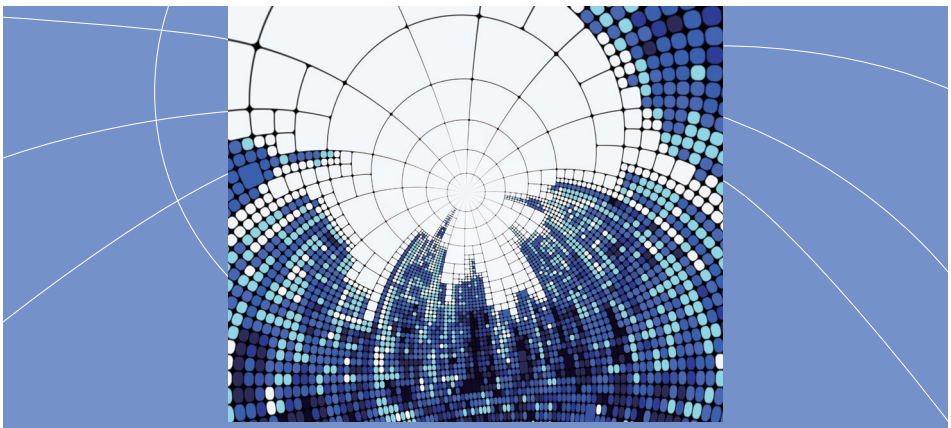


计算复杂性 50 年：王浩与计算理论

尼 克

1. 引言

如果说图灵 1936 年那篇开天辟地的文章[1]奠定了计算理论的基础，那么说库克（Stephen Cook）1971 年的文章《定理证明过程的复杂性》[2]是计算复杂性的开山之作，一点也不夸张。从库克文章发表日开始算，计算复杂性理论 50 岁了（2021 年）。2021 年又是库克的导师王浩冥诞 100 年。本文回溯计算复杂性的起源，并力图梳理王浩和这门学科的关系。

2. 计算复杂性的源头

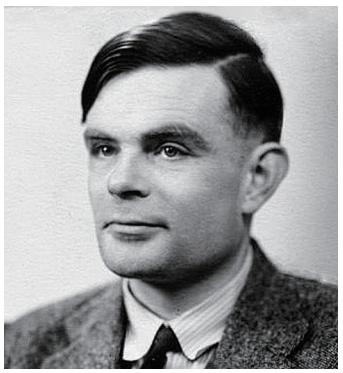
计算复杂性的基本概念的形成可以归功于科巴姆（Alan Cobham）、哈特马尼斯（Juris Hartmanis）与斯特恩斯（Richard Edwin Stearns）。科巴姆于 1964 年发表了文章《函数的内在计算难度》[3]，哈特马尼斯与斯特恩斯 1965 年在《美国数学会会刊》上发表了《论算法的计算复杂性》[4]。这两篇文章定义了计算的时间和空间复杂性，科巴姆甚至探讨了用多项式时间作为有效计

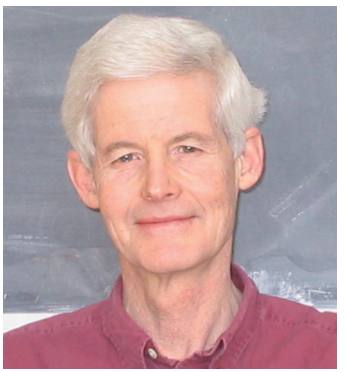
[1] Turing，A.M. (1936-1937), "On Computable Numbers, with an Application to the Entscheidungsproblem," *Proceedings of the London Mathematical Society*. 42. pp. 230–65.

[2] Cook, Stephen (1971). "The complexity of theorem proving procedures". *Proceedings of the Third Annual ACM Symposium on Theory of Computing*. pp. 151–158.

[3] Cobham, Alan (1965). "The intrinsic computational difficulty of functions". In Bar-Hillel, Yehoshua (ed.). *Logic, Methodology and Philosophy of Science: Proceedings of the 1964 International Congress*. Studies in Logic and the Foundations of Mathematics. Amsterdam: North-Holland. pp. 24–30.

[4] Hartmanis, J.; Stearns, R. E. (1965), "On the computational complexity of algorithms", *Transactions of the American Mathematical Society*, 117: 285–306.





230 A. M. TURING [Nov. 12,

ON COMPUTABLE NUMBERS, WITH AN APPLICATION TO THE ENTSCHEIDUNGSPROBLEM

By A. M. TURING.

[Received 28 May, 1936.—Read 12 November, 1936.]

The "computable" numbers may be described briefly as the real numbers whose expressions as a decimal are calculable by finite means. Although the subject of this paper is ostensibly the computable numbers, it is almost equally easy to define and investigate computable functions of an integral variable or a real or computable variable, computable predicates, and so forth. The fundamental problems involved are, however, the same in each case, and I have chosen the computable numbers for explicit treatment as involving the least cumbrous technique. I hope shortly to give an account of the relations of the computable numbers, functions, and so forth to one another. This will include a development of the theory of functions of a real variable expressed in terms of computable numbers. According to my definition, a number is computable if its decimal can be written down by a machine.

In §§ 9, 10 I give some arguments with the intention of showing that the computable numbers include all numbers which could naturally be regarded as computable. In particular, I show that certain large classes of numbers are computable. They include, for instance, the real parts of all algebraic numbers, the real parts of the zeros of the Bessel functions, the numbers π, e, etc. The computable numbers do not, however, include all definable numbers, and an example is given of a definable number which is not computable.

Although the class of computable numbers is so great, and in many ways similar to the class of real numbers, it is nevertheless enumerable. In § 8 I examine certain arguments which would seem to prove the contrary. By the correct application of one of these arguments, conclusions are reached which are superficially similar to those of Gödel†. These results

† Gödel, "Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme, I", *Monatshefte Math. Phys.*, 38 (1931), 173–198.

11. DOMINOES AND THE AEA CASE OF THE DECISION PROBLEM*

It has recently been established that the AEA case is unsolvable and forms a reduction class. Several people have looked into possible directions along which the result can be strengthened, using in part earlier methods developed by Büchi for the F & AEA case. There are three different aspects. First, unsolvable AEA subcases such as restrictions on the number of dyadic predicates, on the form of the quantifier free component, on the complexity of the models (e.g., finite, essentially periodic, etc.). Second, solvable AEA subcases. Third, the detailed structure of the reduction of the general case to the AEA case. A survey of these questions is presented.

I. INTRODUCTION AND SUMMARY

Since all mathematical theories can be formulated within the framework of the predicate calculus (quantification theory, elementary logic), Hilbert spoke of "the" decision problem when he was referring to the problem of finding a general algorithm to decide, for each given formula of the predicate calculus, whether it is satisfiable in some non-empty domain (or, has a model). He called this the main problem of mathematical logic. It is familiar today that this problem in its general form is unsolvable in a technical sense which is widely accepted as implying unsolvability according to the intuitive meaning. An interesting problem is to investigate the limits of decidable subdomains and the underlying reasons of the phenomenon of undecidability.

Recently, the general problem has been reduced to the formally simple case of formulas of the form $AxEuAyMxyu$, where M is quantifier-free and contains neither the equality sign nor function symbols.[4] It is, therefore, of special interest to study the AEA case in greater detail. Moreover, the simplicity of the AEA prefix makes it possible to confine our attention to the essential problems without distraction by extraneous complexities.

The first published attempt to settle the AEA case appeared in a paper by the author (reference 14, pp. 23–32), in which steps toward a positive solution were reported, and a domino problem (the unrestricted domino problem) was formulated in connection with a procedure for deciding some AEA formulas. During the spring of 1960, when this part was written, it was also shown that the origin-constrained domino problem is unsolvable, although the result was not included in this part.

Immediately after the appearance of reference 14, Büchi studied the domino

* First published in *Mathematical Theory of Automata*, pp. 23–35. Polytechnic Press, 1963. Reproduced by permission of the author.

210

Stephen A. Cook

University of Toronto

Summary

It is shown that any recognition problem solved by a polynomial time-bounded nondeterministic Turing machine can be "reduced" to the problem of determining whether a given propositional formula is a tautology. Here "reduced" means, roughly speaking, that the first problem can be solved deterministically in polynomial time provided an oracle is available for solving the second. From this notion of reducible, polynomial degrees of difficulty are defined, and it is shown that the problem of determining tautologyhood has the same polynomial degree as the problem of determining whether the first of two given graphs is isomorphic to a subgraph of the second. Other examples are discussed. A method of measuring the complexity of proof procedures for the predicate calculus is introduced and discussed.

Throughout this paper, a set of strings means a set of strings on some fixed, large, finite alphabet Σ. This alphabet is large enough to include symbols for all sets described here. All Turing machines are deterministic recognition devices, unless the contrary is explicitly stated.

1. Tautologies and Polynomial Reducibility.

Let us fix a formalism for the propositional calculus in which formulas are written as strings on Σ. Since we will require infinitely many propositional symbols (atoms), each such symbol will consist of a number of Σ's followed by a number in binary notation to distinguish that symbol. Thus a formula of length n can only have about $n/\log n$ distinct function and predicate symbols. The logical connectives are $\&$ (and), $\vee$ (or), and $\neg$(not).

The set of tautologies (denoted by {tautologies}) is a

certain recursive set of strings on this alphabet, and we are interested in the problem of finding a good lower bound on its possible recognition times. We provide no such lower bound here, but Theorem 1 will give evidence that {tautologies} is a difficult set to recognize, since many apparently difficult problems can be reduced to determining tautologyhood. By reduced we mean, roughly speaking, that if tautologyhood could be decided instantly (by an "oracle") then those problems could be decided in polynomial time. In order to make this notion precise, we introduce query machines, which are like Turing machines with oracles in [1].

A query machine is a multitape Turing machine with a distinguished tape called the query tape, and three distinguished states called the query state, yes state, and no state, respectively. If M is a query machine and T is a set of strings, then a T-computation of M is a computation of M in which initially M is in the initial state and has an input string w on its input tape, and each time M assumes the query state there is a string u on the query tape, and the next state M assumes is the yes state if $u \in T$ and the no state if $u \notin T$. We think of an "oracle", which knows T, placing M in the yes state or no state.

Definition

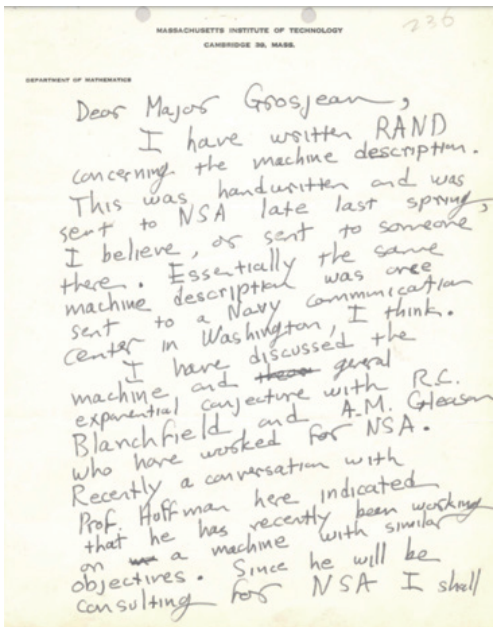
A set S of strings is P-reducible (P for polynomial) to a set T of strings (if there is some query machine M and a polynomial $Q(n)$ such that for each input string w, the T-computation of M with input w halts within $Q(|w|)$ steps ($|w|$ is the length of w), and ends in an accepting state iff $w \in S$.

It is not hard to see that P-reducibility is a transitive relation. Thus the relation $\leq_P$ on

算的衡量，但他博士没有毕业就进入工业界，后来帮助美国东部精英学校卫斯理安（Wesleyan）大学创办了计算机系；而哈特马尼斯和斯特恩斯则获得1993年的图灵奖。这两篇文章成文都在1963年，有意思的是科巴姆的文章是发表在科学哲学的会议上，尽管那会议和逻辑关系密切，但由此可见计算复杂性那时真是没人待见，会议论文集也出版在1965年。库克在哈佛读博期间就和科巴姆相熟，哈特马尼斯到哈佛做过一次关于计算复杂性的演讲，库克记忆深刻。

哈特马尼斯1988年发现了一封哥德尔1956年3月20号写给冯•诺依曼的信[5]，信中哥德尔指出：一个问题的难度可以表达为在图灵机上求解该问题所需步骤的函数，这个函数就是算法复杂性。于是，他把计算复杂性理论的诞生日推早到1956年。

[5] Hartmanis, J (1989), "Godel, von Neumann and the P=?NP Problem," *Bulletin of the European Association for Theoretical Computer Science*, pp.101-107.


MASSACHUSETTS INSTITUTE OF TECHNOLOGY
CAMBRIDGE 39, MASS.

DEPARTMENT OF MATHEMATICS

Dear Major Grosjean,

I have written RAND concerning the machine description. This was handwritten and was sent to NSA late last spring, I believe, or sent to someone there. Essentially the same machine description was once sent to a Navy communication center in Washington, I think.

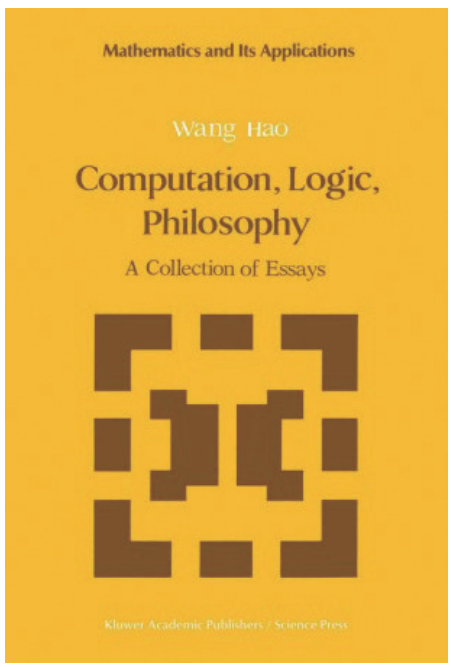
I have discussed the machine and ~~theor~~ general exponential conjecture with R.C. Blanchfield and A.M. Gleason who have worked for NSA.

Recently a conversation with Prof. Hoffman here indicated that he has recently been working on a machine with similar objectives. Since he will be consulting for NSA I shall

纳什给 NSA 的信

在 2012 年解密的美国国家安全署（NSA）的一批文件中，发现了一封天才纳什 1955 年 1 月给 NSA 的信[6]。这封信手写在麻省理工的办公信纸上，一共 8 页。纳什在前一年（1954）就提出了多项式复杂性和指数复杂性的区别，他推测指数复杂性对加密算法是有用的。纳什和当时麻省理工的几位同仁还讨论过所谓“指数猜想”，其中就有后来发明霍夫曼编码的霍夫曼（David Huffman），且英雄所见略同，霍夫曼也有类似想法。于是麻省理工学院计算理论大家西普瑟（Michael Sipser）认为纳什是复杂性概念的原创者[7]。

但更早，王浩在 1953 年曾写过一篇文章“Recursiveness and calculability”，文中提出了 speed function 的概念，这其实就是复杂性的雏形。可惜此文从未公开发表。王浩 1953 年夏天将此文提交给 *British Journal of Philosophy of Science*。评审的修改意见很长，王浩 1954 年做了修改，经过和评审的几次沟通，王浩失去了进一步修改的耐心。1984 年北大的逻辑学家吴允曾看到 1954 年手稿后建议王浩将此文收录进文集《计算，逻辑，哲学》（*Computation*，*Logic*，*Philosophy*）[8]，并将文章题目改为“The Concept of Computability”。不知道吴允曾先生有没有看到过 1950 年代王浩和杂志编辑的通讯。王浩还为此文写了一篇后记讲述了这段趣事。据此，我们也可以说计算复杂性起源于 1953 年。



《计算，逻辑，哲学》

我甚至曾把复杂性理论的缘起回溯得更早[9]。1939 年，哲学家维特根斯坦在剑桥开讲“数学基础”（Foundations of Mathematics），内容主要是维特根斯坦转型期思想的杂烩。此时刚从美国回到英国的图灵在剑桥赋闲，计划开讲数理逻辑，碰巧也把自己的课名为“数学基础”，当他得知大名鼎鼎的维特根斯坦也

[6] Nash, John（1955）. Communication to NSA.

[7] Sipser，Michael (1992), "The History and Status of the P versus NP Question," *24th ACM STOC - 5/92.*

[8] Wang，Hao (1989)，*Computation, Logic, Philosophy*，Springer Netherlands.

[9] 尼克（2014），哲学评书，浙江大学出版社 .