THE COMPLEXITY OF GOLDBACH'S CONJECTURE

FRANK VEGA

ABSTRACT. On the one hand, the Goldbach's conjecture has been described as the most difficult problem in the history of Mathematics. This conjecture states that every even integer greater than 2 can be written as the sum of two primes. On the other hand, P versus NP is considered as one of the most important open problems in computer science. This consists in knowing the answer of the following question: Is P equal to NP? In computational complexity theory, another major complexity class is ASPACE(S(n)) for some S(n). It is known that ASPACE(log n) = P. We prove if the complexity class NP is equal to ASPACE(S(n)) for some S(n) = o(log n), then the Goldbach's conjecture is false.

1. INTRODUCTION

Number theory is a branch of pure mathematics devoted primarily to the study of the integers and integer-valued functions [19]. Goldbach's conjecture is one of the most important and unsolved problems in number theory [9]. Nowadays, it is one of the open problems of Hilbert and Landau [9]. Goldbach's original conjecture, written on 7 June 1742 in a letter to Leonhard Euler, states: "... at least it seems that every number that is greater than 2 is the sum of three primes" [7]. This is known as the ternary Goldbach conjecture. We call a prime as a natural number that is greater than 1 and has exactly two divisors, 1 and the number itself [22]. However, the mathematician Christian Goldbach considered 1 as a prime number. Euler replied in a letter dated 30 June 1742 the following statement: "Every even integer greater than 2 can be written as the sum of two primes" [7]. This is known as the strong Goldbach conjecture.

Using Vinogradov's method [21], it has been showed that almost all even numbers can be written as the sum of two primes. In 1973, Chen showed that every sufficiently large even number can be written as the sum of some prime number and a semiprime [5]. The strong Goldbach conjecture implies the conjecture that all odd numbers greater than 7 are the sum of three odd primes, which is known today as the weak Goldbach conjecture [7]. In 2012 and 2013, Peruvian mathematician Harald Helfgott published a pair of papers claiming to improve major and minor arc estimates sufficiently to unconditionally prove the weak Goldbach conjecture [10], [11].

The strong conjecture has remained unsolved. For small values, the strong Goldbach conjecture can be verified directly. Nils Pipping in 1938 laboriously verified the conjecture up to $n \leq 10^5$ [16]. More values have been checked with the use of computers. A pair of primes that sum to an even integer is known as a Goldbach

2010 *Mathematics Subject Classification.* Primary 11Axx, 68Q15, 11A41, 68Q45.

Key words and phrases. number theory, complexity theory, primes, regular languages.

partition. Oliveira e Silva has run a distributed computer search that has verified the existence of Goldbach partitions for $n \leq 4 * 10^{18}$ until 2013 [14].

We define a non-regular language under the assumption that the Goldbach's conjecture is true. Indeed, the possible language can be easily proved that might be non-regular using the Pumping lemma for regular languages [17]. However, we prove this language is actually regular when the complexity class NP is equal to $ASPACE(S(n))$ for some $S(n) = o(\log n)$. This result is based on the breakthrough result that checking whether a number is prime can be decided in polynomial time [2]. In this way, if $NP = ASPACE(S(n))$ for some $S(n) = o(\log n)$, then the Goldbach's conjecture is false.

2. Background Theory

In 1936, Turing developed his theoretical computational model [20]. The deterministic and nondeterministic Turing machines have become in two of the most important definitions related to this theoretical model for computation [20]. A deterministic Turing machine has only one next action for each step defined in its program or transition function [20]. A nondeterministic Turing machine could contain more than one action defined for each step of its program, where this one is no longer a function, but a relation [20].

Let Σ be a finite alphabet with at least two elements, and let Σ^* be the set of finite strings over Σ [4]. A Turing machine M has an associated input alphabet Σ [4]. For each string w in Σ^* there is a computation associated with M on input w [4]. We say that M accepts w if this computation terminates in the accepting state, that is $M(w) = $ "*yes*" [4]. Note that M fails to accept w either if this computation ends in the rejecting state, that is $M(w) = $ "*no*", or if the computation fails to terminate, or the computation ends in the halting state with some output, that is $M(w) = y$ (when M outputs the string y on the input w) [4].

Another relevant advance in the last century has been the definition of a complexity class. A language over an alphabet is any set of strings made up of symbols from that alphabet [6]. A complexity class is a set of problems, which are represented as a language, grouped by measures such as the running time, memory, etc [6]. The language accepted by a Turing machine M, denoted $L(M)$, has an associated alphabet Σ and is defined by:

$$L(M) = \{w \in \Sigma^* : M(w) = \text{“}yes\text{”}\}.$$

Moreover, $L(M)$ is decided by M, when $w \notin L(M)$ if and only if $M(w) = $ "*no*" [6]. We denote by $t_M(w)$ the number of steps in the computation of M on input w [4]. For $n \in \mathbb{N}$ we denote by $T_M(n)$ the worst case run time of M; that is:

$$T_M(n) = max\{t_M(w) : w \in \Sigma^n\}$$

where Σ^n is the set of all strings over Σ of length n [4]. We say that M runs in polynomial time if there is a constant k such that for all n, $T_M(n) \leq n^k + k$ [4]. In other words, this means the language $L(M)$ can be decided by the Turing machine M in polynomial time. Therefore, P is the complexity class of languages that can be decided by deterministic Turing machines in polynomial time [6]. A verifier for a language L_1 is a deterministic Turing machine M, where:

$$L_1 = \{w : M(w, c) = \text{“}yes\text{”} \textit{ for some string } c\}.$$

We measure the time of a verifier only in terms of the length of w, so a polynomial time verifier runs in polynomial time in the length of w [4]. A verifier uses additional information, represented by the symbol c, to verify that a string w is a member of L_1. This information is called certificate. NP is the complexity class of languages defined by polynomial time verifiers [15].

We use o-notation to denote an upper bound that is not asymptotically tight. We formally define $o(g(n))$ ("little-oh of g of n") as the set

$$o(g(n)) = \{f(n) : \textit{ for any positive constant } c > 0, \textit{ there exists a constant}$$
$$n_0 > 0 \textit{ such that } 0 \leq f(n) < c \times g(n) \textit{ for all } n \geq n_0\}.$$

For example, $2 \times n = o(n^2)$, but $2 \times n^2 \neq o(n^2)$ [6].

In theoretical computer science and formal language theory, a regular language is a formal language that can be expressed using a regular expression [3]. The complexity class that contains all the regular languages is REG. An alternating Turing machine is a nondeterministic machine with two kinds of states, AND states and OR states [15]. It accepts if and only if the tree of all computation paths, considered as an AND-OR tree, evaluates to 1: Here "Accept" corresponds to 1 and "Reject" to 0 [15]. The complexity class $ASPACE(f(n))$ is the set of decision problems that can be solved by an alternating Turing machine M, using space $f(n)$, where n is the length of the input [13]. It is known that $ASPACE(\log n) = P$ [15].

3. Results

Definition 3.1. We define the Goldbach's language L_G as follows:

$$L_G = \{1^{2 \times n} 0^p 0^q : n \in \mathbb{N} \wedge n \geq 3 \wedge p \textit{ and } q \textit{ are odd primes } \wedge 2 \times n = p + q\}.$$

Theorem 3.2. *If the Goldbach's conjecture is true, then the Goldbach's language L_G is non-regular.*

Proof. If the Goldbach's conjecture is true, then the Goldbach's language L_G is equal to the another language L' defined as follows:

$$L' = \{1^{2 \times n} 0^{2 \times n} : n \in \mathbb{N} \wedge n \geq 3\}.$$

We can easily prove that L' is non-regular using the Pumping lemma for regular languages [17]. □

Definition 3.3. We define the verification Goldbach's language L_{VG} as follows:

$$L_{VG} = \{(2 \times n, p, q) : n \in \mathbb{N} \wedge n \geq 3 \wedge p \textit{ and } q \textit{ are odd primes } \wedge 2 \times n = p + q\}.$$

Theorem 3.4. *$L_{VG} \in P$.*

Proof. This result is based on the breakthrough result that checking whether a number is prime can be decided in polynomial time [2]. Certainly, we can check in polynomial time whether p and q are odd primes and the other verifications can be easily done in polynomial time as well. □

Definition 3.5. We define the Goldbach's language with separator L_{SG} as follows:

$$L_{SG} = \{0^{2 \times n} \# 0^p \# 0^q : n \in \mathbb{N} \wedge n \geq 3 \wedge p \textit{ and } q \textit{ are odd primes } \wedge 2 \times n = p + q\}$$

where # is the blank symbol.

Lemma 3.6. *The Goldbach's language L_{SG} is the unary representation of the verification Goldbach's language L_{VG}.*

Proof. This is trivially true under the definition of both languages. □

Theorem 3.7. *If $NP = ASPACE(S(n))$ for some $S(n) = o(\log n)$, then $L_G \in REG$.*

Proof. If $NP = ASPACE(S(n))$ for some $S(n) = o(\log n)$, then we obtain that $L_{VG} \in ASPACE(S(n))$ because of Theorem 3.4 and $P \subseteq NP$ [15]. Since $NP = ASPACE(S(n))$ for some $S(n) = o(\log n)$ implies $P = NP$ [15], then there is an alternating Turing machine which decides L_{SG} that uses space that is smaller than $c \times \log\log n$ for all $c > 0$, because of L_{SG} is the unary version of L_{VG} due to Lemma 3.6 [8]. Certainly, the standard space translation between the unary and binary languages actually works for alternating machines with small space when $P = NP$ [8]. This means that under the assumption of $P = NP$: If some language belongs to $ASPACE(S(n))$, then the unary version of that language belongs to $ASPACE(S(\log n))$ [8]. In this way, we obtain that $L_{SG} \in REG$ because of $REG = ASPACE(o(\log\log n))$ [12]. In addition, we can reduce in a nondeterministic constant space the language L_G to L_{SG} just nondeterministically inserting the blank symbol # within an arbitrary position between the 0's on the input. Moreover, this nondeterminism reduction inserts the blank symbol # between the 1's and 0's and converts the 1's to 0's from the original input of L_G just generating the final output to L_{SG}. Consequently, we prove $L_G \in REG$ under the assumption that $NP = ASPACE(S(n))$ for some $S(n) = o(\log n)$, since REG is also the complexity class of languages decided by nondeterministic Turing machines in constant space [18]. □

Theorem 3.8. *If $NP = ASPACE(S(n))$ for some $S(n) = o(\log n)$, then the Goldbach's conjecture is false.*

Proof. If the Goldbach's conjecture is true, then $L_G \notin REG$ as a consequence of Theorem 3.2. However, if $NP = ASPACE(S(n))$ for some $S(n) = o(\log n)$, then $L_G \in REG$ due to Theorem 3.7. In this way, the Goldbach's conjecture cannot be true under the assumption that $NP = ASPACE(S(n))$ for some $S(n) = o(\log n)$. □

4. Conclusions

Statistical considerations that focus on the probabilistic distribution of prime numbers present informal evidence in favour of the strong conjecture for sufficiently large integers: The greater the integer, the more ways there are available for that number to be represented as the sum of two other numbers, and the more "likely" it becomes that at least one of these representations consists entirely of primes. Hence, this is a possible evidence that $NP \neq ASPACE(S(n))$ for all $S(n) = o(\log n)$ is "likely" to be true as well.

The P versus NP problem is the major unsolved problem in computer science [1]. This is considered by many to be the most important open problem in the field [1]. This work also proves under the assumption of $P = NP$: When the Goldbach's conjecture is true, then $ASPACE(\log n) \neq ASPACE(S(n))$ for all $S(n) = o(\log n)$. Note, this result is already proved, but only when $S(n)$ is fully space constructible [13]. However, if $\lim S(n) = +\infty$ and $S(n) = o(\log n)$, then $S(n)$ is not fully space constructible [13]. Hence, this is a still open problem [13].

References

1. Scott Aaronson, *$P \overset{?}{=} NP$*, Electronic Colloquium on Computational Complexity, Report No. 4 (2017).
2. Manindra Agrawal, Neeraj Kayal, and Nitin Saxena, *PRIMES is in P*, Annals of Mathematics **160** (2004), no. 2, 781–793.
3. Alfred V. Aho and John E. Hopcroft, *The Design and Analysis of Computer Algorithms*, Pearson Education India, 1974.
4. Sanjeev Arora and Boaz Barak, *Computational complexity: a modern approach*, Cambridge University Press, 2009.
5. Jing-run Chen, *On the Representation of a Large Even Integer as the Sum of a Prime and the Product of Two Primes at Most*, Sci. Sinica **16** (1973), 157–176.
6. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein, *Introduction to Algorithms*, 3 ed., The MIT Press, 2009.
7. Leonard Eugene Dickson, *History of the Theory of Numbers: Divisibility and Primality*, vol. 1, New York, Dover, 2005.
8. Viliam Geffert and Dana Pardubská, *Unary Coded NP-Complete Languages in ASPACE (log log n)*, International Journal of Foundations of Computer Science **24** (2013), no. 07, 1167–1182.
9. Richard K. Guy, *Unsolved Problems in Number Theory*, 3 ed., New York, Springer-Verlag, 2004.
10. Harald A. Helfgott, *Minor arcs for Goldbach's problem*, arXiv preprint arXiv:1205.5252 (2012).
11. ———, *Major arcs for Goldbach's theorem*, arXiv preprint arXiv:1305.2897 (2013).
12. Kazuo Iwama, *$ASPACE(o(\log\log n))$ is Regular*, SIAM Journal on Computing **22** (1993), no. 1, 136–146.
13. Pascal Michel, *A survey of space complexity*, Theoretical computer science **101** (1992), no. 1, 99–132.
14. Tomás Oliveira e Silva, *Goldbach conjecture verification*, April 2013, At `http://sweet.ua.pt/tos/goldbach.html`. Accessed 31 January 2020.
15. Christos H. Papadimitriou, *Computational complexity*, Addison-Wesley, 1994.
16. Nils Pipping, *Die Goldbachsche Vermutung und der Goldbach-Vinogradovsche Satz*, Acta. Acad. Aboensis, Math. Phys. **11** (1938), 4–25.
17. Michael O. Rabin and Dana Scott, *Finite automata and their decision problems*, IBM journal of research and development **3** (1959), no. 2, 114–125.
18. John C. Shepherdson, *The Reduction of Two-Way Automata to One-Way Automata*, IBM Journal of Research and Development **3** (1959), no. 2, 198–200.
19. Joseph H. Silverman, *A Friendly Introduction to Number Theory*, 4 ed., Pearson Education, Inc., 2012.
20. Michael Sipser, *Introduction to the Theory of Computation*, vol. 2, Thomson Course Technology Boston, 2006.
21. Ivan M. Vinogradov, *Representation of an Odd Number as a Sum of Three Primes*, Comptes Rendus (Doklady) de l'Académie des Sciences de l'USSR **15** (1937), 169–172.
22. David Wells, *Prime Numbers, The Most Mysterious Figures in Math*, John Wiley & Sons, Inc., 2005.

Joysonic, Uzun Mirkova 5, Belgrade, 11000, Serbia
E-mail address: `vega.frank@gmail.com`