because of limitations inherent in logical reasoning. Logical systems do indeed have limitations that we do not usually ascribe to people.

Existing or Nonexisting?

Finally, there are largely philosophical theories about the physical conditions necessary for the mechanization of intelligence; we call these existence theories. For example, there is a school of thought that asserts that intelligence is a nonphysical property of living organisms, and cannot be recreated in a machine. Another school holds that intelligence is an "emergent" property of organic matter. Silicon-based microcircuits that are used in digital computers are "inadequate", but when we eventually learn how to build machines out of organic compounds, we might have a chance of inducing intelligent behavior.

One other school believes that intelligence is a functional property of formal systems, and is completely independent of any physical embodiment. This latter viewpoint is the one with which we will be primarily concerned.

We will discuss the attributes of intelligence and intelligent behavior, describing mechanisms that are capable of achieving such behavior in both living organisms and machines. However, we will not provide a precise definition of intelligence. Nor will we do much to "explain" or elucidate the conscious awareness that seems to be an essential component of human intelligence. Introspectively, there appears to be an "inner entity," the mind, which views the world through the body's sensory organs, "thinks," "understands," and causes the body to react in an appropriate manner.

Philosophers and Theories of Mind

A primary concern of philosophy is the attempt to understand the relationship between the internal world of our conscious awareness and the external physical world. Plato held that the mind (psyche) was in charge of the body and directed its movements. In the "Phaedrus" Plato spoke of the mind as having both appetitive desires and higher desires, and having also a rational capacity to control, direct, and adjudicate between these two types of desires. Later theories held that man was made of two substances, mind and matter.

The theory that the mind and body are distinct, known as "dualism," was given its classical formulation by Descartes in the seventeenth century. In his "Discourse on Method" (1637) he argued that the universe consists of two different substances: mind, or thinking substance, and matter, which can be explained by science and mathematics. Only in man are mind and matter joined together. His concept was that mind was an immaterial nonextended substance that engages in rational thought, feeling, and willing. Matter conforms to the laws of physics with the exception of the human body, which Descartes believed is causally affected by the mind, and which causally produces certain mental events. Thomas Hobbes, John Locke, and David Hume originated the idea that thoughts obey physical laws and can be characterized as computational processes. A basic problem that must be dealt with in this theory is how interaction can occur between the nonphysical and the physical.

Current Thought

The current dominant school of thought regards mind as being a purely physical phenomenon. Carl Sagan in "The Dragons of Eden" sums up this new view succinctly: "My fundamental premise about the brain is that its workings -- what we sometimes call 'mind' -- are a consequence of anatomy and physiology and nothing else." A similar view by R.M. Restak in "The Brain" is based on a belief that signals from the brain will some day be understood:

> Since the development of appropriate technologies, it has become obvious that thoughts, emotions, and even elementary sensations are accompained by changes in the state of the brain...a thought without a change in brain activity is impossible...to understand the "mind," therefore, it is necessary to understand the brain --- how concepts are arrived at, the mechanisms underlying perceptions, memory, the neuro-chemistry of our emotions, and so on.

The information processing model is used by Newell and Simon. They view formal logic as a way of capturing ideas by symbols, and the algorithmic alteration of such symbols as leading to mindlike activity: "The persistence of concern with the mind-body problem can be attributed in part to the apparent radical incongruity and incommensurability of 'ideas' -- the material of thought -- with the tangible biological substances of the nervous system."

Duality of Mind and Body

Those who take the above computational point of view feel that the mind-body problem will disappear when we have demonstrated the operation of mind using formalisms and algorithms for manipulating symbols.

But one should not think that all modern researchers look at duality with scorn. In his final book, "The Mystery of the Mind," the famous neurosurgeon Wilder Penfield doubts that an understanding of the brain will ever lead to an explanation of the mind: "Consciousness of man, the mind, is something not to be reduced to brain mechanisms." Another example of this point of view is contained in "The Self and the Brain" by Karl Popper and John Eccles, an updated plea for dualism, the belief that the brain and the mind are distinct entities.

Until someone provides convincing proof of the physical basis of mind, we can expect the mind-body debate to continue.

Assessing Human Intelligence

There is no formal or scientific definition of intelligence that is widely accepted. If intelligence cannot be defined, then it certainly cannot be measured in any precise or comprehensive manner. If intelligence tests do not measure intelligence, what do they measure?

The purpose of most of these tests is to predict the future performance of the person being tested with respect to an ability to compete or perform in an academic program or in a skilled work task. Whether or not an "intelligence test" actually does have the required predictive power can only be determined by extensive testing in the specific application area.

There are a number of intelligence tests in widespread use, one of the most popular being the Terman-Merrill revision of the Binet-Simon intelligence scale. The original Binet-Simon work was performed in the period 1905-1911. Binet insisted on three cardinal principles for using his test:

1. The scores are a practical device and are not intended as the basis for a theory of intellect. They do not define anything innate or permanent. What they measure is not "intelligence."
2. The scale is a rough, empirical guide for identifying mildly retarded and learning-disabled children who need special help. It is not a device for ranking normal children.
3. Whatever the cause of difficulty in children identified for help, emphasis should be placed on improvement through special training. Low scores should not be used to mark children as innately incapable.

All of his warnings were disregarded, and his scale was used as a routine device for testing all children. The Binet-Simon test was superseded by Terman's 1916 standard version, and then by the Terman-Merrill revision of 1937, and by a later revision in 1960. It is interesting to note that the procedure for selecting questions for this test was that the questions had to satisfy certain preconceived notions of what results the test should produce. This is standard practice in all intelligence test construction. For example, questions that yield systematically higher scores for either boys or girls are eliminated. By use of question selection and scoring procedures, the test was constructed so that for the white American population, biased somewhat toward urban and above-average socioeconomic level persons, the scores would have a normal distribution with an average score of 100, and a standard deviation of 16. This means that 50 percent of the reference group (white Americans) would score under 100; 85 percent would score under 116; 97.5 percent would score under 132, etc.

Another commonly used intelligence test, the Wechsler intelligence scale, uses separate tests for adults and for children. This test is divided into two main parts, one to test predominantly verbal ability, and a second to test performance. Even though the Wechsler and the Binet tests have somewhat different philosophies and different categories of questions, they use similar principles of test construction and produce scores that are in reasonable agreement.

The Challenging of Intelligence Tests

Starting in the 1960s, the role and value of intelligence tests have been seriously challenged. In particular, critics have argued that these tests take too narrow a view of intelligence, and that they are based on such dubious assumptions as: (a) A child is born with a fixed or predetermined level of intelligence; (b) IQ tests can measure this intelligence; (c) IQ scores will show little variation from early childhood to old age; and (d) the tests employed, relatively unchanged since their introduction in the early 1900s, are good predictors of human performance. Not surprisingly, political and social concerns have been intermixed with issues of scientific validity in addressing the question of what is reasonable and meaningful in regard to the testing of human intelligence.

Assessing Machine Intelligence

If one were offered a machine purported to be intelligent, what would be an appropriate method of evaluating this claim? The most obvious approach might be to give the machine an IQ test. As will be seen in later chapters, we already know how to build machines that can perform quite well on selected portions of such a test. For example, machines can currently solve high school algebra problems, solve the type of geometric analogy problems used on IQ tests, answer questions about the content of a simple story, parse English sentences, etc. However, none of this would be completely satisfactory because the machine would have to be specially prepared for any specific task that it was asked to perform. The task could not be described to the machine in a normal conversation (verbal or written) if the specific nature of the task was not already programmed into the machine. Such considerations led many people to believe that the ability to communicate freely using some form of natural language is an essential attribute of an intelligent entity.

The Turing Test

In 1950, Alan Turing proposed an "imitation game" to provide an operational answer to the question, "Can a machine think?" The

(please turn to page 25)

Artificial Intelligence and Natural Language Systems

Susan J. Scown
Digital Equipment Corporation
200 Baker Ave.
Concord, MA 01742

"As with all natural language systems, the more limited is the domain of discourse, the better is the translation from human words to computer concepts."

Communicating with Computers

When we want to communicate with computers, we have to do it on their terms, in their language, and in their media. We have to learn Assembler, BASIC, COBOL, FORTRAN, or some other programming language. These languages tend to be unforgiving if we put a period in the wrong spot or leave out a parenthesis or some other minute detail. There are currently some attempts to alleviate the communication problem with interfaces such as menus, online help facilities, and graphic icons. These are still somewhat cumbersome, require some training, and tend to be very system- or application-dependent.

Natural Language Communication

Now, researchers are developing "natural language" systems that can accommodate our native tongues, such as English, Japanese, and French. Research and implementation in natural language communication have addressed problems of input and output and how to get the computer to manipulate and respond to expressions in natural language. Natural language input and output are done through both text and sound. Communication via text is accomplished with traditional hardware mechanisms -- keyboards, printers, and video monitors. Natural language communication via sound is accomplished with hardware mechanisms packaged as speech recognition and synthesis systems. Some of these recognition and synthesis systems are supported by AI (artificial intelligence) technologies that interpret spoken input or generate humanlike audible speech output.

Conventions in discussing areas within the field of natural language are as follows. "Natural language" is an umbrella term for communication with computers using our native languages. It includes language input, output, and understanding. "Natural language" is also sometimes used in a more restrictive sense to refer to the text branch of the language problem. "Speech understanding" refers to the ability of computers to respond correctly to spoken language. And "speech generation" refers to the ability of computers to output spoken language.

As we will discuss later, some practical commercial systems are already available in two of the areas of natural language communication-speech generation and natural language interfaces. Other goals, particularly speech understanding, require a great deal more research before they will become widely used in commercial settings.

Language Understanding

When we say that a computer "understands" language, we mean that it is able to process the plain language of the user, carry out the command, and generate appropriate output.

Uses for computers that can understand natural language include

- Interfaces to systems such as databases or operating systems, expert advisory systems, or robots.

- Machine translation systems to translate written materials from one language to another.

- Document-understanding systems that enable a computer to read and understand the information in documents well enough to summarize and redirect points of importance to various recipients and to organize and store information in order to answer questions on the contents.

Syntax and Parsing

Two kinds of analyses are performed on input to natural language understanding systems. One focuses on syntax (how words are structured into expressions), and the other focuses on semantics (the meaning of words within the context of expressions). These analyses together allow natural language systems to generate a paraphrase of the input expression in an internal representation language.

Syntax -- the role of each of the parts of a sentence -- is analyzed by parsing. Parsing is the process of identifying what function each word in a sentence performs and whether the input sentence is "legal" (complete and grammatical), according to a particular grammar. A variety of grammars, supported by different theories about verbal communication, have been used to govern the parsing process in natural language systems.

Semantics and Meaning

Semantic analysis, which focuses on the meaning of words and phrases, is then needed to clarify the system's understanding. Semantic analysis proceeds by associating words and their roles in an input sentence with information about the problem domain stored in the system's data or knowledge base. This knowledge can constitute a background context of expectations against which to interpret the input. The stored knowledge might be descriptions of objects, events, and relationships in the problem domain, descriptions of typical situations that might be encountered in the problem domain, or chains of events or procedures that occur given certain conditions in the problem domain.

In addition to using the information in the system's knowledge structures to analyze input, the system can store the input itself in knowledge structures, increasing and refining its knowledge as processing goes along. This helps the system to understand input in contexts larger than single sentences. The stored information provides a way to link references in one sentence to referents in another.

Natural language systems can combine information from the syntactic and semantic analyses to generate formalized representations of input sentences. The formalizations generated by the analyses can be stored in the knowledge base for comparison to other input or they may set off some activity in the system, such as answering a database query.

Even with the support of a grammar, parsing is difficult without other kinds of information. People use their knowledge about the world to help clear up ambiguities in the parsing process. For instance, we human beings can extract sense from a sentence like "Sea otters crack mussels on rocks as they swim on their backs." We parse the phrase "as they swim on their backs" as modifying "otters" because we have knowledge about the world that tells us that rocks and mussels don't swim on their backs. (We would understand the sentence even though it is badly constructed.) A system without that knowledge would have trouble deciding how to parse the sentence. Other parsing ambiguities arise in words with double meanings, in idioms, and in pronouns. Semantic analysis helps to resolve these difficulties.

Ellipsis and Metaphors

Ellipsis also presents difficulties to computers. Ellipsis occurs when words are left out, creating grammatical incompleteness. People interpret elliptical expressions from the context. As an example, if someone said, "A gang of five robbers held up the bank this morning," a reporter asking, "Any arrested?" would be using an elliptical expression that most listeners would understand.

Another problem for natural language systems is created by metaphors, like "Maria Garcia is a pillar of the community." Metaphors cannot be directly translated by a computer without explicit instruction.

Computer programs also have not yet been created that can read between the lines to interpret a message based on the goals of the communicator. For instance, if you are wrapping a package and ask someone, "Do you have any tape?" you are making a request, not taking an inventory.

Natural Language Interfaces

Natural language interfaces allow people to use subsets of their native language to communicate with computers in restricted domains. It is expected that a major use of the technology will be in organizations that query databases. Managers, office workers, and technical professionals will be able to get needed data from the computer without going either to data processing courses or through the data processing department. This will make it worthwhile to pose ad hoc questions that may not warrant the development of full-fledged programs but that do contribute useful information to the problem at hand. A long-range goal is to make it easier and more comfortable for people to use their home computers and to program and control robots.

A few commercial natural language interfaces to databases are already on the market. One of the earliest of the commercial products was INTELLECT, developed by Artificial Intelligence Corporation, Waltham, Massachusetts. INTELLECT is intended for use as a front-end interface to information retrieval applications in areas such as finance, marketing, manufacturing, and personnel. INTELLECT parses the user's natural language query into an internal representation that sets off a database search. To do this, in addition to using a grammar, the system also draws on knowledge of the database structure, database contents, a built-in data dictionary, and an application-specific dictionary. When a query results in the generation of more than one possible interpretive paraphrase, INTELLECT resolves the ambiguity by assigning preference ratings to the different paraphrases, choosing the one most highly rated due to its consistency with information in the database. If necessary, INTELLECT even asks the user which of several interpretations is correct.

Scripts

An approach to interfaces that emphasizes semantics has been taken by Professor Roger C. Schank, chairman of Yale University's computer science department, director of Yale's Artificial Intelligence Laboratory, and founder of Cognitive Systems, Inc., New Haven, Connecticut. Cognitive Systems has developed natural language interfaces to databases and conversational advisory systems. Cognitive Systems' products "map" natural language input into conceptual representations based on Schank's theory of conceptual dependencies, which capture the meaning of the input. In addition to storing information about meaning in conceptual representations, these systems incorporate information about the problem domain in various knowledge structures, such as scripts. These knowledge structures aid interpretation by providing the system with expectations, contexts in which to understand input. A script, for instance, is a description of what happens in a situation that conforms to a stereotype. When a scripted topic is presented to a system employing this knowledge structure, it has a set of expectations that helps it resolve ambiguities.

Cognitive Systems' products also combine expert systems with natural language systems to form "intelligent retrieval systems." These natural language systems set up a context and keep track of it during a conversation. This adds to the system's fund of expectations, helping to resolve ambiguity. It is also possible to build into the system profiles of the users' goals, so that the system can retrieve not only the specific information requested, but also related information that would be of interest to the user. And again, the profiles provide a context that helps the system decide how to interpret a request: one person's "year," for instance, might be fiscal, while another's might be the calendar year.

Natural language interfaces vary in how much users must conform their input to the structure of the system, some allowing for the use of language that is more natural than others. Most allow users to modify the dictionaries with which the systems are equipped and to add their own entries, with varying degrees of ease and allowing varying kinds of definitions. Some systems handle sentences in which words have been omitted by proposing a fleshed-out interpretation to the user and asking if this was the correct reading. On encountering a spelling error, some natural language systems ask for a correction, some automatically correct the word, and others simply stop processing the sentence. At this time, individual natural language interfaces must be specialized for particular subject matter contexts in order to interpret words and phrases correctly.

Limits of Knowledge

It should also be noted that today's natural language systems, and AI systems in general, are limited in knowledge, but the systems are usually not aware of these limits. Unlike people, the systems "assume" that they operate in a closed world, that their knowledge of the domain is complete and adequate. When a natural language system responds "no" to a query, it may really mean that the system doesn't know the answer. In reality, objects and relationships pertaining to most real-world domains change or are not modeled in the system. Another problem is that an appropriate response frequently depends on knowledge of the user's motivation in asking a question, and current systems are naive in this area.

Nevertheless, natural language interfaces are now available that successfully lessen communication obstacles to problem solving in some computing tasks and speed up access to information.

(please turn to page 25)

Fischler – *Continued from page 21*

game is played with three people, a man, a woman, and an interrogator who may be of either sex. The interrogator stays in a room apart from the other two, and attempts to determine which of the other two is the man and which is the woman. The man tries to convince the interrogator that he is the woman. Communication between the interrogator and each person is by teleprinter, and the interrogator is free to ask any question of the participants.

Suppose we now ask the question, "What will happen when a machine takes the part of the man in this game?" Turing felt that a machine could be considered "intelligent" when the interrogator decides wrongly as often when the game is played with the machine as when the game is played between a man and woman. It should be noted that to accomplish this, the machine must be able to carry out a dialogue in natural language and reason using an enormous database of "world knowledge." The "man-woman" formulation proposed by Turing is not usually stressed in describing the imitation game. Instead, the theme is usually the idea of a machine convincing an interrogator that it is a person.

The Turing test has more historical and philosophical importance than practical value; Turing did not design the test as a useful tool for psychologists. For example, failing the test does not imply lack of intelligence. The important central idea is that the ability to successfully communicate with a discerning person in a free and unbounded conversation is a better indication of intelligence than any other attribute accessible to measurement.

Man Is Not the Only Intelligent Animal

If we examine the attributes of intelligent behavior that were presented earlier, we can find examples of superior animal performance in each of the attribute categories. Until recently, however, it was believed that only man, of all animals, could produce (as opposed to understand) structured linguistic phrases to communicate meaning. Experiments recently have demonstrated that chimpanzees can learn American Sign Language and can learn to assign word meanings to physical tokens (e.g., small colored plastic disks), and then arrange these tokens into structured sentences to communicate with their trainers. Thus in an objective sense, it appears possible that man differs from the higher mammals mainly in degree of intellectual ability rather than in having some unique and unshared capability.

In a related sense, recent work by Gordon Gallup addresses the question: "Do minds exist in species other than our own?" Gallup defines "mind," "consciousness," and "self-awareness" to mean essentially the same thing. His operational test for self-awareness is that an organism can identify itself in a mirror; for example, a child can recognize his reflection at approximately a year and one half to two years of age. Gallup discovered that while humans, chimpanzees, and orangutans can learn to recognize themselves in mirrors, no other primates can! Thus, even though gorillas appear to possess some degree of linguistic competence, gorillas fail this particular test for self-awareness. Our understanding of the relationship between self-awareness, language, and intelligence is still at a very primitive state.

(Continued in next issue)

Based on Chapter 1, *Intelligence*, of *Intelligence: The Eye, the Brain, and the Computer* by M.A. Fischler and O. Firschein, published by and copyright 1987 by Addison-Wesley Publishing Co., Reading, MA 01867. Reprinted with permission.

Scown – *Continued from page 24*

Machine Translation

Work in machine translation from one natural language to another has revealed that the subtleties of human language do not easily yield to computerization. The word-for-word translation systems of twenty years ago just didn't work. Research in machine translation has made it increasingly clear that human language cognition is a very complex ability that requires many kinds of knowledge, including knowledge of the structure of sentences; the meaning of words; the patterns of conversation; the expectations, goals, and beliefs of the partner with whom you're conversing; and a great deal of knowledge about the world as well as knowledge about the particular topic of conversation.

Current implementations that most closely approach automatic translation may use syntactic and semantic information in order to translate words in context. Different systems require varying degrees of human assistance to edit machine-translated drafts or to assist in translating elements outside the bounds of the systems' abilities. In addition, systems described as "fully automatic" are, at this time, restricted to small domains. The speed of translation may be as slow as 600 words per hour for output that requires little editing or as fast as 60,000

(please turn to page 26)

Blakeslee – *Continued from page 17*

in time and win converts to new and wonderful cosmologies. However, the future also has a great loss in store for astronomers who have practiced their science in the second half of the twentieth century.

By tradition, astronomers have always gone "to the mountain" on nights assigned to them on the telescopes. The new telescopes, though, will be too complicated for astronomers to operate. Observing rooms will be computer consoles located away from the electronically sensitive telescope.

In the future, says NOAO director John T. Jeffries, an astronomer may be called up one evening and told, "Conditions on the mountain are perfect tonight for your experiment. Get ready right away." The astronomer will observe through an office computer hooked to a microwave link and give instructions to the telescope operator on the mountain. The telescope will be thousands of miles away, where the night skies are not polluted by city lights. And, sitting at their computers hooked to their gigantic computerized telescopes, astronomers will peer deep into the galaxy and far into the universe as photons never before seen are harvested.

As time goes on, Jeffries says, into the first third of the next century, "things may change and we'll go to the moon." Viewing the universe on the dark side of the moon has long been a dream of astronomers. "But for a long time yet," Jeffries says, "we'll stay here on earth in the electronic age of telescopes."

Based on an important article in *Mosaic*, Vol. 17, No. 2, Summer 1986, published by National Science Foundation, Washington, DC 20550. Reprinted from this magazine in the public domain.

Ω

Scown – *Continued from page 25)*

words per hour for output that is likely to need considerable editing. Faster speeds are achieved in some systems by constraining the input to shorter sentences or by setting lower standards for the quality of the output. As with all natural-language systems, the more limited is the domain of discourse, the better is the translation from human words to computer concepts.

(Continued in next issue)

This article is based on an interesting and important excerpt from Chapter 2 of *The Artificial Intelligence Experience: An Introduction*, by Susan J. Scown, copyright 1985 by and published by Digital Equipment Corporation, Maynard, MA 01754. Reprinted with permission.

Weizenbaum – *Continued from page 13*

First, and on the most elementary level, I must say that the rule "If I don't do it, someone else will" cannot serve as a basis of moral behavior. Every crime imaginable can be justified on its basis. For example, "If I don't steal the sleeping drunk's money, someone else will."

But it is not at all trivial to ask after the meaning of effectiveness in the present context. Surely, effectiveness is not a binary matter, an either/or matter. If what I say here were to induce a strike on the part of all scientists with respect to weapons work, that would have to be counted as effective. But there are many much more modest degrees of effectiveness toward which I aim.

I think it was George Orwell who once wrote, "The highest duty of intellectuals in these times is to speak the simplest truths in the simplest possible words." For me that means first of all the duty to articulate the absurdity of our world in my actions, my writings and with my voice. I hope thereby to stir my students, my colleagues, everyone to whom I can speak directly. I hope thereby to encourage those who have already begun to think similarly, and to be encouraged by them, and possibly rouse all others I can reach out of their slumber. Courage, like fear, is catching! Even the most modest success in such attempts has also to count as effectiveness. Beyond that, in speaking as I do, I put what I here discuss on the public agenda and contribute to its legitimation. These are modest goals that can surely be reached. But each of us must believe "it cannot be done without me."

Based on a report in the *Bulletin of the Computer Professionals for Social Responsibility*, November 1986, published by CPSR Inc., P.O. Box 717, Palo Alto, CA 94301; reprinted with permission.

Ω

CACBOL – *Continued from page 2*

7. Tools for Knowledge Engineering
   - Criteria for choosing a tool
   - Hardware environments
   - Classification of available tools
   - Survey of available tools

(Source: Catalog of Professional Development Seminars, Feb.-July, 1987, Control Data Corp., Institute for Advanced Technology, 1450 Energy Park Dr., St. Paul, MN 55108)

Ω

Annual Index

for "Computers and People", 1986, Volume 35

Articles and Short Reports

This index is arranged by last name of main author, then title (which may be condensed), then issue, then number of page.

Front Covers

Editorials

**(see page 28) If a topic can be expressed in more than one way, it is easy to list possible expressions (or possible subjects and predicates), so that the student's answer can also be compared and judged by the computer system as correct or wrong.

The Computer Almanac and Computer Book of Lists (CACBOL)

Instalments 45 to 50
16 lists, described on page 5 of each issue

Games and Puzzles for Nimble Minds and Computers

Maximdidge, Numble
each issue, page 28

Opportunities for Information Systems

Instalments 1 to 6
each issue, page 28

Opportunities for Information Systems

— Instalment 8

THE ASSESSING OF FREE WRITING

Edmund C. Berkeley, Editor

What is a good way to judge or assess the free writing of a student?

One of the arduous tasks of a teacher with a class of 15 or more students is the judging or assessing of their free writing. This task requires much effort. It is difficult to give this task to a computer system because understanding what the student is writing is regularly beyond the power of the computer system. As a result many compromises have been made:

1. The true-false question (where the chance is great that a guess will receive full credit)

2. The question with 1 correct answer and 4 "distractors" (a guess will probably receive at least 1/5 credit)

3. A question where only 1 of 60 choices is correct

4. A question where 2 or 3 paragraphs of answer are needed; and so on

In many written answers to questions, a teacher must judge free writing. For example a teacher may need to judge a couple of paragraphs submitted as an answer to a question in a test. The teacher must look for (1) certain topics, and (2) points to be covered for that topic. The student must show in his answer that he knows the topics and knows the points. This is true for many kinds of answers to many kinds of questions. Often however the correct answers must be expressed in free writing.

How do we come to grips with this situation?

A practical solution is quite easy.

If a topic or a point can be expressed in one and only one way, it is easy to use the expression worded as it stands as the evidence to the computer that the student's answer is correct. The words surrounding the expression are relatively unimportant, and can usually be ignored. **

Also, the teacher does not have to use uncritically the score of points reckoned by the computer system for a student's answers. He should make certain that the computer program developed for 2 or 3 different excellent answers for a class of 100 students assigns those students appropriate scores and produces reasonable and valid scores for everybody else.

In any case, what is needed here is assistance for the teacher, not his replacement. To manage the work of assessing free writing for 100 students, in such a way as to go from a load of 20 hours for the teacher to a load of 4 hours, is a substantial gain.

Ω

** see page 27

Games and Puzzles for Nimble Minds and Computers

Neil Macdonald
Assistant Editor

NUMBLE

A "numble" is an arithmetical problem in which: digits have been replaced by capital letters; and there are two messages, one which can be read right away, and a second one in the digit cipher. The problem is to solve for the digits. Each capital letter in the arithmetical problem stands for just one digit 0 to 9. A digit may be represented by more than one letter. The second message, expressed in numerical digits, is to be translated using the same key, and possibly puns or other simple tricks.

NUMBLE 8703

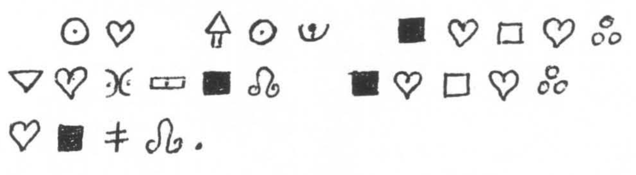
```
              T H E
        *   H E D G E
        -------------
            N A T G
          D D H D
        H B O S
      N A T G
    A S D T
    -----------------
    = A A B A E E A G
```

66512 43X31 OY8X8 65135Z1

MAXIMDIDGE

In this kind of puzzle, a maxim (common saying, proverb, some good advice, etc.) using 14 or fewer different letters is enciphered (using a simple substitution cipher) into the 10 decimal digits or equivalent signs, plus a few more signs. The spaces between words are kept. Puns or other simple tricks (like KS for X) may be used.

MAXIMDIDGE 8703



SOLUTIONS

MAXIMDIDGE 8701: Aim the bow at a definite end.
NUMBLE 8701: Do not love the moon more than the sun.

Our thanks to the following person for sending us solutions: T.P. Finn, Indianapolis, IN – Maximdidge 8611.

Ω