***Applied Mathematics*, 2013, 4, 1635-1636**
Published Online December 2013 (http://www.scirp.org/journal/am)
http://dx.doi.org/10.4236/am.2013.412222

The RSA Cryptographic Protocol Is Not Secure

Cristian Dumitrescu
Kitchener, Canada
Email: cristiand43@gmail.com, cristiand41@hotmail.com

Received August 31, 2013; revised September 30, 2013; accepted October 7, 2013

ABSTRACT

In this article I describe a randomized algorithm based on random walks with two absorbing barriers that solves the satisfiability problem (known to be NP complete) with arbitrary high probability. As a consequence of this algorithm, I also prove that the RSA cryptographic protocol is not secure.

Keywords: The Satisfiability Problem; Hamming Distance; Random Walk with Two Absorbing Barriers

1. Useful Notions That Are Used for the Analysis of the Algorithm

A Boolean expression is said to be in conjunctive normal form (CNF) if it is of the form $E_1 \wedge E_2 \wedge E_3 \wedge \cdots \wedge E_k$, and each E_i, called a clause (or conjunct), is of the form $\alpha_{i1} \vee \alpha_{i2} \vee \alpha_{i3} \cdots \vee \alpha_{ir}$, where each α_{ij} is a literal, either x or x, for some variable x.

A Boolean expression is said to be in disjunctive normal form (DNF) if it is of the form $F_1 \vee F_2 \vee F_3 \cdots \vee F_k$, and each F_j, called a clause (or disjunct), is of the form $\beta_{j1} \wedge \beta_{j2} \wedge \beta_{j3} \wedge \cdots \wedge \beta_{jr}$, where each β_{jk} is a literal, either y or y, for some variable y.

A Boolean expression in CNF form is called satisfiable if there is some assignment of 0's and 1's to the variables that gives the expression the value 1.

The satisfiability problem is to determine, given a Boolean expression, whether it is satisfiable. We note that any Boolean expression E in CNF form has a dual Boolean expression E^* in DNF form (we only replace every $\wedge$ with a $\vee$, and replace every $\vee$ with a $\wedge$). For example, if $E(x_1, x_2, x_3, x_4) = (x_1 \vee x_2) \wedge (x_3 \vee x_4)$, then $E^*(x_1, x_2, x_3, x_4) = (x_1 \wedge x_2) \vee (x_3 \wedge x_4)$. We note that the binary vector $x = (x_1, x_2, x_3, \cdots, x_n)$ represents a solution for the equation $E(y_1, y_2, y_3, \cdots, y_n) = 1$ (in CNF form) if the binary vector $\neg x = (\neg x_1, \neg x_2, \cdots, \neg x_n)$ represents a solution for the dual equation
$E^*(y_1, y_2, y_3, \cdots, y_n) = 0$. This follows immediately from the equivalence

$$\left(E(y_1, y_2, y_3, \cdots, y_n) = 1\right) \leftrightarrow \left(\neg E(y_1, y_2, y_3, \cdots, y_n) = 0\right)$$

An expression is said to be 3-CNF if each clause has exactly three distinct literals.

Theorem 1 (see Reference [1]). L_{3SAT}, the satisfiability problem for 3-CNF expressions, is NP-complete.

The Hamming distance $d_H(\boldsymbol{x}, \boldsymbol{y})$ between two vectors $\boldsymbol{x}$, $\boldsymbol{y}$ is the number of components in which they differ. It is known that the Hamming distance $d_H(\boldsymbol{x}, \boldsymbol{y})$ satisfies the conditions for a metric.

Related to the theory of symmetric random walks (in one dimension), we have the following theorem.

Theorem 2 (see Reference [2]). Limit theorem for first passages. For fixed t, the probability that the first passage through r occurs before epoch $t \cdot r^2$ tends to

$P = 2 \cdot \left(1 - N\left(\frac{1}{\sqrt{t}}\right)\right)$, as $r \to \infty$, where N is the normal distribution function. We note that when $t \to \infty$, then P tends to 1.

2. The Description and Analysis of the Algorithm

We consider a Boolean expression $E(y_1, y_2, y_3, \cdots, y_n)$ in 3-CNF form with n variables. We want to determine whether it is satisfiable.

Step 1. Randomly generate a binary vector
$x = (x_1, x_2, x_3, \cdots, x_n)$, each x_i, will be 0 or 1 in a random manner.

Step 2. If $x = (x_1, x_2, x_3, \cdots, x_n)$ is a solution for the equation $E(y_1, y_2, y_3, \cdots, y_n) = 1$, then return
$x = (x_1, x_2, x_3, \cdots, x_n)$.

Otherwise, if $x = (x_1, x_2, x_3, \cdots, x_n)$ is a solution for the equation $E^*(y_1, y_2, y_3, \cdots, y_n) = 0$, then return $\neg x = (\neg x_1, \neg x_2, \cdots, \neg x_n)$.

Otherwise go to Step 3.

Step 3. Randomly choose a component of the vector *x*, and flip its binary value. In other words, randomly choose a value i in the set {1, 2, ⋯ *n*}, and flip the value of x_i. If x_i is 0, then put $x_i = 1$, and if x_i is 1, then put $x_i = 0$.

Repeat Steps 2 and 3 for $t \cdot n^2$ cycles (where *t* is a fixed number). If no solution for our Boolean equation has been found after $t \cdot n^2$ cycles, then the 3-CNF expression under consideration is considered not satisfiable.

If the 3-CNF expression is not satisfiable, then the algorithm will report that the expression is not satisfiable. If the expression is satisfiable, then the initially randomly generated vector will be at a certain Hamming distance d from a solution of the equation $E(y_1, y_2, y_3, \cdots, y_n) = 1$, and at Hamming distance *n* – *d* from the corresponding solution of the equation $E^*(y_1, y_2, y_3, \cdots, y_n) = 0$. With each cycle (in particular step 3), it is as likely that the Hamming distance *d* will increase or decrease. The Hamming distance *d* will increase or decrease with probability $\frac{1}{2}$. We basically have a symmetric random walk with two absorbing barriers. From Theorem 2, we see that by choosing a large t, we can make the probability that the algorithm will fail as small as we want (we say that the algorithm will fail if it reports that our 3-CNF form is not satisfiable, when in fact it is satisfiable).

We can also use parallel computers (processors) that deal with the same problem in parallel, and the probability that the algorithm will fail on all simultaneously will be as small as we want. We can design the system so that we can run the algorithm for many problems, for a time comparable with the age of the Universe, and we can expect it to fail once or twice. I think that this is acceptable, in relation to practical applications. Since we are dealing with a NP-complete problem, this algorithm will solve a multitude of problems.

3. Applications

This algorithm will have applications in industry, medicine, and many other domains of activity (where efficiency is an issue, see Reference [3]). It can also be proved that the RSA cryptographic protocol, on which most of the Internet transactions and activity are based, is **not secure**. RSA relies on the assumption that it is easy to multiply numbers, but very difficult to factor them. Here is an randomized algorithm for factoring large numbers, that is polynomial, and it is based on a random walk with an absorbing barrier and a reflecting barrier.

We consider a large *n*-bit number *N* written in binary. We want to factor it.

For m taking values from 1 to *n*, perform the following three steps (actually, for each *m*, perform many cycles, as described below).

Step 1. Randomly generate an *m*-bit binary number *x*.
Step 2. If *x* is a divisor of *N*, then return *x*.
Otherwise go to Step 3.
Step 3. Randomly choose a bit of the *m*-bit number *x*, and flip its binary value. In other words, randomly choose a value *i* in the set {1, 2, ⋯ *m*}, and flip the *i*-th bit of *x*.

Repeat Steps 2 and 3 for $t \cdot m^2$ cycles (where *t* is a fixed number).

This algorithm runs for a total of $C \cdot n^3$ cycles (where the constant *C* can be determined), and it either finds a divisor of *N*, or else says that *N* is prime (we make sure that we exclude 1 and *N* itself, when we run the algorithm, also other implementation details must be taken into consideration), and the probability of failure will be as small as we want (a similar theoretical analysis applies, as described in the previous section, but here we have a random walk with one absorbing barrier and one reflecting barrier).

4. Conclusion

For all practical purposes, we can assume that P = NP, even the conjecture P ≠ NP might be true, if we exclude randomized algorithms. It is also interesting to note that all financial transactions over the Internet are based on a cryptographic protocol that is not secure. In Reference [4] it is mentioned a somewhat similar algorithm for the 2-SAT problem, which is not NP complete. The generalization to 3-SAT and other NP complete problems and possible applications of these ideas were presented in this article.

5. Note

The symmetrical random walk model used here is just an approximation. In fact, we have to use a random walk with probabilities varying from place to place. This more exact model could lead to more different conclusions than presented here, in this article.

REFERENCES

[1] J. E. Hopcroft and J. D. Ullman, "Introduction to Automata Theory, Langiages, and Computation," Addison-Wesley Publishing Company, Cambridge, 1979.

[2] W. Feller, "An Introduction to Probability Theory and Its Applications," John Wiley & Sons, New York, 1968.

[3] L. Fortnow, "The Golden Ticket, P, NP, and the Search for the Impossible," Princeton University Press, Princeton, 2013.

[4] C. H. Papadimitriou, "On Selecting a Satisfying Truth Assignment," *Proceedings of the* 32*nd Annual Symposium on Foundations of Computer Science*, San Juan, 1-4 October 1991, pp. 163-169.