An approach towards the solution of NP-Complete Problem

Hari Om Sharan[1], Rajeev Kumar[1] and Vikas Solanki[2]

[1]Computer Science Department College of Engineering
Teerthanker Mhaveer University, Moradabad.(India)
[2]Computer Science Department, IET, Mangalayatan University, Aligarh, (India).
E-Mail: **sharan.hariom@gmail.com**, **rajeevphd@hotmail.com**

Abstract: DNA Computing is an alternative method for computations. It is based on the observation that in general it is possible to design of series of biochemical experiments involving DNA molecules which is equivalent to processing information encoded in these molecules. Cook's Theorem tells that if one algorithm for an NP-complete or an NP-hard problem will be developed, then other problems will be solved by means of reduction to that problem. The minimum vertex cover problem is a classic graph optimization problem and has been shown to be NP-Complete. In this paper, we propose a DNA algorithm for solving the *minimum vertex-cover problem.*

[Hari Om Sharan, Rajeev Kumar and Vikas Solanki. An approach towards the solution of NP-Complete Problem. Researcher. 2011;3(6):49-52]. (ISSN: 1553-9865). http://www.sciencepub.net.

Keywords: DNA Computing; NP-complete problems; NP-hard problems; Minimum Vertex Cover Problem; Cook's Theorem.

INTRODUCTION:

DNA computing is one interdisciplinary research area that is growing fast since DNA molecules are implemented in a computational process. One of the main objectives of this research area is to produce, in near future, a biologically inspired computer based on DNA molecules to replace or at least beneficially complement with a silicon based computer. Since R. Feynman has suggested to construct a computer from molecules in 1964 [1]. It took 30 years till Adleman in 1994 making proof of the principle study that DNA molecules can solve an NP problem of Hamiltonian Path Problem (HPP) through bio-chemical procedure [2].

DNA is a basic storage medium for all living cells. The main function of DNA is to absorb and transmit the data of life for billions years. Roughly, it is around 10 trillions of DNA molecules could fit into a space, the size of a marbles. Since all these molecules can process data simultaneously, theoretically, we can calculate 10 trillions times simultaneously in a small space at one time. DNA computing is more generally known as molecular computing. It is interdisciplinary field where it is a combination of biology, chemistry, mathematics and computer science. Computing with DNA offers a completely new paradigm for computation. The main idea of computing with DNA is to encode data in a DNA strand form, and laboratory techniques of molecule biology, called as bio-operations, will be involved to manipulate DNA strands in a test tube in order to simulate arithmetical and logical operations. It is estimated that a mix of 1018 DNA strands could operate 104 times faster than the speed of a today's advanced supercomputer [3].

Vertex cover is NP-complete; we don't expect to find a polynomial time algorithm for finding a minimum size vertex cover. The size of a vertex cover produced by the approximation algorithm is at most twice the minimum size of a vertex cover. The vertex cove problem is to find a vertex cover of minimum size in a given undirected graph. We call such a vertex cover an optimal vertex cover. This problem is the optimization version of an NP-complete decision problem.

Given the common belief that NP-hard optimization problems cannot be solved exactly in polynomial time, much research has been devoted in the past twenty years to drive efficient approximation algorithm. i.e. algorithms that deliver solutions whose value is guarantee to be within some multiplicative factor from the optimum. In order to evaluate the performance guarantees of such approximation algorithms, it is important to understand how far we can go. i.e. to prove, for any approximable problem, which is the best approximation achievable in polynomial time.

In the general case, a very simple 2-approximate algorithm has been known for thirty years [4], and no better approximation algorithm has been found until now. Slightly better approximation guarantees are achievable over bounded degree graphs [5]. On the negative side, the minimum vertex cover problem has been shown to be Max SNP-hard even when restricted to graphs with maximum degree 3 by Papadimitriou and Yannaakasis [6]. Their reduction is from MAX 3-SAT and uses explicit construction of expander graphs [7]. Combining this reduction, the non-approximability results by Ballare et al. [8] and the best known explicit construction of expanders [9], one can show that Minimum vertex cover is not 1.00036-approximableon bounded degree graphs. Ballare et al. [8] give a 1.0688 lower bound for the general minimum vertex cover problem by using a different technique, namely, they reduce directly from the computation of a verifier using a somehow “complementary” version of the FLGSS reduction [10]. However, their method does not apply when classes of graphs in which a fixed bound on the maximum degree or some other density constraints are considered.

Minimum Vertex Cover Problem: The minimum vertex cover problem arises in various important applications, including in multiple sequence alignments in computational biochemistry. Several approaches, such as the use of a parameterized algorithm [11] and the use of a simulated annealing algorithm [12], have been developed to solve this problem. Since DNA computing, which uses parallel computing, can be used to solve large problems, this study introduces an alternative molecular computing approach to solve the minimum vertex cover problem.

A vertex cover for a graph *G* is a set of vertices *V* so that every edge of *G* is incident to at least one vertex in *V*. Namely, *V* covers the edges of *G*. The Minimum Vertex Cover problem is to find the minimum set of vertices that cover all edges. Given an undirected graph *G* = (*V*, *E*), $m=|V|$ and $n=|E|$ are defined as the numbers of vertices and edges, respectively. A vertex edge incidence matrix A=(a_{ij}) of *G* is defined as $a_{ij}=1$ if edge *j* is incident to vertex *i*; otherwise $a_{ij}=0$, with $i=1,\ldots,m$; $j=1,\ldots,n$. The Minimum Vertex Cover problem can be stated as follows.

→ minimize $\sum_{i=1}^{n} xi$

subject to $\sum_{i=1}^{n} aijxi \geq 1$ and $x_i \in \{0,1\}$, with $I = 1,\ldots,m$; $j = 1,\ldots,n$

Vertex Cover Problem is NP-Complete:

A ***vertex cover*** of an undirected graph $G = (V, E)$ is a subset $V' \in V$ such that if $(u, v) \in E$, then $u \in V'$ or $v \in V'$ (or both). That is, each vertex "covers" its incident edges, and a vertex cover for *G* is a set of vertices that covers all the edges in *E*. The size of a vertex cover is the number of vertices in it.

The vertex-cover problem is to find a vertex cover of minimum size in a given graph. Restating this optimization problem as a decision problem, we wish to determine whether a graph has a vertex cover of a given size *k*.

Since VERTEX-COVER is NP-complete, we don't expect to find a polynomial-time algorithm for finding a minimum-size vertex cover.

DNA algorithm for Vertex Cover Problem:

The following DNA algorithm is proposed to solve the vertex-cover problem:

Step 1: Encoding of the problem in DNAs

Encoding the vertices: For each vertex, synthesize a random 10-based palindrome DNA strand where Vi represents the i^{th} vertex.

Encoding the edges: For each directed edge *Vi* *Vj* , synthesize a 10-base DNA strand consisting complementary of 3’ 5-mer sequence of *Vi* and complementary of 5’ 5-mer sequence of *Vj* .

Each vertex *Vi* in the graph has to be associated with a designed palindrome 10-mer sequence of DNA denoted by *Vi*. For each edge *Vi* *Vj* in the graph, an oligonucliotide 3’ 5-mer complementary sequence of Vi followed by 5’ 5-mer complimentary sequence of *Vj* to be synthesized.

Step 2: Create an empty set for vertices and take a copy of edge set

After the completing of step 1 (i.e. encoding the vertices and edges) we create an empty set for vertices (V) and edges (E) after that we will create a copy of edge set (E').

Step 3: Repeatedly picks an edge (*Vi*, *Vj*) from the copy of edge set

To avoid repetition of the nodes in the DNA strands an effective method, SSCP has to be used. The mobility in gel electrophoresis of double stranded DNA's of a given length is relatively independent of nucleotide sequence. In contrast, the mobility of single strands can vary considerably as a result of only small changes in nucleotide sequence. This fact led to the development of single-stranded conformation polymorphism (SSCP) techniques [13]. SSCP is the simplest and most used method of mutation detection. PCR is used to amplify the region of interest and the resultant DNA can be separated as single-stranded molecules by electrophoresis in a no denaturing polyacrylamide gel. A strand of single-stranded DNA folds differently from another if it differs by a single base, and it is believed that changes of structure of the DNA results in different motilities for the two strands. These mutations can be detected as the appearance of new bands on auto radiograms (radioactive detection), by silver staining of bands or the use of fluorescent PCR primers which are subsequently detected by an automated DNA sequencer (non-radioactive detection). Since all the nodes encoded are palindrome DNA strands, repetition of the nodes can lead to formation of Hairpin loop structures [14]. These hairpins like structures have to be eliminated from single stranded DNA strands by the above process. The hairpin loop strands are not considered for deriving the solution.

Step 4: Amplification of DNA paths by PCR

Amplification of DNA paths that begin with vertex source and end with vertex destination to be performed. Two specific primers that can anneal with source vertex and destination vertex are to be added to the PCR reaction.

Step 5: Mark the endpoints *Vi* and *Vj* to empty set

In step 4 we picked an edge (*Vi*, *Vj*) from the copy of edge set (E'), here we mark its endpoints say *Vi* and *Vj* to the set of vertices which was created in step 2 as an empty set.

Step 6: Delete the edges from the copy of edge set.

In this step we will remove all the edges from the copy of edge set (E') that are covered by either vertex *Vi* or vertex *Vj*. And this process will repeat until the copy of edge set will be empty.

Step 7: Sequencing of DNA strands

The strands obtained in the step 5 are now to be sequenced. The weights of the strands are determined by reading the sequence. The set having the minimum number of vertices that covers entire graph is our desired solution.

Concluding Remark:

This paper has proposed a faster approach for finding the solution for minimum vertex cover problem using DNA Computing. Because the DNA Computing, due to its high degree of parallelism, can overcome the difficulties that may cause the problem intractable on silicon computers, however using DNA computing principles for solving simple problems may not be suggestible. To make the DNA computing applicable in practice further research in both fields- Computer science and biology – is necessary. Computer science needs to develop more elaborate DNA algorithms, while better enzymes and protocols are needed to from biology to manipulate DNA molecules more selectively with minimal errors.

References:

[1] R. P. Feynman, Miniaturization, New York, Reinhold, pp.282-296, 1961

[2] L. M. Adleman, Molecular computation of solutions to combinatorial problems, Sciences, vol.266, no.5187, pp.1021-1024, 1994.

[3] L. Kari, From micro-soft to bio-soft: Computing with DNA, Biocomputing and Emergent Computation: Proc. of the BCEC97, World Scientific, Skovde, Sweden, pp.146-164, 1997.

[4] F. Gavril. Manuscript cited in [18], 1974.

[5] B. Monien and E. Speckenmeyer. Some further Approximation algorithms for the vertex cover problem. In proceedings of CAAP83, pages 341-349. LNCS 159, Springer Verlag, 1983.

[6] C. H. Papadimitriou and M. Yannakakis. Optimization, Approximation and Complexity classes. Journal of Computer and System Sciences, 43:425-440, 1991. Preliminary version in Proc. of STOC'88.

[7] O. Gabber and J. galil. Explicit construction of linear sized super concentrators. Journal of Computer and System Sciences, 22:407-425, 1981.

[8] M. Bellare, O. Goldriech and M. Sudan. Free bits, PCP's and non-approximability-towards tight results (3rd version). Technical Report TR95-24 Electronic Colloquium on Computational Complexity, 1995. Priliminary version in Proc. of FOCS'95.

[9] A. Lubotzky, R. Philips and P. Sarnak. Ramanujan Graphs. Combinatorica, 8:261-277, 1988.

[10] U. Feige, S. goldwasser, L. lovasz, S. safra and M. Szegedy. Approximating clique is almost NP-complete. In Proceeding of the 32nd IEEE

Symposium on Foundations of Computer Science, pages 2-12, 1991.

[11] R.G. Downey and M.R. Fellows, “Fixed parameter tractability and completeness II: completeness for W[1] Theory,” Comp. Sci., vol. 141, 1995, pp.1-2.

[12] X. Xu and J. Ma, “An efficient simulated annealing algorithm for the minimum vertex cover problem,” Neurocomputing, vol.69, 2006, pp. 913-916.

[13] Martyn Amos, Gheorghe Paun, Grzezorg, Rozenberg and Arto Salomaa, “Topics in the theory of DNA computing,” Theoretical computer science, 287, 2000, 3-38.

[14] K. Sakamoto, H. Gouzu, K. Komiya, D. Kiga, S.Yokoyoma, T. Yokomori and M. Hagiya, “Molecular computation by Hairpin formation,” Science, 288, 2000, 1223-1226.

5/18/2011