Computing:
Some Scientific Aspects

Leopoldo Bertossi
Carleton University
School of Computer Science
bertossi@scs.carleton.ca
www.scs.carleton.ca/~bertossi

Good News!

A mathematical problem had been open, without answer, since the 1930s

The conjecture (believed, but not proven):

Every Robbins Algebra is a Boolean Algebra

There was neither a proof nor a refutation, despite efforts of many good mathematicians

In December 1996, Robbins' conjecture was proved by means of a computer program

The new appeared in the New York Times, and was rapidly broadcasted via internet ...

http://www-c.mcs.anl.gov/home/mccune/ar/robbins/index.html

With Major Math Proof, Brute Computers Show Flash of Reasoning Power

The achievement would have been called creative if a human had done it.

By GINA KOLATA

Computers are whizzes when it comes to the grunt work of mathematics. But for creative, and elegant solutions to hard mathematical problems, nothing has been able to beat the human mind. That is, perhaps, until now.

A computer program written by researchers at Argonne National Laboratory in Illinois has come up with a major mathematical proof that would have been called creative if a human had thought of it. In doing so, the computer has, for the first time, got a toehold into pure mathematics, a field described by its practitioners as more of an art form than a science. And the implications, some say, are profound, showing just how powerful computers can be at reasoning itself, at mimicking the flashes of logical insight or even genius that have characterized the best human minds.

Computers have found proofs of mathematical conjectures before, of course, but those conjectures were easy to prove. The difference this time is that the computer has solved a conjecture that stumped some of the best mathematicians for 60 years. And it did so with a program that was designed to reason, not to solve a specific problem. In that sense, the program is very different from chess-playing computer programs, for example, which are intended to solve just one problem: the moves of a chess game.

"It's a sign of power, of reasoning power," said Dr. Larry Wos, the supervisor of the computer reasoning project at Argonne. And with this result, obtained by a colleague, Dr. William McCune, he said, "We've taken a quantum leap forward."

Dr. Wos predicts that the result may mark the beginning of the end for mathematics research as it is now practiced, eventually freeing mathematicians to focus on discovering new conjectures, and leaving the proof to computers.

But the result also may challenge the very notion of creative thinking, raising the possibility that computers could take a parallel path to reach the same conclusions as great human thinkers. Or it may be that since no one has any idea how humans think, the magnificent bursts of creativity that spring apparently full blown from the minds of geniuses are actually a result of hidden, computer-like drudge work in the unconscious recesses of the brain.

Dr. Stanley Burris, a mathematician at the University of Waterloo in Canada, said that the result was "the first sort of real breakthrough in automated theorem proving," and that it did seem to be different in kind from what went before. It shows, he said, that "it's a very thin line between the mechanical and the creative, and it may disappear."

Dr. Robert Boyer, a computer scientist at the University of Texas in Austin, hedged. "I think it's the most remarkable result in automated theorem proving in 30 years," he said, and "clearly a form of computer thinking." But, he added, "I don't want to make too much of that." It's best, he said, to think of a computer as "just another colleague, one that is sometimes helpful, but often not."

Dr. McCune's proof concerns a conjecture that is the very epitome of pure mathematics. "It has no applications," Dr. McCune said. His computer program proved that a set of three equations is equivalent to a Boolean algebra, that set of rules, familiar to generations of high school students, that govern unions and complements, and intersections among

Continued on Page B10

Computer Proof Shows Reasoning

Continued From Page B5

sets.

The problem was first posed in the 1930's by Dr. Herbert Robbins, who is New Jersey Professor of Mathematics at Rutgers University in New Brunswick. Dr. Robbins said that he worked on the problem for some time, and then passed it on one of the century's most famous logicians, Dr. Albert Tarski of Stanford University. Dr. Tarski, who is now dead, worked on the problem, included it in a book, and handed it out to graduate students and visitors.

Dr. Burris, for example said that Dr. Tarski suggested the problem to him in the early 1970's, while he was visiting Stanford for a couple of months. Dr. Tarski, he said, "liked to throw out challenging problems to people passing through."

While mathematicians were batting around Dr. Robbins's problem, computer scientists were striving to see if they could get computers to reason. Among them was Dr. Wos, who started working on automated reasoning in the 1960's. It was a time when computers were primitive, clunky and slow, and researchers were divided on how to proceed. Some believed the key was to figure out how humans reasoned and then to create computer programs that mimicked the process. Dr. Wos disagreed.

"Nobody knows how humans reason," he said. "When you talk to mathematicians and say, 'I understand you proved a great theorem. How did you do it?' They'll say, 'Well, I walked around my house a lot and I read some papers and I thought.'"

So he and his colleagues followed a different path. "We didn't ask ourselves what people do when they think," Dr. Wos said. "That was irrelevant." he said. Instead, he said: "We asked how can you tell a computer what this problem is about? How can you get it to draw conclusions that follow inevitably and logically from hypotheses and thereby prove theorems?"

He and his colleagues began writing programs in which the computer would assume that the hypothesis in question was false and would then examine the consequences. If it found a contradiction, that would be proof that the hypothesis was true. The computer would also assume that the hypothesis was true and do the same thing, looking for contradictions that would show it was false.

To prevent the computer from getting lost in checking out lengthy chains of extended consequences, the investigators added strategies like ignoring any logical statements that contained more than 100 symbols.

Dr. Wos's computer programs were soon able to find proofs for basic mathematical problems. "We could do the problems sometimes better than the students and sometimes, once in a great while, better than the professors could," Dr. Wos said.

For more than 16 years, Dr. Wos and his colleagues stuck to problems from mathematics textbooks. "Dr. Wos explained that when the computer tried to prove something for which a proof existed, and failed, the investigators knew that "there's a problem with our program." If they try to solve an unsolved problem, and fail, they have no way of knowing whether they missed something obvious.

"My own mathematician friends would say, 'Wos, why are you doing what we already know? Why don't you give us something new?'" Dr. Wos said. In the early 1970's, he said, he told one of his badgering friends that he thought it would be another 30, 40 or even 50 years before computers could solve major problems that had stumped mathematicians.

The first time they tried something new was in 1978, when Dr. Wos said, "a little baby problem" came along. They solved it, and then solved five others like it. Dr. Wos was ecstatic.

The group kept adding strategies to its programs. It added one recently that said to try things that worked in previous problems. Dr. Wos said some of his colleagues scoffed at that and that he himself did not know if it would work. But, he said, it turned out to be surprisingly useful.

In 1979, Dr. Wos learned about Dr. Robbins's problem. Although he and his colleagues tried to solve it from time to time with ever more refined computer programs, they failed. Dr. McCune joined the group in 1984 with a new Ph.D. from Northwestern University and a thirst to see how far he could push computers.

The fact that the Robbins conjecture was certifiably hard — it had, after all, stumped some of the best minds in mathematics and had gone unsolved for decades — appealed to Dr. McCune. But the problem also thwarted his best computer reasoning programs.

Finally, on Oct. 2, Dr. McCune gave the Robbins conjecture to a new automated reasoning program that he had written called EQP, for equational prover. Eight days later, on Oct. 10, the computer spewed out a proof. Dr. McCune, a low-key researcher, said he was "amazed." Dr. Wos, his exuberant supervisor, said "Bill was in heaven."

Encouraged, Dr. McCune tried to get the computer to refine the proof. He started his program searching for a better proof on Nov. 15. It found one on Nov. 25.

The proof will be published within a few months in The Journal of Automated Reasoning, Dr. Wos said. He will also put it on the Internet and will request more problems as certifiably hard as Dr. Robbins's problem.

When he saw that he had actually proved Dr. Robbins's conjecture, Dr. McCune called the 81-year-old mathematician at his office at Rutgers, amazing him with the news. In a telephone interview, Dr. Robbins said he was delighted. "Isn't that marvelous," he said. "I'm glad I lived long enough to see it."

But Dr. McCune certainly had a different experience than a mathematician might have had if he or she had created the proof with pure thought. So was something lost in the process? Did Dr. McCune have anything resembling a Eureka moment?

Well, not quite, Dr. McCune said. "I have a good feeling," he said. "In a sense, I have a feeling that the computer has been creative."

Dr. McCune, however, said he had little interest in speculating on the philosophical implications of his work. "I just work on the problems and try to solve them," he said.

Not so Dr. Wos, who loves to ponder the meaning and future of human reasoning. After all, he said, he insisted on calling his computer project "automated reasoning" rather than "automated theorem proving," as some suggested, explaining that "it's not just mathematics that I care about."

Dr. Wos predicts that in a few decades computers might be as agile at reasoning as they now are at calculating.

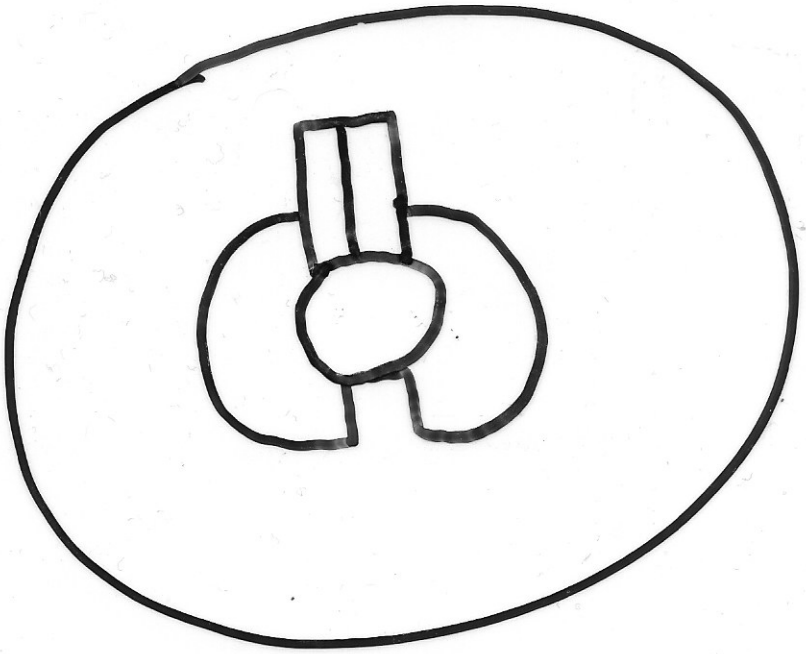
Anything New?

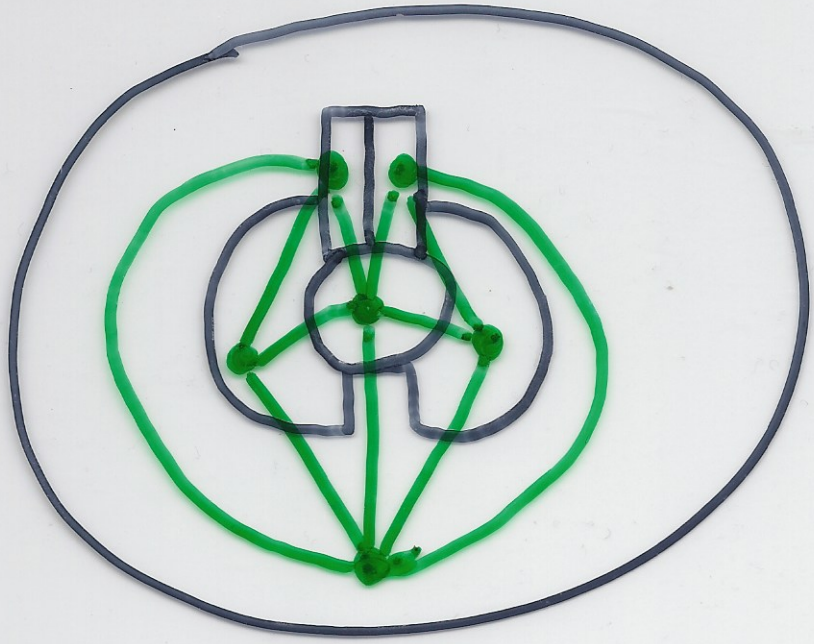
Computers had already had key roles in mathematical research

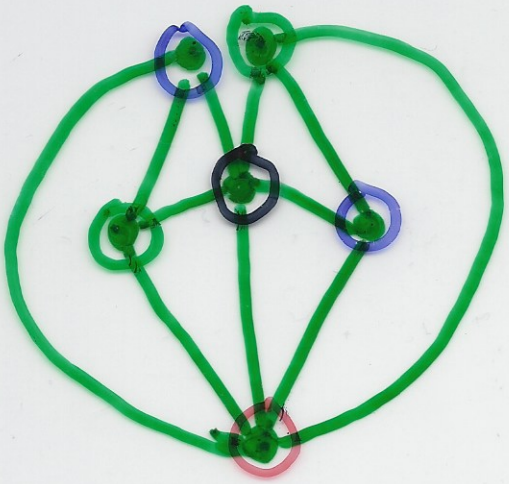
I) In 1976 the mathematicians K. Appel and W. Haken proved the *4 Colours Conjecture* by means of a computer program: *4 colours suffice to paint a map in such a way that any two adjacent countries have different colours*

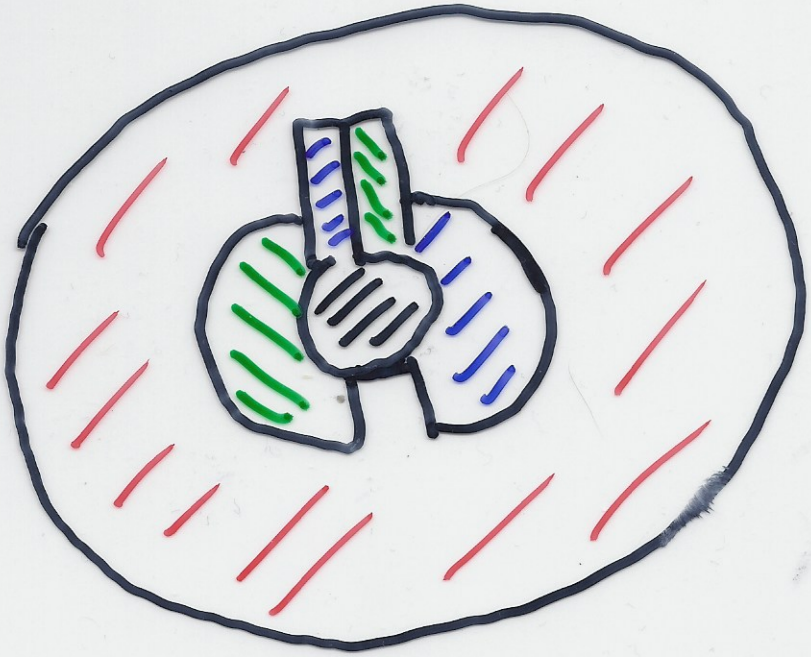
This conjecture was open for a long time (at least since 1850)

By the end of 19th century a proof was given, but after about 15 years it was found incorrect (what is a correct mathematical proof?)









The proof in 1976 reduced the problem to a combination of cases:

- If the conjecture was false, that would have to be reflected in some of 2000 possible cases
- Each of the cases was checked by computer and the "wrong" case did not occur

There was a combination of sophisticated mathematics and computer power

The proof was criticized by the mathematical community

In particular, what about correction of the program?

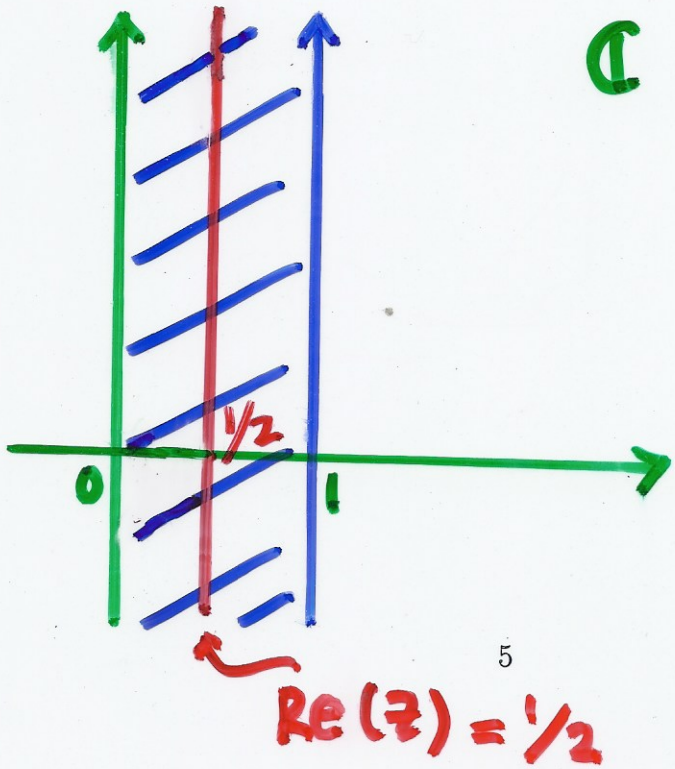
II) In the 80s an enormous number of cases associated to the an open conjecture in mathematics were checked

Riemann's Conjecture: *The roots of a certain function of a complex variable fall all on certain straight line of the complex plane*

$$\zeta(z) = \sum_{n=1}^{\infty} \frac{1}{n^z}$$

$0 \leq$ Real part of $z \leq 1$ and $\zeta(z) = 0 \Rightarrow$

Real part of $z = \frac{1}{2}$



With hours of supercomputing Riemann's hypothesis was positively verified for the first 10^9 zeros in the band

It was necessary to design and develop very efficient algorithms based on sophisticated mathematics

III) 1988: More than 400 computers distributed through the world used their spare time to factorize for the first time a number that had escaped previous attempts

the new was published and highlighted in the New York Times!!

Need to rethink cryptographic protocols that are based on the assumption that integer factorization is intractable

The New York Times

36 Copyright © 1988 The New York Times

WEDNESDAY, OCTOBER 12, 1988

The Question

9,412,343,807,369,262,946,971,172,136,294,514,357,528,981,379,583,092,541,347,532,211,542,640,121,301,690,098,634,689,611,468,911,681

The Answer

88,769,222,313,428,390,912,218,077,095,850,708,048,977 × 106,069,104,653,837,470,812,981,399,842,972,948,409,834,611,525,790,577,216,753

Source: Dr. Andrew M. Odlyzko, AT&T Bell Laboratories, Murray Hill, N.J.

Most Ferocious Math Problem Is Tamed

By MALCOLM W. BROWNE

By pooling together the output of hundreds of computers at their establishments, a team of mathematicians succeeded yesterday in solving a massive computational problem that had defied all previous efforts. The achievement is likely to force cryptographers to reassess the future application of some codes used by governments and banks.

At 2:01 A.M., Pacific daylight time, the last sequence of numbers required for the solution popped up in a computer laboratory in Palo Alto, Calif., and news of the triumph was flashed to collaborators around the world. The team had succeeded for the first time in splitting a number 100 digits long, into two large prime factors.

The factors of a number are smaller numbers which, when multiplied by each other, yield the larger number. A prime number is one that is evenly divisible only by one or by itself. The prime factors of 15, for example, are 3 and 5.

The two factors found for the 100-digit number, which was selected by an elaborate mathematical screening process to pose the maximum possible difficulty, are respectively 45 digits and 55 digits long.

The organizers of the project were Dr. Mark S. Manasse of the Digital Equipment Corporation's Systems Research Center in Palo Alto and Dr. Arjen K. Lenstra of the University of Chicago. But a dozen users of some 400 computers in the United States, the Netherlands and Australia were re-

Continued on Page 11, Column 2

Army of 400 Computers Tames Most Ferocious Math Problem

Continued From Page 1

cruited to join in the project, donating computing time from intervals when the computers were not needed for their regular work.

Several of the most secure cipher systems devised in the past decade are based on the fact that large numbers are extremely difficult to factor, even using the most powerful computers for long periods of time. The accomplishment of factoring a 100-digit number "is likely to prompt cryptographers to reconsider their assumptions about cipher security," Dr. Lenstra said in a telephone interview.

"We Have Upped the Ante"

His colleague, Dr. Manasse, added: "What this shows is that a cryptographer would avoid using a cipher on any factorable number smaller than about 200. The cipher system still works, but we have upped the ante." Using larger numbers makes the work of cryptographers more cumbersome and time-consuming.

One cryptographic system based on the difficulty of factoring large numbers was invented in 1977 by Ronald L. Rivest, Adi Shamir and Leonard Adleman of the Massachusetts Institute of Technology. The three mathematicians patented their system and incorporated computer chips especially designed to apply it. In this and similar systems, digits replace each letter in a text message, and the entire sequence of digits is treated as a single large number. A multiplication operation is then performed on this number, and to decipher the result requires either that the recipient possesses the key or breaks the code by factoring the large number.

This kind of cipher system is regarded as an elite and custom-made for such routine cipher messages as those used for secret government messages and the transfer of funds between banks. But according to Dr. Ron Graham, a mathematician at A.T.&T. Bell Laboratories, the "RSA System" (which takes its initials from the names of its inventors) is used freely

> **Cryptographers must reassess codes used by governments and banks.**

extensively for the secure transmission of encryption and decryption keys from one organization to another.

Guarding Secret Keys

Encryption and decryption keys must be protected more securely than any other secret message, because these are the keys that allow either the intended recipient of a cipher message or a spy to decipher it. Since such a key is relatively short, the expense of encrypting and decrypting it is a disadvantage more than offset by the protection it offers security.

The RSA method can accommodate factorable numbers of virtually any length as cipher keys, so yesterday's achievement does not compromise the method itself.

"Even if a user were to adopt a key 200 digits long," Dr. Manasse said, "it would take a gigantic organization working with single-minded dedication to accomplish what our powers has done. I doubt that any organization using this kind of cipher is seriously worried by what we have done."

Theoretically, a really large computer such as the Cray, operating continuously, could have solved the problem in about one week, an expert said. But renting such supercomputers costs many thousands of dollars an hour. By spreading many simpler computers, which they were able for a few minutes or hours, the problem was solved at virtually no cost.

Progress toward factoring a 100-digit number has been rapid this year, even though no new methods of factoring have been developed. Just three weeks ago, the same group led by Dr. Manasse and Dr. Lenstra established another record by factoring a number 90 digits long, and last spring, a Netherlands mathematician, Herman te Riele, factored 92 digits. In the 1980's the record has been broken about once a year, Dr. Lenstra said.

Technique Is Not New

Dr. Andrew M. Odlyzko, a mathematician at A.T.&T. Bell Laboratories in Murray Hill, N.J., said that the basic technique used for the calculation was not new.

The algorithm, or computer method, the team applied, "which is called the 'quadratic sieve,' was invented some years ago by Dr. Carl Pomerance of the University of Georgia at Athens," Dr. Odlyzko said. "In a way, it's slightly disappointing that we have not come up with any real advance over that method in factoring large numbers. There are a lot of variations of the same general method, but they are all roughly equal in efficiency."

The great advantage of the quadratic sieve is that it enables its user to break an extremely complex computational problem down into a large number of relatively simple computations. Each of these computations can be carried out independently and in any sequence.

The strategy, Dr. Lenstra explained, is to calculate a "matrix" of numbers, a square consisting of 50,000 rows and 50,000 columns, in each of which is a number that contributes some information about the solution of the problem. Ideally, a computer could be assigned to calculate each square in the matrix, but in this case, only about 400 computers were available.

"In any case, however," Dr. Manasse said, "we approached the problem as a massively parallel computation, one in which many processors were working along parallel lines simultaneously. As each small task was completed, the computer user would send in the result by electronic mail.

It Was All There

"Last weekend, we realized that we had enough of these preliminary computations to put them together, using a technique called Gaussian elimination. That gave us the final result that is the prime," he said.

Gaussian elimination is a procedure analogous to the elimination of variables in an algebraic expression by canceling out the values.

Dr. Manasse said that the factoring program had attracted the interest of mathematicians and computer scientists in Canada, West Germany and other countries, and that many had volunteered to participate in a continuation of the program.

Next: 104 Digits

The factoring of a 104-digit number took less than one month, and the pace is likely to continue. "By the end of the winter I would expect that we will have factored a number of 106 digits," Dr. Manasse said.

Asked whether mathematicians had signed new factoring techniques that might speed the process, he replied: "I'm not anxious to see any radically new techniques emerge. Of course, if they're out there, we'll have to use them. But I would like to see the RSA encryption system continue as a secure safeguard."

All encryption systems are under the continuous scrutiny of the National Security Agency, the Federal agency chiefly responsible for safeguarding American ciphers and cracking the ciphers of other nations. Experts from the agency attend meetings of mathematicians and computer scientists and follow developments in the field closely.

"It's a safe bet that the N.S.A. knows all about what we've done," Dr. Manasse said, "although we've had no communication from them. In any case, this is not going to give them nightmares. It may just make the code makers a little more cautious. We've done something that once would have been regarded as practically impossible."

The New

For the proof of the *Robbins' Conjecture* an "automated reasoner" was used: basically OTTER (Argonne National Laboratory)

The conjecture -now Theorem- was proven using a general purpose computational program, implemented to prove general theorems

Not designed to prove a theorem in particular

That is, like humans beings that use general reasoning capabilities to prove theorems

We could attribute "intelligence", "creativity", etc. to this program; characteristics usually applied to human beings

The mathematics involved in the proof of this theorem was being done by the computer, not by a human being supported by the brute force of a computer

See article by Larry Wos in Comm. ACM, 41, 6, 1998:

What Was Done?

The prover established, in essence, that the Equations for Robbins' Algebras:

$$x \vee y = y \vee x$$

$$(x \vee y) \vee z = x \vee (y \vee z)$$

$$\neg(\neg(x \vee y) \vee \neg(x \vee \neg y)) = x$$

Imply the Equations for Boolean Algebras:

$$x \vee y = y \vee x$$

$$(x \vee y) \vee z = x \vee (y \vee z)$$

$$\neg(\neg x \vee y) \vee \neg(\neg x \vee \neg y) = x$$

That is, the prover showed to be capable of performing equational reasoning and of doing proofs in the context of a mathematical theory

What is Needed?

(1) To know first what is a mathematical proof; at least to recognize one when done by a computer

- What are the admissible deductive steps in a mathematical proof?
- What is "the logic" behind deductive/mathematical reasoning?

Answer requires the study of human activity of mathematical reasoning

Requires converting it into a discipline subject to scientific/mathematical investigation

Mathematical Logic emerges as a mathematical discipline whose object of study is the logic of (human) mathematical reasoning

Mathematical logic now investigates other forms of reasoning (not only the one used in mathematics)

(2) Requires expressing the logic in terms that can be represented and processed in/by a computer

- In symbolic terms
- By means of mechanical and deterministic processes

  That transform symbolic representations into new symbolic representations

(3) Requires establishing results that ensure that the symbolic processes that are supposed to capture the logic of mathematical reasoning are *strong enough* and *can be trusted* (K. Goedel, 1930)

- Everything that is proven through the symbolic processes is indeed something that is a mathematical consequence in the usual sense

- Furthermore and hopefully: that every theorem in the usual sense is provable in the symbolic sense

  (The symbolic proof may not be known, but that is another problem ...)

(4) To implement the symbolic proof mechanisms in a computer it is necessary to design:

- Fast and simple deductive *steps*

- Strategies, heuristics, to combine, choose, search, and guide the deductive steps, in order to obtain complete proofs

  (human being are also confronted to alternative steps when they do proofs)

A Scientific Problem?

All we have described so far is basically a scientific problem; it involves

- A scientific study of the observable phenomenon of mathematical reasoning

- A mathematical formalization and modeling, in this case symbolic, of certain mathematical processes

  With the purpose of making them "computable"

- A study of the relationship between the original phenomenon and its symbolic/computational counterpart

- A computational implementation of the symbolic formalism

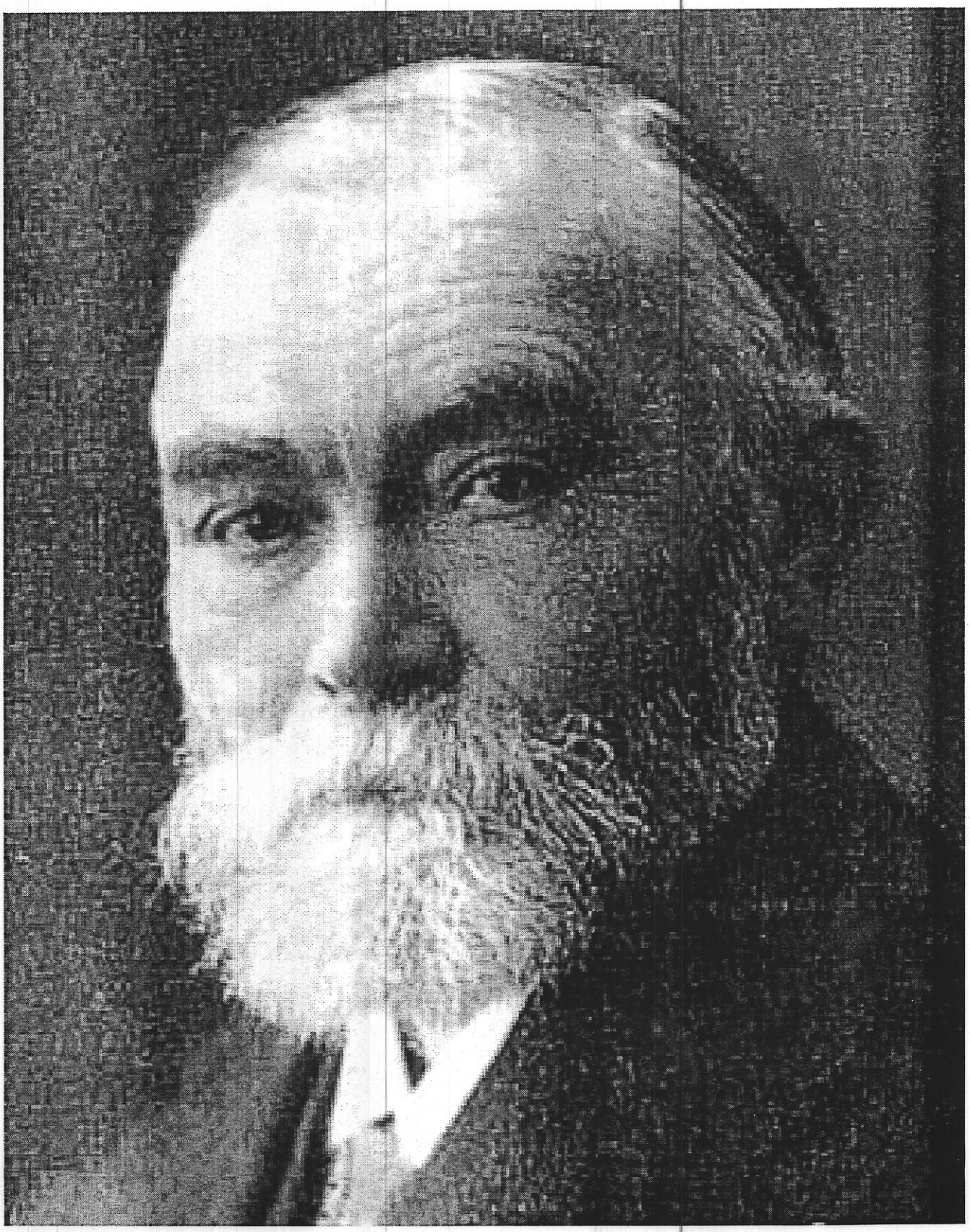
Some new fundamental, relevant and natural questions arise:

- To what extent and in what sense can we use symbolic representations and symbolic processing to model and implement mental activities associated to human intelligence?
- What are the limits of what is computable?
- What are the limits of what is *practically* computable?
- What is a computer?

A science of the computable ...

A science of certain aspects of intelligence ...

There has been mathematical logic at least since the work by G. Boole, G. Frege, B. Russell, D. Hilbert, A. Tarski, K. Goedel, ...



There has been AI since the late 50s

There has been computer science, for a long time, explicitly since the mid 30s

Some Scientific Landmarks

1. The idea of solving problems in an algorithmic manner can be traced back to antiquity

   Remember arithmetic, Euclid, greatest common divisor, ...

2. The idea of mechanizing logical processes can be found already in the Middle Age

   There was Aristotle's logic much before that

   Mathematical logic as known today starts in the 19th century

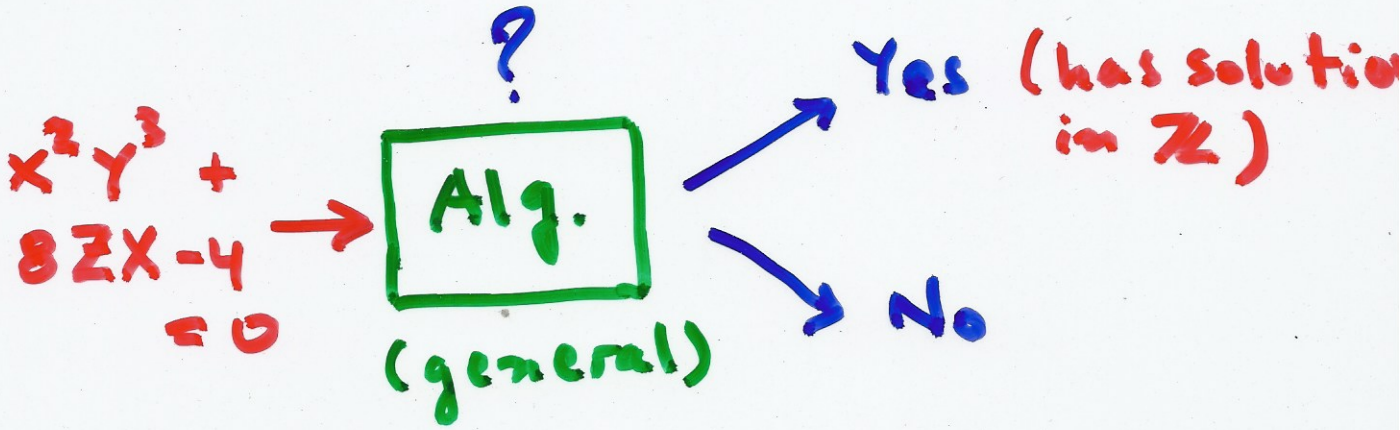
3. In 1900 problems about the solvability by algorithmic means of certain algebraic problems are explicitly formulated

David Hilbert: 10th problem of his list of 23 open mathematical problems (keynote presentation at International Congress of Mathematics, Paris, 1900)

Is there a mechanical procedure to decide if an arbitrary diophantine equation has a solution in the integers?

E.g. $X^2Y^3 + 8ZX - 4 = 0$

(a polynomial equation with multiple variables and integer coefficients)



Notice:

- If such an algorithm would have been found, it would have been good enough to exhibit it to the mathematical community, who would have recognized it as such

  If mathematically correct and with the intuitive characteristics of an algorithm ...

- If it would not have existed, how to prove that?

  This amounts to prove that "There is no algorithm that ..."

  To prove something of this kind we need to have a mathematical characterization of the intuitive notion of algorithm

There was no such mathematical definition in 1900

No way to answer negatively ...

4. In the early 1930s the first mathematical models of (an ideal) computer, algorithm, computable function, (computationally) decidable problem, ... appear

   - Turing machines (Alan Turing)
   - Recursive functions (Kurt Goedel, Steve Kleene)
   - Lambda calculus (Alonso Church)
   - Post systems (Emil Post)
   - ...

HW

δ	e_2	δ	$+1$	e_4
1	e_2	1	$+1$	e_2
$\#$	e_6	a	-1	e_0

SW

Different mathematical characterizations of the same phenomenon, from different intuitions and perspectives, but ...

It could be mathematically proved that they all lead to the same class of computable functions!

Each model can be simulated by means of any other!

A very interesting moment in history of science!!

Computer science is born as mathematical discipline!!

Before the inception of modern, practical computers ...

5. One starts (and only then) to **prove** that certain problems -of computational nature- are not soluble by a computer, e.g.

- There is no algorithm to decide if another algorithm will stop (Turing's halting problem)

- There is no algorithm to decide if a logical formula with predicates, variables, quantifiers is always true (valid) (A. Church)

- Hilbert's 10th Problem is undecidable, i.e. there is no algorithm ... (M. Davis, D. Putnam, J. Robinson, and finally, Y. Matiyasevich in 1970)

- ...

6. Moving from the qualitative to the quantitative, people start investigating the complexity of computational problems and of specific algorithms (mainly in the 70s)

   Some hard computational problems are exhibited

   Hard in the sense that from their efficient solvability (unknown yet) the efficient solvability of a vast class of many other problems depends

   A class that contains thousands of computational problems of practical and theoretical importance for which is it not known today if they can be solved by means of *deterministic algorithms in polynomial time*, i.e. if they belong to the class P

Example: (Steve Cook^{TA}, 1971) Satisfiability of an arbitrary propositional formula, e.g.

$$(p \vee q \vee \neg r) \wedge (\neg p \vee q \vee s) \wedge \cdots (\neg q \vee r \vee t)$$

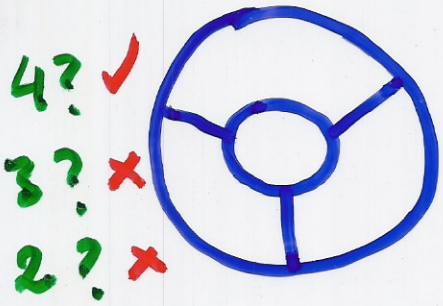
"Is there a assignment of 0*s and* 1*s to the propositional variables in the formula that makes it true?"*

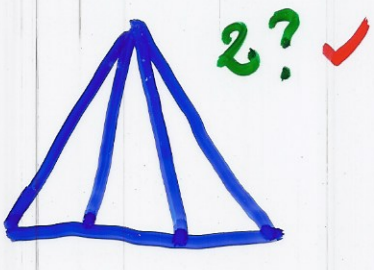
No known if $SAT \in P$, and hard in the sense above

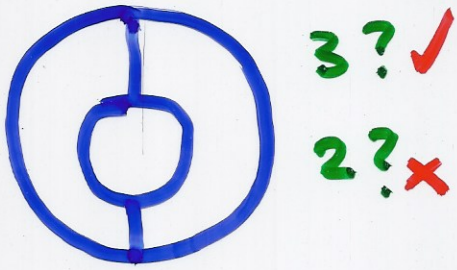
Example: (Richard Karp^{TA}, 1972) (by means of efficient reduction between problems):

- 3-Colorability: Given a graph, decide if it can be colored with three colours

  Nodes connected by an edge must have different colors

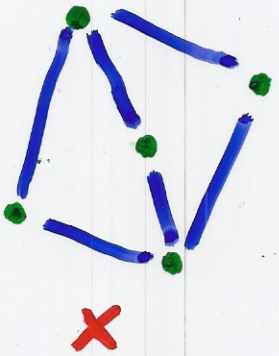




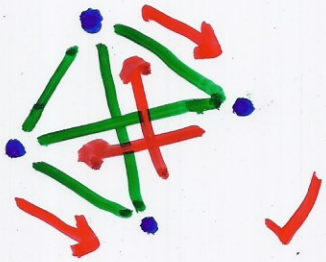


Not known if $3C \in P$, and ...

- Hamiltonian Circuit: Given a graph, decide if there is a simple tour that passes through all the nodes



Not known if $HC \in P$, and ...



If one of these three problems falls into the class P, the other 2 do as well; and thousands of other problems whose complexity status is not known

If one of these problems is proven to be outside the class P, i.e. it is intractable, then the same happens to the other two, and to thousands ...

7. Problems are exhibited that are provably intractable

Example: (Michael Fisher, Michael Rabin^{TA}, 1974)

The problem of deciding if an arbitrary logical formula about the real numbers is true wrt the theory of the real numbers (is solvable, but) requires in the worst case exponentially many steps in the size of the formula (and this for any decision algorithm)

$\forall x \forall y \exists z (x+y=z \cdot z$
$\wedge \ldots)$

Conclusions

- A computer can do *and discover* non trivial mathematical proofs with general purpose automated theorem provers

- Computer science is a real science, with firm mathematical and logical basis

- There is a rich interaction between mathematics and computing; each of these disciplines motivating and giving feedback to the other

  A creative and dynamic symbiosis, like the relationship between physics and mathematics early in the 20th century

- Mathematical logic emerges with a set of languages, concepts, tools, and methods, that can be used in computing

On the other side, most of the research in mathematical logic today is motivated by problems originated in computing

- There is also a rich interaction between computing and other scientific disciplines
  - Physics: Quantum computing; chaos, complex systems, ...
  - Biology: Biological computing, bioinformatics, computing in molecular biology (genomics)
  - Cognitive Science: Mental models, perception, computational linguistics, computational models of the brain, ...
  - Sociology, Economy: Modeling and simulation of societies, organizations, with emergent behaviour from agents interactions
  - ...

And yet a lot of fun things to do and discover!!!