COMPLEXITY OF COMPUTATION

M.O. Rabin

Rapporteurs: Dr. I. Mitrani
Mr. P. White
Mr. D. Wyeth

The first of Dr. Rabin's lectures was devoted to the assessment of the complexity of several combinatorial problems. The main question of interest in each case is whether the problem is of polynomial, super-polynomial or exponential complexity. That, however, is an open question for a large number of problems.

Examples:

1) The Graph Isomorphism problem: Given two graphs Γ_1 and Γ_2, is there a mapping φ of the Γ_1 vertices onto a subset of the Γ_2 vertices such that the images of two connected vertices are connected?

2) The Map Colouring problem: Given a graph Γ and a number of colours, can the vertices of Γ be coloured in such a way that no two connected vertices have the same colour?

3) The Satisfiability problem of propositional logic: Given a logical expression $A(p_1, p_2, \ldots, p_k)$, can a truth assignment of the propositions $p_1, p_2, \ldots p_k$ be found which will make A true?

In general, any problem of this nature can be reduced to one of deciding whether a given set of objects has a given property. The notion of a problem can thus be formalised as follows.

Let P be a set of bit strings. Decide, for any given bit string w, whether $w \in P$ or not.

The number of bits in w is called the size of the problem. If AL is an algorithm for solving the problem, the complexity $F_{AL}(n)$ of AL is defined as the maximum number of steps which AL takes to solve problems of size n. If there exist constants C and k such that

$$F_{AL}(n) \leq C n^k$$

for all n, then the problem is said to be of polynomial complexity. If no such constants exist, then the problem is of superpolynomial complexity. If there exists a constant C and a problem size n_o such that

$$F_{AL}(n) \geq C^n$$

for all $n \geq n_0$, then the problem is of exponential complexity. Of course the order of complexity may change depending on the automaton which solves the problem, but for the purpose of distinguishing between polynomial and exponential complexity we may assume that the problem is being solved by a Turing machine.

Definition: A problem P polynomially reduces to a problem Q if there is a Turing Machine calculable map $\varphi : \{0,1\}^* \to \{0,1\}^*$ such that $\varphi(w) \in Q$ if, and only if, $w \in P$ and furthermore the time to calculate $\varphi(w)$ is a polynomial function of $|w|$.

It is clear that if P is polynomially reducible to Q and the complexity of Q is polynomial, then the complexity of P is polynomial. Conversely, if the complexity of Q is not polynomial, then that of P is not polynomial.

The notion of polynomial reducibility leads to that of polynomial completeness, defined as follows:

A problem P is polynomially complete if

1) P is solvable in polynomial time on a non-deterministic Turing Machine.

2) Every problem which is solvable is polynomial time on a non-deterministic Turing Machine is polynomially reducible to P.

A non-deterministic Turing Machine is one for which at every step of the computation (defined by a state S and symbol under the head σ) there are a number of options of the type: print σ, move X, go to S. All halting computations must produce the same answer. The time that the non-deterministic Turing Machine takes to solve the problem is defined as the time of the shortest computation.

The remainder of the lecture was taken up by an outline of the proof of the following result, due to S. Cook:

Theorem The propositional satisfiability problem is polynomially complete.

First one has to show that the problem can be solved by a non-deterministic Turing Machine in polynomial time. This follows from the fact that a possible computation is the one which guesses an

assignment of truth values for the propositions and then verifies (deterministically) that the expression is satisfied. The verification can be done in about n^3 steps.

Second, every problem which can be solved by a non-deterministic Turing Machine in polynomial time must be shown to be polynomially reducible to the propositional satisfiability problem. This is done as follows. Suppose that T is a non-deterministic Turing Machine which accepts a set of strings S in polynomial time. For every input string w one can construct a propositional expression A (in conjunctive normal form) such that A is satisfiable if, and only if, $w \in S$. Furthermore, the size of A is a polynomial function of the size of w.

The idea of the construction is that A is a conjunction of sub-expressions, each making an assertion about the computation. All assertions are true if, and only if, $w \in S$. Furthermore, since the length of the computation is a polynomial function of $|w|$, the number of squares scanned, and hence the length of each sub-expression, is a polynomial function of $|w|$.

In the course of his second lecture Dr. Rabin proved that several very important and interesting problems were polynomially complete. The first result illustrated was in the form of the following theorem.

Theorem Satisfiability of proposition formulae is polynomially reducible to the satisfiability of proposition formulae which are conjunctions of disjunctions of at most three literals, (where by a literal we mean either a propositional variable or its negation).

The proof of this theorem is illustrated by example. Suppose we have a proposition formula and to each of the logical symbols we associate new proposition variables. Equivalences can then be set up so it can readily be seen that, by correctly choosing the new proposition variables, the original formula is reducible to a proposition formula which is a conjunction of at most three literals.

The next topic concerns the concept of restricted colouring problems and in these restricted problems there is no graph, merely a set of n vertices and three colours. A colouring means that each of the vertices will be assigned a colour and a restriction is a condition

of the form that the pair (i,j) of vertices shall not be coloured by one of these colours. The map colouring problem is of course a special case of this.

Theorem A restricted colouring problem is reducible to the ordinary map colouring problem, and from the computational point of view they are equivalent.

The next step is to show that satisfiability of special formulae, to which satisfiability of all formulae is reducible, is not just reducible to but is a restricted colouring problem.

Consider the special formula which is a conjunction of small formulae each a disjunction of at most three variables. Since the conjunction is true if and only if each of the conjuncts is true, we say that if the second term is true then it is coloured by b and if the third term is true then it is coloured by c. So it is a question of colouring the terms or nodes, and for that to be a truth value assignment there are no restrictions. And thus from such a formula, interpreting each of the conjuncts as a vertex, certain restrictions are obtained and this is a restricted colouring problem. This problem if solvable implies that the formula can be made true, otherwise the formula cannot be made true. Hence satisfiability of special formulae is a restricted colouring problem and we obtain the following result.

Theorem The map three-colouring problem is polynomially complete.

This is because the propositional satisfiability problem is polynomially complete and is reducible in polynomial time to satisfiability of special formulae, which is a restricted colouring problem, and every restricted colouring problem is reducible to a map colouring problem which involves only three colours.

The next topic concerns n-cliques.

An n-clique is a graph Γ_n on n vertices, every two of which are connected.

Now the clique problem is the computational problem that the graph Γ contain an n-clique as a sub-graph. Alternatively it is an isomorphism problem: is Γ_n isomorphically embeddable in Γ?

This clique problem is polynomially complete and this can be shown by observing that the special propositional formulae satisfiability problem is a clique problem.

Summing up so far we have the problem:

Given a set of strings, $B \subseteq \{0,1\}^*$, is $w \in B$ for given w's?

We now introduce the following notation:

P is the class of all problems which are solvable by deterministic algorithms in polynomial time;

NP is the class of all problems which are solvable by (possibly) non-deterministic algorithms in polynomial time.

We can assert

$$B \in P \Rightarrow \{0,1\}^* - B \in P.$$

However this is not true, on the face of it, for problems which can be solved by non-deterministic algorithms.

There are now the following three cardinal questions.

1) Is P = NP? Either of the two answers would be interesting. If this is true then this would mean that every algorithm which is non-deterministic and terminating in polynomial time can be reduced to a deterministic algorithm terminating in polynomial time. If however, $P \neq NP$, as is currently suspected, then the complexity of the solution to these problems is not bounded by n^k for any k.

2) Is NP closed under complementation? If this is false then of course you have proved that $P \neq NP$.

3) Does NP contain problems of truly exponential complexity? So here is an example of a situation where very original, but rather simple, ideas shed a completely new light on problems which people were trying to solve over the years. Also this gives a caution signal that there may be an inherent exponential complexity with seemingly very simple combinatorial problems.

The next chapter in the lecture discusses a class of results obtained by Michael Fisher and Dr. Rabin where there are definite answers with respect to the complexity of the decision problems in question.

First a discussion of the theories of addition of natural numbers and the addition of real numbers.

Let

$N = \{0,1,2..\}$, the set of all natural numbers including zero,

R = the set of all real numbers.

Starting with the two structures which are commutative semi-groups

$$\mathcal{N} = \langle N, + \rangle \text{ and } \mathfrak{R} = \langle R, + \rangle$$

we then consider elementary statements about addition which can be expressed by using first-order predicate logic. Taking a formalism with the quantifiers $\forall$, the logical symbols, and the non-logical symbol +, we have the idea of sentences $\sigma\ \forall x\ \forall y\ [x+y=y+x]$ and $\tau\ \forall x\ \forall z\ \exists y\ [x+y=z]$. Now the first sentence is true for natural numbers and this is written

$$\mathcal{N} \models \sigma \quad \text{(where } \models \text{ means 'is true in')}$$

$$\text{Also } \mathcal{N} \models \sigma, \neg\tau \text{ and } \mathfrak{R} \models \sigma, \tau.$$

Next we introduce the notation $Th(\mathcal{N})$ for the theory of addition of natural numbers, and we say $Th(\mathcal{N}) \triangleq \{\sigma | \mathcal{N} \models \sigma\}$ ($\triangleq$ means 'is by definition').

Presburger (c. 1930) proved that $Th(\mathcal{N})$ is decidable; $Th(\mathcal{N})$ is called Presburger's Arithmetic, denoted by PA.

Tarski (c. 1929) had shown that the theory of addition and multiplication of real numbers $Th(\langle R,+,x\rangle)$ is decidable; this is Tarski's Arithmetic, TA.

Theorem There exists a $c>0$ such that for every algorithm, AL, for deciding PA there exists an n_0 (n_0 depends upon the algorithm) such that for every $n \geq n_0$ there exists a sentence σ, $l(\sigma)=n$ so that AL takes more than $2^{2^{cn}}$ steps to decide σ, where $l(\sigma)$ denotes the length of σ.

Thus every algorithm from a certain point is doubly exponentially bad. Two questions arise as a result of this; How big is n_0? How small is c? The quantity c depends on the formalism and so a formalism can be devised with certain shorthands which has the effect of increasing the size of c until c becomes about 0.05. It then turns out that

super-exponential explosion sets in with formulae which are of the order of the size of the algorithm.

Theorem let AX be a system of axioms for PA such that $\sigma \in$ AX is decidable in polynomial time. Given n_0 and $d>0$ for every $n \geq n_0$ there exists a true theorem, $w \in$ PA, $l(w)=n$, whose shortest proof (from AX) is longer than $2^{2^{dn}}$.

So the trouble is not just that we wish to solve this algorithmically in a deterministic way and this makes our procedures very long, but even if we introduce heuristics, instead of algorithms, it isn't an answer because the proofs to be produced will be so long that for modest n the size will exhaust the physical size of the universe.

Similar results hold for TA except that the lower bound is exponential not doubly exponential.

In the third lecture, Dr. Rabin gave a brief outline of how the results concerning the exponential complexity of the theory of addition of real numbers and the super-exponential complexity of the addition of natural numbers are proved.

Suppose there is an additive semigroup $\langle H,+\rangle$ which contains the natural numbers as a subsystem, for example the real numbers. Only addition is considered; however it is possible to reproduce a multiplication table up to a very large number using a short formula involving just addition. The following key lemma is a precise statement of this.

Lemma

Let $\langle H,+\rangle \supseteq \langle N,+\rangle$ (e.g. $\langle H,+\rangle = \mathbb{R}$). There exists a $d>0$ so that for every n there is a formula $Mn(x,y,z)$, $l(Mn) \leq dn$, satisfying $a,b,c, \in, H$ $\langle H,+\rangle \models Mn(a,b,c)$ if and only if $a \in N$, $a \leq 2^{2^n}$, $a.b=c$, where l denotes length.

The size of the formula $Mn(x,y,z)$ is bounded by dn, and $Mn(x,y,z)$ is satisfied by the triple a,b,c if and only if a is a natural number less than or equal to 2^{2^n} and $a.b=c$. It is necessary to define a.b. Since a is a natural number, a.b is by definition

$$a.b \triangleq \underbrace{b+b+\ldots+b}_{\text{sum of a b's}}$$

and

$$0.b \stackrel{d}{=} 0.$$

Thus a short formula of size up to dn written just in terms of addition, expresses multiplication by a, where a is a variable integer assuming values between 0 and 2^{2^n}. In particular, the following result follows

$$Mn(a,0,0) = a \in N \quad a \leq 2^{2^n},$$

that is $a.0=0$.

Put in another way, a short formula of size dn can single out from among the real numbers, integers up to 2^{2^n}. This in itself is a non-trivial fact, because it is not difficult to define integers up to 2^n using systems of equations, but to extend it to integers up to 2^{2^n} is **rather hard.**

Once we have integers up to 2^{2^n} and their multiplication table, it is possible to code arbitrary 0,1 sequences up to size 2^n in terms of unique integers up to 2^{2^n} (since the number of 0,1 sequences of size up to 2^n is 2^{2^n}). The trick is to write a number m as

$$m = e_0 + e_1 .2 + e_2 .2^2 + \ldots + e_k 2^k \quad \text{where } K \sim \log_2 n.$$

So m corresponds to the sequence $(e_0, e_1, e_2, \ldots, e_k)$. Since the number m ranges between 0 and 2^{2^n}, these sequences range over all 0,1 sequences of length 2^n.

Since it is possible to code 0,1 sequences of length up to 2^n, it means that one can talk about non-deterministic computations terminating in exponential time. Using standard Gödel type methods, one can diagonalise over these computations and it then follows that every decision procedure (for the addition of real numbers) should take at least 2^{cn} steps, where c is a constant.

This result applies essentially to every group; the reals, the complex numbers and therefore the field of complex numbers. In the case of a general field, instead of considering the additive group, the multiplicative group is used. An element a is fixed as a parameter and $a^0 a^1 a^2 \ldots$ used to code the sequences. Thus, as before, one can

recapture the integers. Hence, it turns out that even for decidable fields the complexity is always at least 2^{cn}.

The complexity of decision procedures in the addition of natural numbers can be deduced from the following lemma.

Lemma For $\langle N,+\rangle$ there exists $d>0$ so that for every n there is a formula $Pn(x,y,z)$, $l(Pn)\leq dn$ for which $a,b,c \in N$, and $\langle N,+\rangle \models Pn(a,b,c)$ if and only $0\leq a,b,c \leq 2^{2^{2n}}$ and $a.b=c$.

Thus a formula of size proportional to n codes the multiplication table up to the number $2^{2^{2n}}$. Employing the same argument as before, it is possible to code sequences of length up to 2^{2^n} and thus the resulting complexity is $2^{2^{cn}}$.

Gödel and Church, by using both addition and multiplication, obtain undecidability results. Since $\langle N,+\rangle$ is decidable by Pressburger's result, we can never hope to define multiplication just by a formula involving addition, but we can define extremely large chunks of the multiplication table by short formulae and this gives rise to the practical undecidability results, which can be seen as chasing after the results of Gödel and Church.

These results point to the moral of this work. There has been a succession of developments concerning the way we view languages. First, we started from natural language as translated into mathematics, a sort of naive set theory, then we used the language in an unrestricted fashion to write arbitrary expressions and sentences. This lead to the paradox of the set of all sets which is not an element of itself, so these arbitrary constructions can be contradictory. The next development was to formalise the system by introducing axioms. It is known that it cannot be proved that one has a consistent system, but one hopes the system is consistent. However, these general languages, like for example a fragment of the theory of addition and multiplication of natural numbers, are still strong enough to enable one to write extremely difficult sentences, sentences which are independent of the axioms. Thus one has an incomplete system, the language is too strong. One can then attempt to restrict oneself to certain very limited fragments, for example $\langle N,+\rangle$, which are decidable. Axioms can be written for $\langle N,+\rangle$ from which the sentences which are true about the

addition of natural numbers can be derived. However, the formal languages still present the problem that they are an extremely compact means of expressing statments. As we have shown, in the language of addition of natural numbers there are hidden large chunks of the multiplication table and this means that one can talk about almost arbitrary computations which are 2^{2^n} big by means of sentences which are of length dn. These languages are very powerful and one can express in them statements which though decidable are so complicated that their shortest proof from the axioms is of size 2^{2^n}. It appears to be a fundamental difficulty that one is destined not to know the true answer to many statements.

It is perhaps appropriate to make some comments on the method of proof, though these comments are completely in the realm of conjecture. We have one method of proof, for which the shortest proof of a sentence is essentially of size 2^{2^n}. No better methods are known, but it is conjectured that these practically undecidable statements are very 'abundant'. The term 'abundant' cannot be defined because there is no measure over the space of statements. Such a measure should not be based on syntactic structure but on semantics, since it is the meaning of sentences that determines whether a sentence is frequent or rare not how it is constructed syntactically. Assuming a measure existed, it is not unlikely that these extremely difficult sentences are very abundant, if not the rule. Consider an analogy with number theory. One can easily stump the experts by asking a question such as 'are there infinitely many primes of the form X^2+1?' When that problem has been solved, the question 'are there infinitely many primes of the form X^3+X^2+17?' can be asked. One can go on and ask many similar questions. It would seem, and this is not based on any information or proof, that this ability to create arbitrary sentences just leads to sentences which are practically beyond our methods of proof. The reason why we are able to prove theorems, and there are books full of them, is that we, in a very true sense, only prove those theorems which we can prove. So by a process of cultural intellectual revolution, we always make epsilon additions and discover certain territories in which we can operate. Outside these territories, there are other territories where the shortest proofs are of sizes beyond

cosmological dimensions and where we will necessarily always be ignorant.

Discussion

Professor Scott asked if it is possible to have an inconsistent system in which the inconsistency was so difficult to prove that it was never found, but if the system was used with ordinary proofs it would still be possible to solve problems with it. Dr. Rabin replied that this was definitely the case. If one has a system in which the shortest proof of a contradiction is of size say $2^{2^{200}}$, then one can work with impunity in such a system. Once one has a contradiction, one can derive 0=1 as usual. This ties in with another comment on proofs. One possible way of shortening the very long computations by algorithms is to permit the existence of errors. An artificial example is that if one wants to be 100% certain of the correctness of a computation to determine whether or not a fifty-digit number is prime, then it will take about 2^{50} steps. But if one is willing to allow a very small margin of error then the time of the computation essentially reduces to linear time in the size of the data. This possibility has not been considered in detail, but perhaps should be because in artificial intelligence when we want to emulate human intelligence, errors are acceptable since they are part of the human condition.