International Journal of Wireless Communications and Mobile Computing
2016; 4(2): 52-55
http://www.sciencepublishinggroup.com/j/wcmc
doi: 10.11648/j.wcmc.20160402.17
ISSN: 2330-1007 (Print); ISSN: 2330-1015 (Online)

Problem of Recognition of Hamiltonian Graph

Kochkarev Bagram Sibgatullovich

Department of Mathematics and Mathematical Modeling, Institute of Mathematics and Mechanics Named After Nikolai Ivanovich Lobachevsky, Kazan (Volga Region) Federal University, Kazan, Russia

Email address:
bkochkar@gmail.com

To cite this article:
Kochkarev Bagram Sibgatullovich. Problem of Recognition of Hamiltonian Graph. *International Journal of Wireless Communications and Mobile Computing.* Vol. 4, No. 2, 2016, pp. 52-55. doi: 10.11648/j.wcmc.20160402.17

Received: January 20, 2016; **Accepted**: February 3, 2016; **Published**: May 28, 2016

Abstract: In this article the author introduces the notions of combinatorial and of polynomial combinatorial sets in enumerative combinatorics. Formulates the problem of finding in combinatorial set of element with an easily recognizable property. The author proposes an efficient algorithm for solving this problem, which cancels known in the theory of algorithms abstract Turing, Church and Markov. We prove the criterion of polynomiality of the formulated problem. As a special case of this problem considers the problem of recognition of a Hamiltonian cycle in an undirected graph. We prove non-polynomiality this problem, which implies in particular the hypothesis of Jacques Edmonds *P≠NP*.

Keywords: Polynomial Problem, NP-problem, NPC-problem, Hamiltonian Cycle, Hamiltonian Graph

1. Introduction

In 1964 A. Cobham [1] and, independently, in 1965 J. Edmonds [2] introduced the concept of complexity class P.

Definition [1-2]. A problem (language) L belongs to P if there exists an algorithm A that decides (recognizes) L in a polynomial number of steps ($< O(n^k)$), where n is the length of the input, and k is some constant. The class of problem P is called polynomial (practical).

According to [3] J. Edmonds also introduced the complexity class NP. This class of problems (languages) that can be checked by polynomial algorithms.

Definition 2 [3]. A problem (language) L belongs to NP if there exists a two-input polynomial-time algorithm A and a polynomial $p(x)$ with integer coefficients that

$$L = \{x \in \{0,1\}^* : there\ exists\ a\ certificate\ y\ with\ |y| \le p(|x|)\ and\ A(x,y) = 1\}$$

In this case we say that algorithm A verifies the solution x in polynomial number of steps of the length x.

According to definition 2, if L belongs to P and $|y| \le p(|x|)$, then $L \in NP$. But, if $L \in P$ and the length of certificate is not bounded from above by a polynomial of the length of x then $L \notin NP$ and P is not subset of NP. Note that in [3-5] $P \subseteq NP$, that is the erroneous statement. According to [3] J. Edmonds also expressed the hypothesis $P \ne NP$. In [6-10] we have constructed classes of polynomial problems with not polynomial certificates. According to the reasoning above this implies the positive solution of problem S. A. Cook [11] and the hypothesis of J. Edmonds $P \ne NP$. In definition 2 of the class NP there is a limit to the length of the certificate generated by the inspection algorithm, apparently caused by the algorithm for checking the correctness of solution of the classical algebraic problem of solving algebraic equation. It is well known that if x_0 is a solution of the equation $p(x) = 0$, where $p(x)$ is a polynomial from x with integer coefficients, then the verification of the correctness of the decision x_0 is the equality $p(x_0) = 0$. Obviously, if the solution is sought in a polynomial number of steps, then the validation also is carried out in a polynomial number of steps. Of course, the way to check the correctness of the solution of the problem depends on the nature of the problem. There are problems that have no solutions, and hence can't be checked, although the certificates to verify the correctness of the decision and there are. For example, in the thirties of the 19 th century Galois was able to prove [12] that for any $n \ge 5$ you can specify unsolvable by radicals of equation n degree with integer coefficients.

Naturally, according to definition 2 of the class NP, any polynomial problem belongs to NP, since the composition of polynomials is a polynomial. But if to remove restriction on length of the certificate that is explainable because the way of check of correctness of the decision depends both on a problem, and from properties of the decision which correctness is checked. Therefore the verification of decision may not be polynomial.

If we are interested not only a solution, but also check of its correctness, usually believe that the decision is a direct task, and check of its correctness – inverse. So finding of the cipher in a cryptosystem is a direct problem, and the decryption is the inverse problem. Therefore in cryptography it is very important that the transcript was algorithmically more difficult than finding the cipher. In this interpretation, a cryptographic system is the solution S. A. Cook's problem, proposed by the Institute Clay among the seven Millennium problems whose solution was presented in [6-10]. Although this problem was resolved already, we can say, seven years ago [7] and published on International conferences [6, 9] and in licensed journals [13-14] and notified to the Institute Clay, but the result is still the Clay Institute not recognized and therefore in [15] this problem is considered not solved. Note that in [16-17] there are positive feedbacks on our work on this issue.

2. NP-complete Problems and Their Complexity

In 1971 S. A. Cook [18] in the class of *NP* allocated a subclass of the most difficult problems of the so-called *NPC* (*NP*-complete) problem and proved *NP* completeness of two specific problems. Independently the concept of *NP*-completeness has also introduced L. A. Levin [19], who also proved the *NP* completeness of several problems. Since then many authors have proved the *NP* completeness thousands of problems, but the nature of their complexity not officially clarified.

Definition 3 [3]. We will say that a language $L_1 \subseteq \{0,1\}^*$ polynomial-time reducible to language $L_2 \subseteq \{0,1\}^*, (L_1 \leq_P L_2)$, if there exists a computable in polynomial number of steps the function $f : \{0,1\}^* \rightarrow \{0,1\}^*$ such that for all $x \in \{0,1\}^* x \in L_1$ if and only if $f(x) \in L_2$. We call the function f the reducing function, and a polynomial-time algorithm F that computes f is called a reducing algorithm.

Definition 4 [3]. A language $L \subseteq \{0,1\}^*$ is called *NP* complete, if

1. $L \in NP$, and
2. $L' \leq_P L$ for any $L' \in NP$.

According to the definition 3, 4 if some *NP* complete problem is polynomial, then all *NP* complete problems are polynomial, and Vice Versa, if some *NP* complete problem is not polynomial, then all *NP* complete problems are not polynomial. All attempts to prove, that some *NP* complete problem is polynomial were in vain. In [20] we built *NP* complete problem that is not polynomial.

In this paper we propose one method of proving this fact using enumerative combinatorics and we prove that the recognition problem undirected Hamiltonian graph, *NP* completeness of which was proven in 1972 R. M. Karp [21], is not polynomial.

Mathematical Institute Clay explains the problem of S. A. Cook [11] by the following example: let's say someone in a big company wants to make sure that in this company there is one of his friends. If he is told that his friend is sitting in the corner, then only a fraction of a second to make sure that it is so. However, if no such information, he will be forced to circumvent the entire room, examining visitors. Note that this example fails because in modern conditions this problem is easily solved, e.g., using a microphone. Another thing, if we want to find some object with easily verifiable (in polynomial time of the length of the input) a symptom among the many combinatorial objects.

Enumerative combinatorics [22] deal with counting the number of elements in the finite set S. In enumerative combinatorics, the combinatorial elements of set S have a simple combinatorial definition, and some additional structure. Shows that the set S contains many elements and the main question is to determine (estimate) their combinatorial number, not a search, for example, some special item. The problem, the solution of which we intend to present in this work is, precisely, to build an effective search algorithm is a special element among the elements of some combinatorial sets.

In enumerative combinatorics [22] is usually given an infinite class of finite sets S_i, where i runs over some set of indices I (the set of nonnegative integers N), and we want to count combinatorial number $|S_i|$ of elements in each S_i "simultaneously". The set S_i we will call combinatorial sets.

However, for our purposes it is sufficient to know the particular answer to the question: "$|S_i|$ polynomial or not from the i?" In the case of an affirmative answer to the question we have to prove that $|S_i| < O(i^k)$, where k is some constant.

Definition 5 [23]. Combinatorial set S_i is called polynomial if $|S_i| \leq O(i^k)$, where k is a constant; otherwise it is called non-polynomial.

Problem statement: let S_i combinatorial set and x some object from S_i with easy to verify (in a polynomial number of steps of $|x|$) characteristic (property) α. It is necessary to construct an algorithm that for $\geq |S_i|$ the number of steps finds the object (element) x. The following algorithm is proposed to solve the problem: we assume that a finite set S_i of combinatorial objects is in some capacity Ω. From Ω we derive successively without returning all objects to the last, inclusive. For the latter object we check that the properties α. If the last retrieved object has the property α, then the problem is solved. If this object does not have property α, then all of the extracted of Ω objects we returned in Ω and renewable the extraction process of Ω objects, but with each subsequent extraction of the object check whether the extracted object is α property or not. Obviously, for some $|S_i| + m, m \leq |S_i|$ step, the object will have the property α and the problem will be solved. From the constructed algorithm follows theorem.

Theorem [23]. If the combinatorial set S_i is polynomial, the formulated problem can be solved in polynomial number of steps, namely, the number of steps t of the constructed algorithm satisfies the inequalities $|S_i| \leq t \leq C|S_i|$, where C is some constant ≥ 1. If combinatorial set S_i is not a polynomial, then the problem is not polynomial. Obviously, if $|S_i| \geq k^i, k \geq 2$, then the combinatorial set S_i is not a polynomial.

The algorithm, which implies the above theorem obviously satisfies all the requirements of the intuitive notion of algorithm, but does not fit [23] the formal definition of the algorithm (for example, "Turing machine"). Therefore, the algorithm cancels famous Turing thesis: "any algorithm can build a Turing machine that is equivalent to a given algorithm". Due to the equivalence of a Turing other well-known formal algorithms (recursive function, normal algorithms) relevant abstracts are also voided.

The problem of finding a Hamiltonian cycle in a undirected graph were studied [3] for over a hundred years. Some Hamiltonian cycle of an undirected graph $G = (V, E)$ is a simple cycle that contains each vertex of V. A graph which contains some Hamiltonian cycle is called Hamiltonian; otherwise, the count is not Hamiltonian. However, not all graphs are Hamiltonian. Every bibartite graph with odd number of vertices is not Hamiltonian [3]. We [3] to formulate the Hamiltonian cycle problem, "does some given graph G Hamiltonian cycle? As a formal language: HAM-CYCLE=$\{G : G \text{ is a Hamiltonian graph}\}$".

Let $G = (V = V_1 \cup V_2, E)$ be a bipartite graph such that $|V_1| = \left\lfloor \frac{i}{2} \right\rfloor, |V_2| = \left\lceil \frac{i}{2} \right\rceil, i$ is an odd natural number, $|E| = \left\lfloor \frac{i}{2} \right\rfloor \left\lceil \frac{i}{2} \right\rceil$. Thus, in the graph G every vertex from V_1 is connected with each vertex of V_2. Furthermore, let $G' = (V, E')$ is the graph obtained from graph G by adding one edge u, connecting two vertices $v \in V_2, v' \in V_2$. As the combinatorial set S_i, we consider the set of all simple paths from a vertex v in the graph G that contains each vertex in V. Obviously, all simple paths from vertex v to vertex v' in graph G, that contains each vertex in V, adding edge u in graph G' are transformed into Hamiltonian cycles. Thus, combinatorial number $|S_i|$ for graph G and G' are the same. It is obvious that $|S_i| = (\left\lfloor \frac{n}{2} \right\rfloor !)^2$. Since $|S_i|$ is not a polynomial, then according to the theorem, the Hamiltonian cycle problem is not polynomial. It also follows the results.

Corollary 1. All NP complete problems are not polynomial.

Corollary 2. Hypothesis J. Edmonds $P \neq NP$ is correct.

3. Conclusions

Certainly, there are polynomial problems, check of which correctness of the decision demands the polynomial number of steps (for example, problems of sorting [24]), however it doesn't mean that the class of polynomial problems enters the class NP, i.e. $P \subseteq NP$ as it was noted in introduction, is the wrong statement.

Thus, the problem $P = NP$, it is trying to solve the authors [25-26] does not exist in nature. Therefore, it is impossible to solve by any means, and not only natural [24]. Once again we repeat: on the one hand, we have shown [6-10] the existence of polynomial problems the verifying the correctness of the decision which requires a non-polynomial number of steps that need cryptography on the other hand, we have proved that NP complete problems are not polynomial.

References

[1] Cobham A. The intrinsic computational difficulty of functions //In Procedings of the 1964 Congress for Logic, Methodology, and the Philosophy of Science.-North-Holland, 1964. - P. 24-30.

[2] Edmonds J. Paths, trees and flowers //Canadian Journal of Mathematics.-1965-Vol. 17. - P. 449-467.

[3] Cormen T. H., Leiserson Ch. E., Rivest R. L., Introduction to Algorithms, MIT Press, 1990. 2002. P. 955.

[4] Cook S. A., The P versus NP Problem //Manuscript prepared for the Clay Mathematics Institute for the Millennium, April, 2000, www.cs.toronto.edusacook.

[5] Razborov A. A. Theoretical Computer Science: vzglyad mathematica, http://old.Computerra.ru/offline/2001/379/6782.

[6] Kochkarev B. S. Prilogenie monotonnykh funktsij algebry logiki k probleme Kuka, Nauka v Vuzakh: matematika, fizika, informatika, Tezisy dokladov Mejdunarodnoj nauchno-obrazovatelnoj konferentsii, 2009, pp. 274-275. (in Russian).

[7] Kochkarev B. S., On Cook's problem, http://www.math.nsc.ru/conference/malmeet/08/Abs.

[8] Kochkarev B. S. About one class polynomial problems with not polynomial certificates //arxiv: 1210.7591v1 [math. CO] 29 Oct 2012/.

[9] Kochkarev B. S., About one class polynomial problems with not polynomial certificates //Second International Conference "Claster Computing"CC2013 (Ukraine, Lviv, June 3-5, 2013) P. 99-100.

[10] Kochkarev B. S., K probleme Kuka //Matematicheskoe obrazovanie v shkole I v Vuze v usloviyakh perekhoda na novye obrazovatelnye standarty. Materialy Vserossiyskoy nauchno-practicheskoy konferentsii s mejdunarodnym uchastiem. TGGPU, Kazan, 2010. s. 133-136 (in Russian).

[11] The Clay Mathematics Institute www.Claymath.org.Millennium Prize Problems.

[12] Kurosh A. G. Kurs vysshey algebry. Izd. F. -M. Literatury, M., 1962, 432. (in Russian).

[13] Kochkarev B. S., Vzaimootnosheniya mejdu slojnostnymi klassami P, NP i NPC. Problems of modern science and education. 2015, 11 (41), s. 6-8 (in Russian).

[14] Kochkarev B. S., Dokazatelstvo gipotezy Edmondsa i reshenie problemy Kuka. Nauka, Tekhnika i Obrazovanie, 2014, 2 (2), s. 6-9. (in Russian).

[15] Ravenstvo klassov P i NP https://ru.wikipedia.org/wiki.

[16] En waarmee toverde het internet vandaag een lach op Uwge https://www.uscki.nl.

[17] Gipoteza J. Edmondsa i problema S. A. Kuka http://www.refereed.ru.

[18] Cook S. A. The complexity of theorem proving procedures. //In Proceedings of the Third Annual ACM Symposium on Theory of Computing. P. 151-158, 1971.

[19] Levin L. A. Universal sorting problems. //Problemy Peredachi Informatsii, 9(3), s. 265-266, 1973.

[20] Kochkarev B. S. Proof of the hypothesis Edmonds's, not polynomial of NPC problems and classification of the problems with polynomial certificates.//arxiv: 1303.2580v1 [cs. CC] 7 Mar 2013.

[21] Karp R. M. Reducibility among combinatorial problems. //In Raymond E. Miller and James W. Thatcher, editors, Complexity of computer Computations, P. 85-103, Plenum Press, 1972.

[22] Stanley R. P. Enumerative Combinatorics v. 1, 1986.

[23] Kochkarev B. S. Ob odnom algoritme, ne soglasuyutchemsya s tezisami Turinga, Churcha i Markova, Problems of modern science and education, M., 2014, 3(21) c. 23-25 (in Russian).

[24] Kochkarev B. S. Gipoteza J. Edmondsa i problema S. A. Kuka. //Vestnik TGGPU, 2011 2 (24) s. 23-24 (in Russian).

[25] Razborov A. A., Rudich S. Natural proof. Proceedings of the 26 th Annual ACM Symposium on the Theory of Computing. P. 204-213 DOI: 10.1145/195058.195134.

[26] Babay priblizilsya k resheniyu pronlemy tysyacheletiya. http://lenta.ru/news/2015/11/20/graphtheory/.