DNA Solutions to Searching for the Hamiltonian Cycle and the Traveling-Salesman Cycle*

Shaw-Jye Wu,† Yuan-Jye J. Wu,‡ and Joe J. Shaw§

October 4, 1996

Abstract

Computational problems for which an algorithm cannot be determined in polynomial time are classified as NP-complete problems. Because of their great capacity to conduct parallel reactions, DNA molecules and their experimental protocols have been proposed to solve such problems, which otherwise are time-consuming for electronic computers. Based on a working archetype models are presented here to compute two NP-complete problems: searching for the Hamiltonian cycle and the traveling-salesman cycle.

1 Introduction

Solving a problem not only correctly but also efficiently is one of the main goals of algorithm design and analysis in computer science. An algorithm is a collection of computational procedures that can be programmed on computers to solve a specific problem. A problem may have more than one algorithm for its solution, but the performance might vary according to different computer architectures or data layouts.

For example, consider the summation problem, $S(n)$, defined as

$$S(n) = a_1 + a_2 + \cdots + a_n,$$

where a_i is a real number for $i = 1, \ldots, n$. The number n is defined as the problem size. A natural way to solve this problem is to add those n numbers one by one. The operations required for such a solution are $n - 1$ additions. However, if a parallel computer with two processors is used, half of those numbers can be assigned to one processor and the rest of the numbers to the other processor. The final answer can be obtained by adding the two local sums resulting from applying the above method individually to each processor.

*This study was supported by funds provided by Mr. Charles Holman.

†Department of Botany and Microbiology, Auburn University, Auburn University, AL 36849, wushawj@mail.auburn.edu

‡Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, IL 60439, jwu@mcs.anl.gov

§Department of Botany and Microbiology, Auburn University, Auburn University, AL 36849, jshaw@ag.auburn.edu

This two-processor algorithm still requires $n-1$ additions, but the elapsed time can be theoretically cut to half of the time required by the primary solution (the actual elapsed time for the two-processor algorithm will be slightly longer than half of the first method's required time because there is a communication cost for sending the local sums between processors for the final addition).

Two conclusions can be derived from this example. First, using a parallel computer may be advantageous for some problems. Second, both algorithms are classified as polynomial time since the leading term, which dominates the total count, has the form αn^k, where k is an integer and both α and k are nonnegative and independent of the problem size n. A problem that can be solved by a polynomial-time algorithm is tractable for extant electronic-based computers.

However, not all problems are so readily addressed, including the class of problems known as NP-complete problems (the term "NP" is short for "nondeterministic polynomial time" [3, p. 927]). No polynomial-time algorithm has been discovered for an NP-complete problem. In our example, a parallel computer with a large number of processors may be a solution to get a polynomial elapsed time with respect to n even though there is no polynomial-time algorithm. Unfortunately, for current electronic-based parallel computers, more processors cause a greater cost in communication channels and synchronization.

Nevertheless, a totally different system that employs biomolecules may revolutionize computational procedures. In 1994, Adleman [1] first introduced an algorithm and a practical DNA computer to solve one type of NP-complete problem, the Hamiltonian path problem. Later, Lipton [8] described the DNA-based solution of another type of NP-complete problem, the SAT problem. These solutions demonstrated both the feasibility of solving computational problems by manipulating biological molecules and the potential usefulness of the massively parallel processing power derived from such an approach. In this article, we propose new algorithms to solve two other NP-complete problems: the Hamiltonian cycle problem and the traveling-salesman problem.

2 Problems and Algorithms

In this section, we introduce some required background about graphs, describe the problems, and propose our algorithms.

2.1 Graphs

A graph consists of two sets: vertex and edge. The vertex set contains a finite number of points, called vertices, and the elements of an edge set, called edges, defining relations on some pairs of vertices. There are two kinds of edges: directed and undirected. A directed edge permits only one direction of transit between a pair of vertices, whereas an undirected edge allows two-way transit between two vertices. Consequently, we have both directed and undirected graphs. Figure 1A shows a directed graph, and Figure 1B gives an example of an undirected graph. Both graphs have the same vertex set {1,2,3,4,5,6}.

A path is a sequence of vertices such that every pair of consecutive vertices has an edge. In Figure 1A, those edges with shadow show a path $< 5, 1, 4, 2 >$. A path is said to be

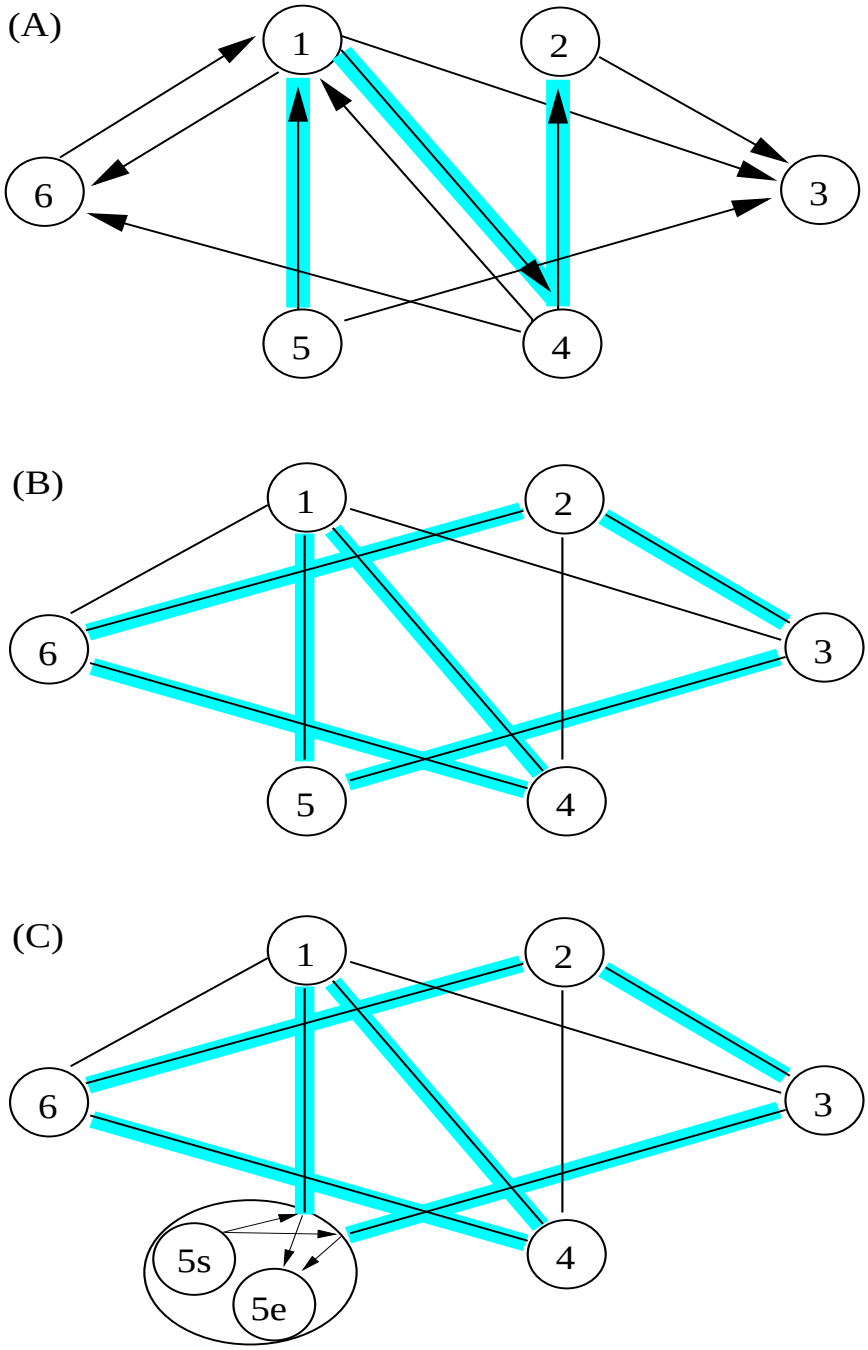

Figure 1: (A) A directed graph with a path $< 5, 1, 4, 2 >$ shown by shaded edges. (B) An undirected graph containing two Hamiltonian cycles, $< 5, 1, 4, 6, 2, 3, 5 >$ and $< 5, 1, 6, 4, 2, 3, 5 >$. The former one is shown by shaded edges. (C) To execute the first step of the algorithm, one of the vertices (vertex 5) is chosen and divided to form two fictitious vertices as starting (vertex 5s) and ending (vertex 5e) points.

simple if all the vertices in the sequence are distinct. Furthermore, a path is called a cycle if the first and the last vertices in the sequence are identical and the path contains at least one edge. Finally, a cycle is said to be simple if all the vertices in the sequence are distinct except the first and the last vertices.

Now we can describe the Hamiltonian cycle and traveling-salesman problem formally.

2.2 The Hamiltonian Cycle

A Hamiltonian cycle is a simple cycle of an undirected graph that contains all vertices of the graph [3, pp. 953–959]. In Figure 1B, the shadowed edges depict a Hamiltonian cycle of a graph. A graph that defines a Hamiltonian cycle is said to be Hamiltonian; not all graphs have a Hamiltonian cycle. The problem will be to decide if a given graph is Hamiltonian. If the answer is yes, we present a cycle. Note that whether every undirected edge in an undirected graph is replaced by two directed edges with opposite directions, then solution for the Hamiltonian cycle problem is similar to find a Hamiltonian path that starts from and ends at the same vertex of a directed graph. Therefore, the following algorithm based on Adleman's algorithm [1] is proposed for DNA-based computation to search for a Hamiltonian cycle:

Given an undirected graph with n vertices,

1. Pick one vertex, say v, as the starting and ending vertex.
2. Form all possible paths.
3. Collect paths that start from and end at the vertex v, and discard the rest. The resulting paths are cycles.
4. Collect cycles that contain n edges, and discard the rest.
5. Collect cycles that visit all vertices, and discard the rest.
6. Examine the result from Step 5. If the result is empty, the graph does not have Hamiltonian cycles, and the process stops here.
7. If the result is not empty, reveal the Hamiltonian cycles.

The graph shown in Figure 1B provides an example of the experimental design for implementing such an algorithm. The vertex 5 is chosen as the starting and ending vertex at Step 1. Because of the cyclic nature, the choice of starting/ending vertex can be random. This vertex will be virtually treated as two vertices, the starting one and the ending one (Figure 1C).

At the Step 2, for $i = 1, \ldots, 4$, and 6, oligonucleotide O_i of 20 bases is assigned to represent vertex i and its complementary strand is denoted as $\overline{O}_i$. As for vertex 5, two additional distinct oligonucleotides, O_{5s} and O_{5e}, as well as their complementary strands, are assigned to represent the practical starting and ending vertices respectively. For each edge connecting vertices i and j, oligonucleotides O_{i-j} and O_{j-i} are designed to represent the two-way traffic, wherein the traffic direction is denoted by the polarity of DNA molecule (Figure 2A). For transit from vertex i to vertex j, the 5' end of O_{i-j} is identical to either all 20 bases of O_{5s} for $i = 5$ or the 10 bases at the 3' end of O_i for all other values of

O_2	20-mer	5'	GGAACCTTGACTGGAACCTT	3'
			||||||||||||||||||||	
$\overline{O}_2$	20-mer	3'	CCTTGGAACTGACCTTGGAA	5'
O_3	20-mer	5'	GATCGATCGATCGATCGATC	3'
O_6	20-mer	5'	AGCTTAAGGCCGGAATTCGA	3'
O_{6-2}	20-mer	5'	CGGAATTCGAGGAACCTTGA	3'
O_{2-3}	20-mer	5'	CTGGAACCTTGATCGATCGA	3'

(A)

O_{6-2} $\downarrow$ O_{2-3}

CGGAATTCGAGGAACCTTGACTGGAACCTTGATCGATCGA

|||||||||||||||||||||

CCTTGGAACTGACCTTGGAA

$\overline{O}_2$

(B)

Figure 2: Vertices and edges are encoded in oligonucleotides for searching the Hamiltonian cycle. (A) 20-mer oligonucleotides O_i are assigned to each vertex (O_2, O_3 and O_6 are shown). Their complementary strands are $\overline{O}_i$ (shown is $\overline{O}_2$). For oligonucleotides O_{i-j} that encode the edges, the direction of transit from vertex i to vertex j is determined by using 10 bases from the 3'end of O_i (except when $i = 5$, in which case all 20 bases of O_{5s} are contained within) and 10 bases from the 5' end of O_j (unless $j = 5$, in which case it is the same as O_{5e}). Bases presented here are for descriptive purpose only. (B) Ligation event (arrow) between O_{i-j} can occur when oligonucleotides anneal properly.

i. Similarly, the 3' end of O_{i-j} is identical to either all 20 bases of O_{5e} for $j = 5$, or the 10 bases at the 5' end of O_j for all other values of j. O_{j-i} is prepared by the same principle as the counterpart of O_{i-j}. When the essential DNA molecules are ready, $\overline{O}_i$ and O_{i-j}/O_{j-i} are mixed in solution and allowed to anneal. $\overline{O}_i$ serve as bridges to bring the various edge oligonucleotides together. A subsequent ligation reaction then generates longer DNA fragments from all the possible combinations of the edge oligonucleotides (Figure 2B).

To implement Step 3, the products from the ligation reaction are amplified by PCR with the primer pair O_{5s} and $\overline{O}_{5e}$. The signals encoded by DNA fragments starting from O_{5s} and ending at O_{5e} are boosted and can be detected visually after agarose gel electrophoresis. Since the graph has only six vertices, seven 20 bp oligonucleotides (corresponding to five original vertices and the two virtual start and stop vertices) are expected to be included in the desired DNA fragments. Thus, at Step 4, those amplified products of 140 base pairs in length will be excised and electroeluted.

The DNA fragments thus collected meet the requirements at Steps 3 and 4: specifically, they all begin with and end at vertex 5 and contain six edges. Any of the fragments that contain all O_i for $i = 1, \ldots, 4$, and 6 will represent one Hamiltonian cycle. Such DNA fragments are extracted by affinity purification. Single-stranded DNA fragments generated from the electroeluted fragments are annealed to biotinylated $\overline{O}_i$. Those fragments that hybridize to biotinylated $\overline{O}_i$ reveal the presence of O_i and are recovered by streptavidin paramagnetic particles. The extraction procedure is re-employed by using the newly isolated O_i-containing DNA with the next biotinylated $\overline{O}_j$. Purification Steps with $\overline{O}_{5s}$ and $\overline{O}_{5e}$ are not required because the prior PCR amplification included them, by definition. The final extract can be amplified again by using the primer pair O_{5s} and $\overline{O}_{5e}$ to create enough molecules for further analysis.

At Step 6, final amplified products are electrophoresed on an agarose gel. If no DNA band is detected, the graph is not Hamiltonian. If a 140 bp DNA band is found, at least one Hamiltonian cycle exists, and the following procedure is implemented.

Multiple graduated PCRs are employed at Step 7 to reveal the cycle. For the first graduated PCR, O_{5s} serves as forward primer and $\overline{O}_i$ for $i = 1, \ldots, 4$, and 6, are used as reverse primers. Note that the amplified DNA fragments with a length of 40 bp denote those vertices that are adjacent to vertex 5. Furthermore, for each Hamiltonian cycle, there will be a pair of affinity purified DNA fragments with opposite orders (e.g., $< 5s, 1, 4, 6, 2, 3, 5e >$ and $< 5s, 3, 2, 6, 4, 1, 5e >$ in Figure 1B). Therefore, at least two distinct fragments with length 40 bp will be formed after the first graduated PCR (vertices 1 and 3 in our example). Users can select either order through choice of the next adjacent vertex.

Suppose that vertex 1 is selected, O_{5s-1} will serve as the forward primer in the second round of graduated PCR, with reverse primers $\overline{O}_2, \overline{O}_3, \overline{O}_4, \text{and} \overline{O}_6$. Amplified DNA fragments with a length of 60 bp will denote those vertices that are adjacent to vertex 1 (vertices 4 and 6 in our example, see Figure 1B). Again, we can pick either O_{1-4} or O_{1-6} as the forward primer for the third round of graduated PCR. If there is only one fragment type with a length of 50 bp, it implies that the adjacent vertex is unique. The search then progresses to those amplified DNA fragments with a length of 70 bp. Eventually, the sizes will reveal the order of a Hamiltonian cycle starting from vertex 5.

2.3 The Traveling-Salesman Problem

Now let us move to the traveling-salesman problem. This problem is a variant of the Hamiltonian cycle problem with more sophisticated conditions. We start with an undirected complete graph in which each pair of vertices are connected each other. The edges of this graph are weighed; that is, an integer is assigned to each edge to represent the cost of travel between the vertices connected by the edge. The total cost of a cycle is the sum of those individual weights of the edges in the cycle. The problem is to find a Hamiltonian cycle of a given graph with the minimum cost [3, pp. 959–960]. An algorithm similar to the one above but with several modifications is proposed to solve the traveling-salesman problem.

Imagine a complete, weighted, and undirected graph with n vertices.

1. Pick one vertex, v, as the starting and ending vertex.

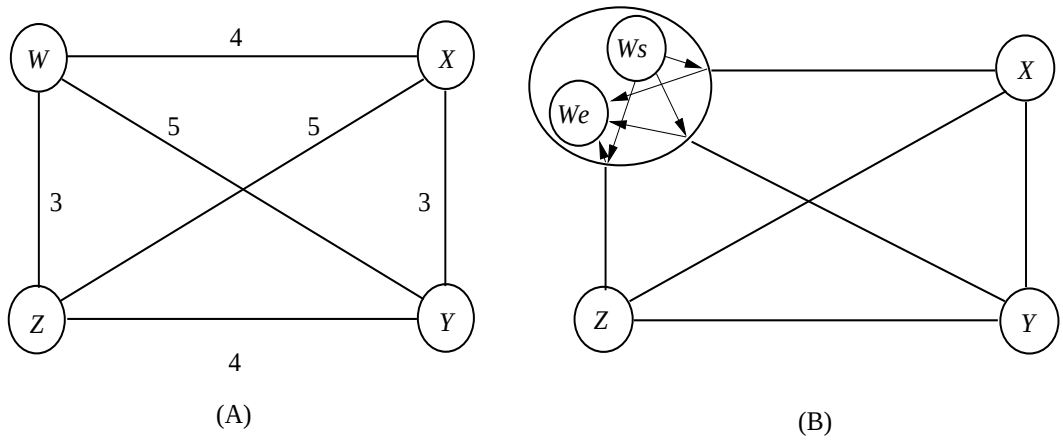

Figure 3: (A) An undirected graph showing the traveling-salesman problem. The digits indicate the relative costs of individual edges between vertices. (B) To facilitate the computation, vertex W is denoted twice, Ws and We, for starting and ending point respectively.

2. Form all possible paths.
3. Collect those paths that start from and end at the vertex v, and discard all other paths. The resultant paths are cycles.
4. Collect cycles that visit all vertices, and discard all others.
5. Collect those cycles with the minimum cost to reveal a traveling-salesman cycle.

The graph shown in Figure 3A is used as an example. At the first step, vertex W is randomly chosen and practically treated as two vertices, Ws and We, to accommodate the dual roles as a starting and ending point (see Figure 3B).

At Step 2, distinct oligonucleotides (O_i) of 18 bases are assigned to represent each vertex, and their complementary strands are denoted as $\overline{O}_i$. However, a more complicated design of the oligonucleotides O_{i-j}/O_{j-i} is required to represent the weighted edges. Basically, they all composed of three segments. The 5' end of O_{i-j} consists of 9 bases, which are identical to the 3' end 9 bases of O_i (except when $i = W$, all 18 bases of O_{Ws} are contained within), while the 3' end contains another 9 bases exactly as same as the 5' end 9 bases of O_j (unless $j = W$, in which case it is all of O_{We}). The third segment of DNA is of variable length and is positioned in the middle of the oligonucleotide. Consider the cost or length ratio depicted in Figure 3A. If the minimum length of the edge is 18 bases (the cost is 3), the other two costs can be represented by oligonucleotides of 24 (cost = 4) and 30 (cost = 5) bases. Hence, for the shortest edges like O_{X-Y}, the third DNA segment contains no extra nucleotides. But, 6 or 12 bases are included for longer edges. Since this additional DNA segment merely represents part of the weighted cost of each individual edge, the sequences can be random (Figure 4A). As described above, O_{i-j} designates one-way traffic from vertex i to vertex j. Both O_{i-j} and O_{j-i} are prepared for each undirected edge.

O_X	18-mer	5'	GGAACCTTTGGGAACCTT	3'
			||||||||||||||||||	
$\overline{O}_X$	18-mer	3'	CCTTGGAAACCCTTGGAA	5'
$\overline{O}_Y$	18-mer	3'	GATCGATCGATCGATCGA	5'
$\overline{O}_Z$	18-mer	3'	TAGCTTAAGCTAGAGCTG	5'
O_{X-Y}	18-mer	5'	GGGAACCTTCTAGCTAGC	3'
O_{Y-Z}	24-mer	5'	TAGCTAGCTNNNNNNATCGAATTC	3'
O_{Z-X}	30-mer	5'	GATCTCGACNNNNNNNNNNNNGGAACCTTT	3'

(A)

```
          O_{X-Y}        ↓              O_{Y-Z}
      GGGAACCTTCTAGCTAGCTAGCTAGCTNNNNNNATCGAATTC
      |||||||||||||||||||||||||||||      |||||||||
CCTTGGAAACCCTTGGAAGATCGATCGATCGATCGA      TAGCTTAAGCTAGAGCTG
      O̅_X          ↑        O̅_Y                    O̅_Z
```

(B)

Figure 4: Vertices and edges are encoded in oligonucleotides for searching the traveling-salesman cycle. (A) 18-mer oligonucleotides O_i are assigned to each vertex (O_X is shown). Their complementary strands are $\overline{O}_i$ (shown are $\overline{O}_X, \overline{O}_Y$ and $\overline{O}_Z$). For oligonucleotides O_{i-j} that encode the edges, the direction of transit from vertex i to vertex j is determined by using 9 bases from the 3' end of O_i (except when $i = W$, in which case all 18 bases of O_{Ws} are contained within) and 9 bases from the 5' end of O_j (unless $j = W$, in which case it is the same as O_{We}). The cost of edges is adjusted by additional bases wedged between both ends (shown are O_{X-Y}, O_{Y-Z}, and O_{Z-X}). Bases presented here are for descriptive purpose only. (B) Ligation events (arrows) between O_{i-j} or $\overline{O}_i$ can occur when oligonucleotides anneal properly. A gap between $\overline{O}_Y$ and $\overline{O}_Z$ caused by additional bases remains open.

Once again, all essential DNA molecules (O_{i-j}/O_{j-i} and $\overline{O}_i$) are mixed and allowed to anneal. A ligation reaction generates longer DNA fragments from all possible combinations of edges oligonucleotides, though not all $\overline{O}_i$ are ligated together because of the gaps created by the additional DNA segments in some O_{i-j} (Figure 4B). Those DNA fragments that start from and end at vertex W are then amplified by PCR with primer pair O_{Ws} and $\overline{O}_{We}$ (Step 3).

In the Hamiltonian cycle problem, amplified DNA fragments resulting from Step 3 are screened on an agarose gel to isolate those sizes corresponding to the sum of vertices. However, because of the variant lengths of O_{i-j}, the size of desired DNA fragment here is not predictable, and therefore affinity purification is conducted directly at Step 4. DNA fragments purified in this step are amplified again by using primer pair O_{Ws} and $\overline{O}_{We}$ to produce a large enough quantity for further analysis.

In the case presented here, six DNA fragments encoding three Hamiltonian cycles are contained in the affinity purified DNA pool (Figure 5). The sizes are 102 bp, 114 bp and 126 bp. At Step 5, agarose gel electrophoresis separates them, and the 102 bp DNA fragments that carry the desired information (the least cost) can be isolated. Of course, many DNA fragments encoding paths other than these six will also exist in the affinity purified DNA pool. However, these fragments are not a concern. They encode multiple visits to some vertices so that they have larger sizes, and thus can be eliminated by the electrophoresis step.

To consummate the final step, multiple graduated PCRs are again employed. In the first round of graduated PCR, the smallest DNA fragment whose size is 36 bp is amplified with primers O_{Ws} and $\overline{O}_Z$. This process provides the information that one of the two vertices adjacent to vertex W in the traveling-salesman cycle is vertex Z. This information is sufficient for conducting the second round of graduated PCR, and the other vertex next to W (vertex X in this case) is irrelevant. In the second round of graduated PCR, DNA fragments with sizes of 60 and 78 bp are amplified using O_{Ws-Z} as the forward primer and $\overline{O}_Y$, $\overline{O}_X$ as the reverse primers, respectively. Amplified DNA fragments are then run on an agarose gel, and the traveling-salesman cycle is revealed.

3 Discussion

Advantages of molecular computation include energy efficiency and high information storage density [1]. The true power of the DNA computation, however, resides in its massive parallel capabilities and unique screening strategies. One Weiss unit of T4 DNA ligase yields approximately 4.8×10^{13} concatenating events in 30 minutes, or 2.7×10^{10} events per second. Thus an immense amount of combination results can be generated in parallel within a short time when scaled up properly. In addition, PCR amplification and affinity purification allow identification of the answer to be straightforward. A conventional electronic-based single-processor computer must employ try-and-error tactics for these nonpolynomial-time algorithms, attempting first one path and then another in a labyrinth of possibilities, finally deciding the answer. In contrast, a DNA-based computer constructs all path combinations simultaneously and picks out the answer directly.

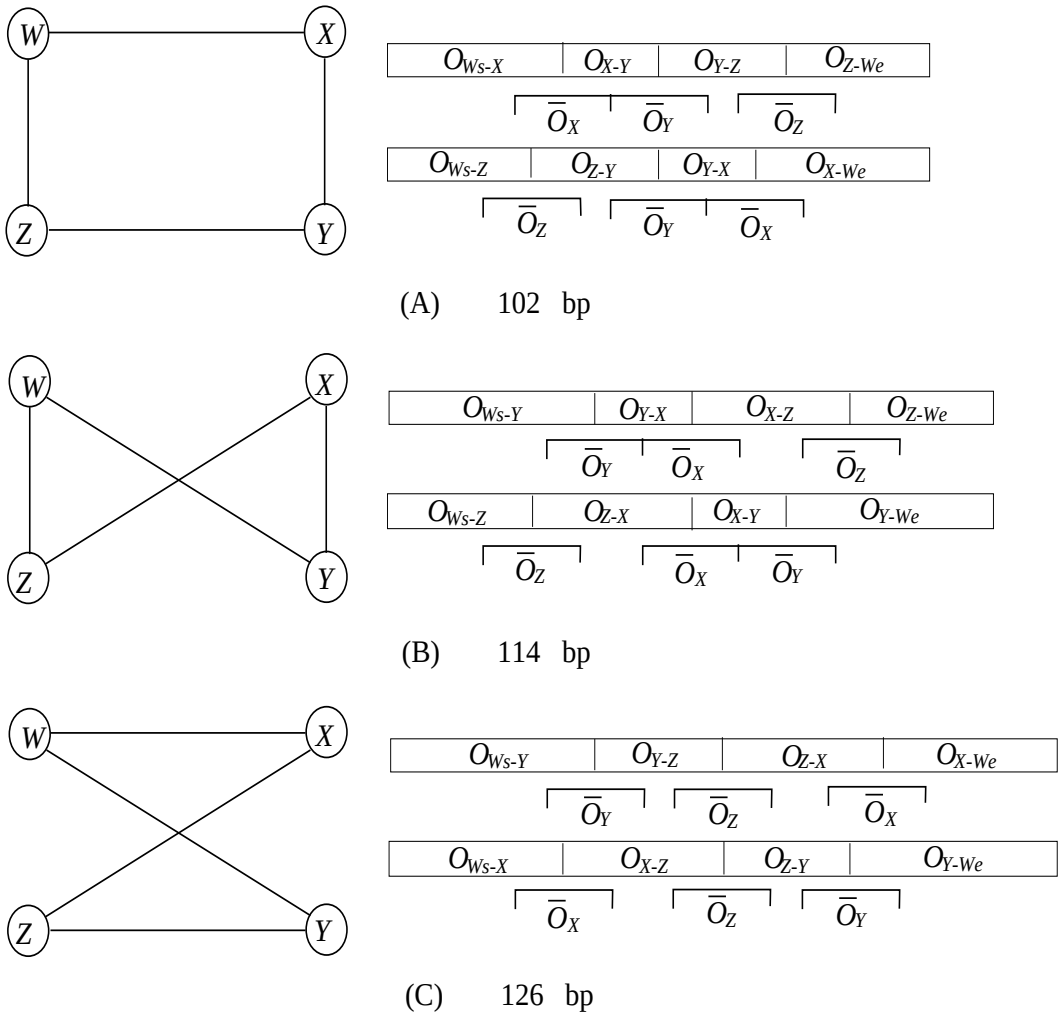

Figure 5: Three Hamiltonian cycles and their encoding DNA fragments. (A) Two DNA fragments with 102 bp encode the cycle $< W, X, Y, Z, W >$. (B) Two DNA fragments with 114 bp encode the cycle $< W, Y, X, Z, W >$. (C) Two DNA fragments with 126 bp encode the cycle $< W, Y, Z, X, W >$. Those two encoding the traveling-salesman cycle (the shortest in length) can be separated from others by size-dependent gel electrophoresis. The answer is revealed by multiple graduated PCRs.

The schematic design of algorithms introduced in this article is based directly on Adleman's model. All techniques involved have been proven and are considered feasible. However, two major differences occur in the problems presented here and the specific propositions are made. First of all, the starting and the ending vertices are the same in the cycle-searching problem. Dividing one of the vertices in the graph into two fictitious vertices (v_{start} and v_{end}) converts every cycle-searching problem into a path-searching problem, although this approach requires the use of extra oligonucleotides during data input. Moreover, the edges are undirected, and this distinction causes the desired answer to be encoded by two diverse DNA fragments concurrently. To unveil the answer, we introduced multiple graduated PCRs to decipher the cycle order. This procedure should also prove useful for solving other path searching problems, such as two Hamiltonian paths within a single graph.

The second innovation in our model is that the edges are weighted in the traveling-salesman problem. This weighting makes the length of desired DNA fragment unpredictable, and affinity purification must be executed prior to size-dependent gel isolation. Although the tripartite design of the oligonucleotides allows the encoding of weight information, the coupling efficiency during oligonucleotide synthesis may cause a potential problem during the synthesis of the the longer oligonucleotides. Assuming an average coupling efficiency of 99.5%, the overall yield would be 90.92% ($0.995^{19} = 0.9092$) for a 20-mer, but only 60.88% for a 100-mer and 36.88% for a 200-mer. Without proper post synthesis purification, the truncated products will certainly interfere with computational reactions. One possible effect is that shorter oligonucleotides will compete with full-length oligonucleotides during the annealing step, decreasing the parallel operation power. A possible solution for this concern may lie in the use of PCR procedures to generate the necessary oligonucleotides, rather than the use of a DNA synthesizer. Thus, a large molecule need be synthesized only once; thereafter, DNA fragments of the desired lengths can be obtained by amplification of various portions of a large DNA molecules. Single-stranded DNA isolation may be required for this alternative method.

The feasibility of using biological molecules for computation has been the subject of controversy and argument since its inception [6, 9]. Even though models have been proposed and DNA computers have been constructed, several fundamental issues remain to be resolved. First, biological reactions usually take minutes or hours to complete. Thus, although the ligation reaction and the polymerase chain reaction provide strong combination and screening power, the overall time consumed may ultimately be impractical (especially if an affinity purification step must be performed sequentially in these cases). Second, the feasibility and accuracy of DNA-based computation when the problem size is very large remains untested. Moreover, the DNA computation models addressed to date solve only very specific types of problems and do not address universal applications. However, recently Guarnieri et al. [7] reported the use of DNA molecules to achieve addition, a very welcome broadening of this new field of molecular computation.

The applications of biocomputation remain to be explored. The prospect of biocomputation should not be restricted to DNA molecules only. Other biological properties and biochemical reactions, such as the signal perception and transduction by a neural network, could perhaps be the basis for a more versatile and intelligent biocomputer. More studies are definitely required to reveal the full capacity of biocomputation, and such studies

should include both computer and biological sciences. Knowledge of how to use biomolecules to achieve computation may also provide insight into the sophisticated machinery of life. Moreover, because the genomes of several organisms have been fully sequenced [2, 4, 5], computation using cellular processes in an entire life form may be a possibility in the future.

Acknowledgements

We thank Po-Ting Wu and Gail Pieper for helpful discussion.

References

[1] L. M. Adleman. Molecular computation of solutions to combinatorial problems. *Science*, 266:1021–1024, 1994.

[2] C. J. Bult, O. White, C. J. Olsen, L. Zhou, R. D. Fleischmann, G. G. Sutton, J. A. Blake, L. M. FitzGerald, R. A. Clayton, J. D. Gocayne, A. R. Kerlavage, B. A. Dougherty, J.-F. Tomb, M. D. Adams, C. I. Reich, R. Overbeek, E. F. Kirkness, K. G. Weinstock, J. M. Glodek, J. L. Scott, N. S. M. Geoghagen, J. F. Weidman, J. L. Fuhrmann, D. Nguyen, T. R. Utterback, J. M. Kelley, J. D. Peterson, P. W. Sadow, M. C. Hanna, M. D. Cotton, K. M. Roberts, M. A. Hurst, B. P. Kaine, M. Borodovsky, H.-P. Klenk, C. M. Fraser, H. O. Smith, C. R. Woese, and J. C. Venter. Complete genome sequence of the methanogenic archaeon, *Methanococcus jannaschii*. *Science*, 273:1058–1073, 1996.

[3] T. H. Cormen, C. E Leiserson, and R. L. Rivest. *Introduction to Algorithms*. MIT Press, Cambridge, Mass., 1990.

[4] R. D. Fleischmann, M. D. Adams, R. White, R. A. Clayton, E. F. Kirkness, A. R. Kerlavage, C. J. Bult, J.-F. Tomb, B. A. Dougherty, J. M. Merrick, R. D. Fleischmann, M. D. Adams, R. White, R. A. Clayton, E. F. Kirkness, A. R. Kerlavage, C. J. Bult, J.-F. Tomb, B. A. Dougherty, J. M. Merrick, K. McKenney, G. Sutton, W. FitzHugh, C. Fields, J. D. Gocayne, J. Scott, R. Shirley, L.-I. Liu, A. Glodek, J. M. Kelley, J. F. Weidman, C. A. Phillips, T. Spriggs, E. Hedblom, M. D. Cotton, T. R. Utterback, M. C. Hanna, D. T. Nguyen, D. M. Saudek, R. C. Brandon, L. D. Fine, L. J. L. Fritchman, J. L. Fuhrmann, N. S. M. Groghagen, C. L. Gnehm, L. A. McDonald, K. V. Small, C. M. Fraser, H. O. Smith, and J. C. Venter. Whole-genome random sequencing and assembly of haemophilus influenzae rd. *Science*, 269:496–512, 1995.

[5] C. M. Fraser, J. D. Gocayne, J. D. White, M. D. Adams, R. A. Clayton, R. D. Fleischmann, C. J. Bult, A. R. Kerlavage, G. Sutton, J. M. Kelley, J. L. Fritchman, J. F. Weidman, K. V. Small, M. Sandusky, J. Fuhrmann, D. Nguyen, T. R. Utterback, D. M Saudek, C. A. Phillips, J. M Merrick, J.-F. Tomb, B. A. Dougherty, K. F. Bott, P.-C. Hu, T. S. Lucier, S. N. Peterson, H. O. Smith, C. A. Hutchison III, and J. C Venter. The minimal gene complement of mycoplasma genitalium. *Science*, 270:397–403, 1995.

[6] D. K. Gifford. On the path to computation with DNA. *Science*, 266:993–994, 1994.

[7] F. Guarnieri, M. Fliss, and C. Bancroft. Making DNA add. *Science*, 273:220–223, 1996.

[8] R. J. Lipton. DNA solution of hard computational problems. *Science*, 268:542–545, 1995.

[9] R. Pool. A boom in plans for DNA computing. *Science*, 268:498–499, 1995.