Molecular Solution to Minimum Vertex Cover Problem Using Surface-based DNA Computation

Chung-Wei Yeh  
Dept. of Management Information System  
Kao Yuan University,  
Kaohsiung, Taiwan 821, ROC  
e-mail: alice@cc.kyu.edu.tw

Kee-Rong Wu  
Dept. of Marine Engineering  
National Kaohsiung Marine University  
Kaohsiung, Taiwan 811, ROC  
e-mail: krwu@mail.nkmu.edu.tw

Abstract—The minimum vertex cover (MVC) problem is a classic graph optimization problem and has been shown to be NP-complete. For an undirected graph \( G, G = (V, E) \) where \( V \) and \( E \) represent the set of vertices and edges, the MVC problem is to find the smallest subset \( V^* \subset V \) such that \( V^* \) contains at least one of the two endpoints of each edge in \( E \). This study finds molecular solutions to the MVC problem in three steps. In the first step, DNA strands are defined to represent the vertices; in the second step, the DNA parallel approach is adopted to construct the algorithm; and in the third step, the solution is read by fluorescence-image technique. The proposed approach applies mostly the surface-based computation model that was presented by Liu. The results demonstrate the capacity of molecular computing for solving the complex vertex cover problem.

Keywords—graph optimization problem; DNA Computing; surface-based DNA computation; minimum vertex cover (MVC) problem

I. INTRODUCTION

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 [1] and the use of a simulated annealing algorithm [2], 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 MVC problem.

Adleman [3], who solved a Hamiltonian path problem (HPP) for a directed graph with seven nodes, established the efficacy of the use of molecules in solution to solve computational problems. Lipton solved a satisfiability (SAT) problem to demonstrate the advantages of exploiting the massive parallelism that is inherent in DNA-based computing. In 1996, Liu et al. [4] presented a molecular surface-based experimental solution to a SAT problem. Accordingly, the solution-based and surface-based methods have been the main models adopted to molecular computation. Instances of the use of DNA computing to solve NP-complete problems reveal that not only can it efficiently solve difficult problems, but also parallel computation has great potential in DNA computing. However, the solution-based method is associated with poor efficiency of purification and a high error rate in the separation steps [4]. The alternative “surface-based” computation method, which involves manipulating the DNA strands that are immobilized on a surface, reduces the loss of DNA molecules during purification. The proposed approach involves the incubation of the solution space that contains the DNA strands and presents a DNA-based algorithm to yield a general solution that is based on the operations defined in Liu’s experiments. The rest of this study is organized as follows. Section II briefly describes the MVC problem. Section III introduces the surface-based DNA computation and presents the DNA algorithm that is used to solve MVC problem. Section IV employs the proposed algorithm to an example. Section V draws conclusions.

II. MINIMUM VERTEX COVER (MVC) 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 MVC 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 MVC problem can be stated as follows.

\[
\text{minimize } \sum_{i=1}^{m} x_i .
\]

subject to \( \sum_{j=1}^{n} a_{ij} x_j \geq 1 \) and \( x_i \in \{0,1\} \), with \( i = 1, \ldots, m; j = 1, \ldots, n \)

III. SURFACE-BASED DNA COMPUTATION

DNA computing that embeds massive parallelism may become a strong alternative to electronic computing since barriers exist to the continued development of traditional silicon-based computers [5]. The mechanism behind the proposed approach for solving MVC is based on the strand design and biological reactions of DNA. A relatively short single-stranded DNA molecule (an oligonucleotide or simply “oligo”) of unique sequence is designed to represent each vertex. Initially, suitable DNA strands are present on a surface, and short single-stranded oligos can be brought...
together and then covalently bonded with selected polarity into longer single-stranded DNA molecules by separating the temporarily generated complementary (double-stranded) base-pairing by thermal denaturation, to represent the all possible vertex cover combinations. A VexCovDNA algorithm, using only practical operations, is then designed to eliminate the vertices that do not cover specific edge(s) from solution space. Accordingly, the solution vertex sets can be produced and the vertex sets with the minimum number of vertex covers can be obtained by further comparison. Two fundamental assumptions in DNA computation are that DNA strands can encode data, and that molecular biological operations can be adopted to perform all computational operations.

A. Background of DNA computation

The DNA molecule is the basis of DNA-based computation [6]. Although the physicist Feynman first proposed the construction of sub microscopic computers in 1961 [7], only in 1994 did Adleman succeed in manipulating DNA strands to solve an HPP in a test tube. Accordingly, various DNA algorithms have been developed to solve NP-complete problems. Section III.B describes the structure of DNA and the basic annealing operation of DNA. Section IIIC presents the operations of surface-based molecular computation. Section IIID describes the VexCovDNA algorithm.

B. The structure and annealing operation of DNA

DNA (deoxyribonucleic acid) is a linear polymer of nucleotide monomer units. Each nucleotide is comprised of a heterocyclic base attached to a deoxyribose sugar. Consecutive nucleotides in a DNA strand are linked by a phosphodiester bond between the 3’ hydroxyl (OH) group on one sugar and the 5’ hydroxyl group on the next sugar. The polarity of a single DNA strand is determined by the fact that at one end it has a terminal nucleotide with a free 5’-OH (and is hence the 3’-end), whereas the opposite end of the strand has a terminal nucleotide with a free 3’-OH (and is hence the 3’-end).

The information content of DNA is encoded by the 5’-3’ sequence of consecutive bases in the polymer, which may be either adenine (A), cytosine (C), guanine (G), or thymine (T). By convention, the 5’-3’ sequence of bases in DNA is written from left-to-right, unless indicated otherwise. Information replication and transfer via DNA can be achieved because DNA naturally forms a double-stranded molecule. This occurs as a result of specific, reversible hydrogen bond formation between sterically complementary ‘base pairs’, either {A and T} or {C and G}, that directly face each other but are on the two oppositely oriented strands of DNA that make up the double-stranded molecule (Fig. 1b). Thus, the two strands in double-stranded DNA are referred to as the reverse-complement (or simply the complement) of each other. DNA computations commonly apply a specific sequence of biological operations in tubes or on surfaces to solve a problem. A set of molecules of DNA strands can be represented by a multi-set of finite strings over \{A, C, G, T\}. When strands are cooled, annealing allows complementary strands to bind together (Fig. 1c).

C. Operations in surface-based DNA computation

- **MARK**: Strands are marked simply by making them double-stranded at the free end. Under suitable conditions, single-stranded DNA hybridizes, or anneals, to form a double-stranded DNA molecule with its Watson-Crick complement.
- **UNMARK**: This is done simply by washing the surface in distilled water and raising the temperature if necessary. In the absence of salt, which stabilizes the double-stranded pairs, the complementary strands of DNA denature from the oligos on the surface and are washed away, leaving only the original single-stranded DNA attached to the surface.
- **DESTROY**: Single-stranded DNA molecules are destroyed using enzymes known as exonucleases. For instance, the enzyme Exonuclease I can be used to destroy single-stranded oligos.

D. DNA algorithm for solving MVC problem

This section presents a surface-based approach for DNA computation, which is shown diagrammatically in Fig. 2, to solve the MVC problem; \( v_i \) represents an identical vertex.

Designed to find the minimum number of vertex covers, the VexCovDNA defines each vertex that is represented by a particular short strand; in which at least four bases mismatch each other. Based on the number of edges \( n \), the program then executes an \( n \)-loop to exclude the strands that contain the vertex that does not cover the current processing edge, known simply as “disqualified strands”; and then finds the minimum number of vertices. The program can thus be represented as:
The computation begins in the solution space T, which represents all possible vertex cover combinations, by applying Generate(T). The construction of T and each step in the algorithm are as follows. Since each edge e_k connects only two vertices, \(<v_i,v_j>\) is defined as known. For a G with n vertices, 4n short oligos are synthesized in Step 1. These are \(v_1, v_2, ..., v_m, \bar{v}_1, \bar{v}_2, ..., \bar{v}_n\), and \(v_1', v_2', ..., v_n'\). The oligo \(v_i\) indicates that vertex i covers the current processing edge; \(v_i'\) indicates that it does not. The oligos \(\bar{v}_i\) and \(\bar{v}_i'\) are the complements of \(v_i\) and \(v_i'\), respectively. After the 4n short oligos have been synthesized, a set of long oligos is generated by combinatorially concatenating vertices, as presented in Fig. 2. Light-directed spatially addressable parallel chemical synthesis method of Fodor et al. [8] can be adopted to represent 2^n possible vertex cover sequence on the surface. Step 2 attaches these strands on a surface. Steps 1 and 2 synthesize all possible solutions (combinatorial set of single-stranded DNA) and immobilize them in an unaddressed surface. Since the vertices \(<v_i,v_j>\) for each e_k are given, inside the n-loop in Step 3, for each edge e_k, the corresponding complementary strands of \(<v_i,v_j>\) are added to the surface in the MARK operation. Therefore, the strands that represent vertices that cover the \(e_k\) are hybridized with the complementary strand that is tagged with a fluorescent label. With an appropriate choice of enzyme, the DESTROY operation is then applied to keep the double-stranded molecules while the single-stranded oligos are enzymatically digested. The strands are not destroyed, indicating that the vertices that are represented by the specific strands cover the corresponding edge. The UNMARK operation is then applied to separate the double-strands into single-strands by thermal denaturation, and the next edge within the loop is checked and so on. At the completion of this step, Readout is performed using the fluorescence-image technique. Whether the strands are bright is determined. Observing their color and recording all positive reactions on the surface enable the number of vertices that cover all edges, corresponding to each solution, to be identified. The minimum number of vertex covers is determined by comparing these values. Furthermore, the time complexity of VexCovDNA is \(O(n^3)\). A common concern in the evaluation of the time complexity of DNA algorithms is that every DNA operation requires a particular amount of time to implement. The construction of the initial sets of strands and the reading out of the final solution are time-consuming.

**IV. Example**

Fig. 3 presents an example of MVC with three vertices and two edges. For three vertices \(V_i-V_j\) and two edges \(e_1\) and \(e_2\), 12 short oligos are defined; they are \(v_1, v_2, v_3, \bar{v}_1, \bar{v}_2, \bar{v}_3, v_1', v_2', v_3', \bar{v}_1', \bar{v}_2', \bar{v}_3'\). The oligo \(v_i\) represents the strand whose vertex \(V_i\) is chosen to cover the processing edge; \(v_i'\) represents that it does not. The oligos \(\bar{v}_i\) and \(\bar{v}_i'\) are the complements of \(v_i\) and \(v_i'\), respectively. In Step 1, the 12 short oligos are synthesized, and the vertices combinatorially concatenate a set of long oligos. The first segment of a long oligos is either \(v_i\) or \(v_i'\). The rest are either \(v_j\) or \(v_j'\). The set of all long oligos that corresponds to the set of combinatorial solution space of size eight is generated by the light-directed spatially addressable parallel chemical synthesis method of Fodor et al. [8].
Figure 3. Figure G with three vertices and two edges.

### TABLE I. RELATIONSHIP BETWEEN OLIGOS AND THE VERTEX-COVER SEQUENCE

<table>
<thead>
<tr>
<th>Strands</th>
<th>S₁</th>
<th>S₂</th>
<th>S₃</th>
<th>S₄</th>
<th>S₅</th>
<th>S₆</th>
<th>S₇</th>
<th>S₈</th>
</tr>
</thead>
<tbody>
<tr>
<td>V₁,V₂,V₃</td>
<td>000</td>
<td>010</td>
<td>001</td>
<td>011</td>
<td>100</td>
<td>101</td>
<td>110</td>
<td>111</td>
</tr>
</tbody>
</table>

In Step 2, the eight strands are attached to the surface. For clarity of explanation, a three-bit binary vertex sequence is designed to identify the oligo combination. If vertex \( V_i \) is chosen to cover the edge, bit \( i \) is “1”; otherwise, it is “0”. Table I presents the relationship between the eight strands and the vertex cover sequence. To determine \( e_1 \), the complements of \( v_1 \) and \( v_2 \) are added to the surface for the initial execution of the loop in Step 3. Therefore, \( \overline{v_1} \) and \( \overline{v_2} \) are added to the surface. In the initial cycle, \( S_1 \) and \( S_3 \) are destroyed. To determine \( e_2 \), the complements of \( v_1 \) and \( v_3 \) are added to the surface for the second time execution of the loop. Therefore, \( \overline{v_1} \) and \( \overline{v_3} \) are added to the surface. In the second cycle, \( S_2 \) is destroyed, yielding the feasible solution set \( S_4 \sim S_8 \). Strands \( S_5 \sim S_8 \) are then compared in Step 4. Observing the color of all positive reactions on the surface yields the number of covering vertices that correspond to each solution. The minimum number of vertex covers is determined by comparing these values. The strand that is finally encoded with the minimum number of vertex covers is strand \( S_5 \), with an encoded value of “100”, since \( S_5 \) has only one vertex, \( V_1 \), to cover all edges.

V. CONCLUSION

The importance of DNA computing technology in applications has increased significantly in recent years. The proposed algorithm, \( VexCovDNA \), demonstrates that the vertex cover problem can be solved using a DNA surface-based algorithm with favorable computational efficiency. The approach described herein mostly transforms vertices into strands, constructs a solution space, and then uses basic biological operations to generate a program that destroys disqualified strands. In this study, the \( VexCovDNA \) algorithm decreases the complexity of the computation linearly with time for the present \( n \) edges. Two major features of the proposed approach are as follows.

1. The time complexity of the proposed algorithm was determined. As presented in \( VexCovDNA \), the time complexity is \( O(n) \). The computational problems of significant complexity may be solved efficiently by the DNA computation.

2. The proposed approach uses only a few oligos. The numbers of unique oligos to be synthesized and added to surface to enable the MARK step to proceed are \( 4m \) and \( 2n \), respectively, where \( m \) and \( n \) are the numbers of vertices and edges.

Since the completeness of the algorithm was verified, this framework suggests that algorithmic approaches that use biological operations to DNA strands may also be extended to solve more NP-complete problems.

ACKNOWLEDGMENT

The authors would like to thank the National Science Council of Taiwan, ROC, for financially supporting this research under Contract No. NSC 97-2218-E-244-001.

REFERENCES