DNA Computing for Complex Scheduling Problem

Mohd Saufee Muhammad, Zuwairie Ibrahim, Satomi Ueda,
Osamu Ono, and Marzuki Khalid

Institute of Applied DNA Computing, Meiji University,
1-1-1 Higashi-Mita, Tama-Ku, Kawasaki-Shi, Kanagawa-Ken 214-8571, Japan
{msaufee, zuwairie, satomixx, ono}@isc.meiji.ac.jp,
marzuki@utmkl.utm.my
http://www.isc.meiji.ac.jp/~i3erabc/IADC.html

Abstract. Interest in DNA computing has increased overwhelmly since Adleman successfully demonstrated its capability to solve Hamiltonian Path Problem (HPP). Many research results of similar combinatorial problems which are mainly in the realm of computer science and mathematics have been presented. In this paper, implementation ideas and methods to solve an engineering related combinatorial problem using this DNA computing approach is presented. The objective is to find an optimal path for a complex elevator scheduling problem of an 8-storey building with 3 elevators. Each of the elevator traveled path is represented by DNA sequence of specific length that represent elevator's traveling time in a proportional way based on certain initial conditions such as present and destination floors, and hall calls for an elevator from a floor. The proposed ideas and methods show promising results that DNA computing approach can be well-suited for solving such real-world application in the near future.

1 Introduction

In 1994, Adleman [1] demonstrated the practical possibility of using molecules of Deoxyribonucleic Acid or DNA as a medium for computation. In his experiment, Adleman successfully solved a directed Hamiltonian Path Problem (HPP) using the tools of biomolecular engineering. Adleman [2] created DNA strands to represent an airplane flight from each of the seven cities, and then combined them to produce every possible route. Given its vast parallelism, the DNA strands yielded 10^9 answers in less than one second.

DNA computation relies on devising algorithms that solve problems using the encoded information in the sequence of oligonucleotides that make up DNA's double helix – the bases Adenine, Guanine, Thymine, and Cytosine (A, G, T, and C, respectively) and then breaking and making new bonds between them to reach the answer.

Research on DNA application to solve engineering problem however has not been very well establish. In this paper DNA computing technique to solve such problem is proposed. Since DNA computing is very suitable to solve combinatorial problems, an

L. Wang, K. Chen, and Y.S. Ong (Eds.): ICNC 2005, LNCS 3611, pp. 1182 – 1191, 2005.

elevator scheduling problem is chosen to be solved using this computing technique. The elevator scheduling problem involves finding an optimal path, or in other word, finding the shortest path for the travel path of the elevators for a building with certain number of elevators and floors. However, this problem is a complex combinatorial problem since certain criteria need to be fulfilled for the problem solution such as initial elevator position, its destinations and hall calls made for an elevator.

As mentioned, the elevator scheduling problem involves finding the elevator shortest travel path. Hence, current research works on DNA computing techniques for solving shortest path is being reviewed. Among others, Nayaranan and Zorbalas [3] proposed a constant proportional length-based DNA computing technique for solving Traveling Salesman Problem (TSP) or shortest path HPP. Yamamoto *et al.* [4] proposed a concentration-controlled DNA computing to accomplish local search for the shortest path problem. Lee *et al.* [5] proposed a DNA computing technique based on temperature gradient to solve the TSP problem. Ibrahim *et al.* [6] on the other hand proposed a direct-proportional length-based DNA computing for shortest path problem. The proposed method for the finding the optimal path of the elevator scheduling problem based on one of the shortest path method is presented in detail in this paper.

2 Biomolecular Operations of DNA

DNA computing involves biomolecular operations to manipulate the DNA strands by DNA synthesis, polymerase chain reaction (PCR), ligation, parallel overlap assembly (POA) and gel electrophoresis operations that are described as follows.

DNA Synthesis. DNA synthesis or replication is the process of copying a double-stranded DNA strand. Presently, a test tube containing approximately 10^{18} DNA molecules are available from commercial DNA synthesis companies at a reasonable price.

Polymerase Chain Reaction (PCR). PCR is an incredibly sensitive copying machine for DNA. DNA strands can be copied exponentially using PCR. PCR proceeds in cycles of 3 steps at different temperatures as illustrated in Fig. 1 [7]. These steps are denaturation (95°C), involves separation of the double strand template, annealing (55°C) where primers are 'annealed' to both the single strands ends and extension (75°C) process where polymerase enzymes are used to extend the primers into replicas of the template. This sequence is repeated causing an exponential growth in the number of templates.

Ligation. Ligation is often invoked after an annealing operation to concatenate strands of DNA. Although it is possible to use some ligase enzymes to concatenate free-floating double-stranded DNA, it is more efficient to allow single strands to anneal together, connecting up series of single-strand fragments, and then use a ligase to seal the covalent bonds between adjacent fragments, as shown in Fig. 2 [8].

Parallel Overlapping Assembly (POA). POA is a method for initial pool generation to solve weighted graph problems. This method is introduced by Stemmer [9] to facilitate *in vitro* mutagenesis. Kaplan *et al.* [10] successfully applied this method to

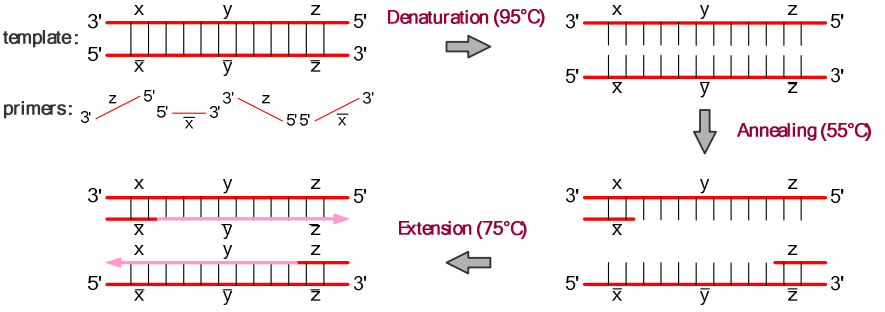

Fig. 1. One cycle of PCR

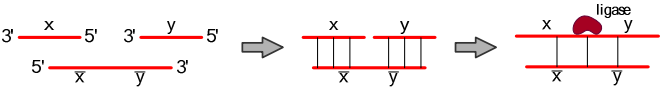

Fig. 2. Ligation process

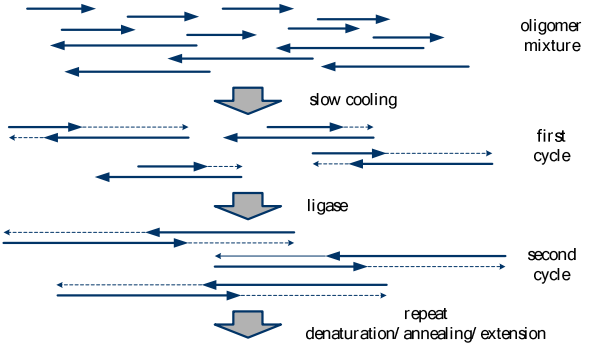

Fig. 3. Parallel overlapping assembly (POA) for initial pool generation. The continuous arrows represent the synthesized oligos which are the input to the computation. The dotted arrows represent the elongated part during polymerization. The arrowhead indicates the 3' end.

generate initial pool consisting of binary numbers to solve maximal clique problem.POA involves thermal cycle where during the thermal cycle, the position strings in one of the oligo is annealed to the complementary strings of the next oligo. In the presence of polymerase enzyme, the oligo 3' end side is extended to form a longer double stranded DNA as depicted in Fig. 3 [11]. A data pool consisting of all possible combinations are thus produced after a number of thermal cycles.

Gel Electrophoresis. Gel electrophoresis is a technique for separating DNA strands according to its length through a gel in an electrical field based on the fact that DNA is negatively charged [12]. As the separation process continues the separation between the larger and smaller fragments increases as depicted in Fig. 4 [13, 14].

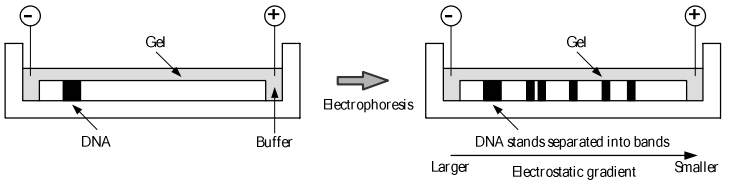

Fig. 4. Gel electrophoresis process

3 Elevator Scheduling Problem

Typically, a building consists of N floors with a total of M elevators. An example of elevator situation at an instance of a time can be illustrated as in Table 1.

The elevator travel path can be represented as a weighted graph problem. This is done by representing the elevator position at floor 1, 2, 3, … , $N-2$, $N-1$, N with nodes V_1, V_2, V_3, … , V_{N-2}, V_{N-1}, V_N respectively. The graph of all possible travel paths of one of the elevator is constructed as shown in Fig. 5.

Table 1. Elevator situation at an instance of time

Floor No	Elevator 1	Elevator 2	…	Elevator M–1	Elevator M	Hall Call
N			…	(N–3, 7, 3)		
N–1	(N–2, 4, 1)		…			↑
N–2			…			↓
:	:	:	:	:	:	:
3		(4, 6, N–2)	…			↑
2			…		(5, 8, N–1)	↓
1			…			

The weight between nodes can be represented as

$$\omega_{|j-i|} = (|\, j-i\,|)T_T + T_S \tag{1}$$

where

i – elevator present floor position
j – elevator destination floor position
$|j-i|$ – total number of floors of elevator movement
T_T – elevator traveling time between two consecutive floors
T_S – elevator stopping time at a floor

The output of the graph, given by sum of the graph weights thus represents the total traveling time of the elevator, i.e.

$$G(E) = \sum_{|j-i|=1}^{N-1} \omega_{|j-i|} \tag{2}$$

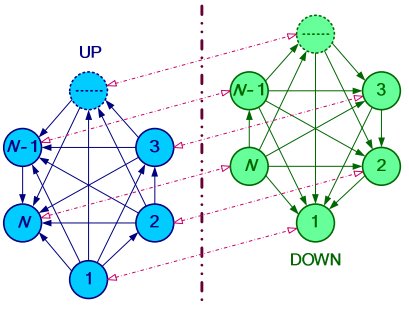

Fig. 5. Graph of all possible travel paths of an elevator

For a building with M elevators, M similar graphs as shown in Fig. 5 can be duplicated representing all M elevators travel paths. The total traveling time of all the elevators can thus be calculated by summing up each of the elevators traveling time as

$$G(E_1, E_2, \cdots, E_{M-1}, E_M) = G(E_1) + G(E_2) + \cdots + G(E_{M-1}) + G(E_M) \tag{3}$$

The optimal travel path is thus given by the minimum total traveling time of all the elevators with all initial conditions and requirements satisfied, i.e.

$$\text{Optimal Travel Path} = G(E_1, E_2, \ldots, E_{M-1}, E_M)_{min} \tag{4}$$

Let us consider a building with 3 elevators and 8 floors. Elevator A is presently at 1st floor and its destination is 4th and 5th floor, elevator B is presently at 6th floor and its destination is 3rd and 2nd floor, and elevator C is presently at 3rd floor and its destination is 6th and 8th floor. There are hall calls at 4th floor going up, and hall calls at 5th floor going down, as illustrated in Table 2.

Table 2. Elevator position for elevator scheduling problem example

Floor No	Elevator A	Elevator B	Elevator C	Hall Call
8				
7				
6		(3, 2)		
5				↓
4				↑
3			(6, 8)	
2				
1	(4, 5)			

The solution to this elevator scheduling problem is to find the optimal travel path for all the elevators that fulfill all initial conditions and requirements defined. Therefore, it is necessary to calculate the total output of the graphs $G(A, B, C)$. The optimal travel path will thus be given by the minimum graph output among all the graph output for all possible travel paths of elevator A, B and C.

4 DNA Computing to Solve Elevator Scheduling Problem

A method proposed by [6] to solve the shortest path problem is being applied to solve the elevator scheduling problem. Using this method, the weights between every node are encoded by oligonucleotide length in a proportional way to represent the elevator's traveling time between floors. A number of steps are performed for the computation process that is discussed below.

Step 1. The elevator position are represented as nodes V_1, V_2, V_3, V_4, V_5, V_6, V_7, V_8 and V_1', V_2', V_3', V_4', V_5', V_6', V_7', V_8' for upward and downward movements respectively representing all the 8 floor positions in the building.

Step 2. The weights between nodes are assigned in such a way that it will directly represent the elevator's traveling time between the floors. Since the building consists of 8 floors, the maximum number of floors that the elevator can travel is $(8 - 1) = 7$ floors. Now, assuming that $T_T = 5$ sec, $T_S = 15$ sec, and representing every 5 sec with 10 units, we have form (1)

$$\omega_0 = 0(5) + 15 = 15 \text{ sec} = 30 \quad , \quad \omega_1 = 1(5) + 15 = 20 \text{ sec} = 40$$
$$\omega_2 = 2(5) + 15 = 25 \text{ sec} = 50 \quad , \quad \omega_3 = 3(5) + 15 = 30 \text{ sec} = 60$$
$$\omega_4 = 4(5) + 15 = 35 \text{ sec} = 70 \quad , \quad \omega_5 = 5(5) + 15 = 40 \text{ sec} = 80$$
$$\omega_6 = 6(5) + 15 = 45 \text{ sec} = 90 \quad , \quad \omega_7 = 7(5) + 15 = 50 \text{ sec} = 100$$

Step 3. Construct a graph with its corresponding weight representing all possible travel path combinations of each elevator that fulfill all the required initial conditions and requirements as shown in Fig. 6. Note that all possible end paths of elevator *A* are joined with the start path of elevator *B*. Similarly, all possible end paths of elevator *B* are joined with the start path of elevator *C*. This is done in order that the total output of the graph *G* (*A*, *B*, *C*) representing the travel path of all the elevators can be calculated.

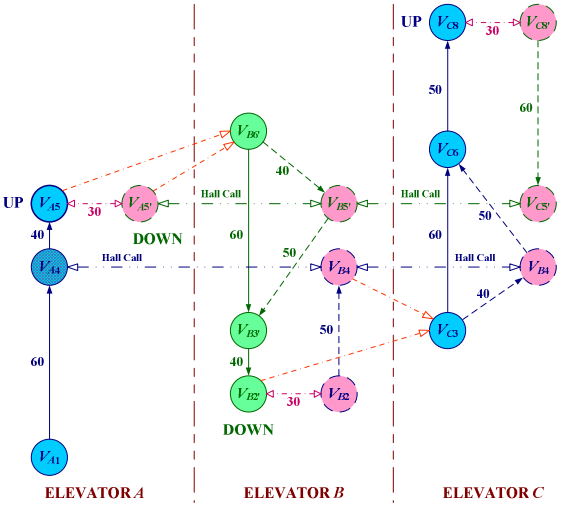

Fig. 6. Graph of all possible travel path combinations of elevators *A*, *B* and *C*

Table 3. DNA sequence for nodes (elevator floor position)

Upward Movement	20-mer Sequence (5'–3')		GC%	T_m (°C)
	V_{ia}	V_{ib}		
V_1	TCATCCTCCC	GTCATTAACT	0.45	59.35
V_2	TTGGCTAAGG	AAGTCGGTAG	0.50	59.32
V_3	GCTCTAAGCT	AGTATCGCGG	0.55	59.24
V_4	CAATACTGCG	CGAATGTTAC	0.45	59.20
V_5	AAATACCAAA	AACATGCCGT	0.35	59.19
V_6	ATAGGGGGGA	CATATCCAAT	0.45	59.19
V_7	CTAATTCTGC	AAACCACACG	0.45	59.18
V_8	AATTTGGGTG	GACCGTAGTA	0.45	59.16
Downward Movement	20-mer Sequence (5'–3')		GC%	T_m (°C)
	V_{ia}	V_{ib}		
$V_{8'}$	ACGGAGTCAA	GTGAATAGCC	0.50	59.15
$V_{7'}$	GGGCTTGATT	GTTCTGAGTT	0.45	59.13
$V_{6'}$	CACATAGACT	GGGGGTTACC	0.55	59.12
$V_{5'}$	GAAGGGGCTC	AAAGTCATAA	0.45	59.11
$V_{4'}$	AACTCGCCTA	GAACTGCCTA	0.50	59.09
$V_{3'}$	CAATATGCTT	TCCGGCTTAT	0.40	59.05
$V_{2'}$	ATCCCAATTA	TGGGTCTCAA	0.40	59.04
$V_{1'}$	CTACTCCCCA	CTCCACAGTT	0.55	59.01

Step 4. Assign a unique DNA sequence for each of the node (elevator floor position) and its direction. Using available software for DNA sequence design named DNASequenceGenerator [15], the sequence is generated as shown in Table 3. The GC contents (GC%) and melting temperature (T_m) of each sequence is also shown in the table. Note that V_i is separated into half-5 end V_{ia} and half-3 end V_{ib}.

Step 5. Synthesize the oligos for every path in the graph according to the following rules [6] so that the oligos length will directly represent the weight between the nodes

(i) If i is a start node and j is an intermediate node, synthesize the oligo as
$$V_{iab}(20) + W_{ij}(\omega_{ij} - 30) + V_{ja}(20)$$
(ii) If i is an intermediate node and j is an end node, synthesize the oligo as
$$V_{ib}(20) + W_{ij}(\omega_{ij} - 30) + V_{jab}(20)$$
(iii) If i and j are both intermediate nodes, synthesize the oligo as
$$V_{ib}(20) + W_{ij}(\omega_{ij} - 20) + V_{ja}(20)$$

where V denotes the DNA sequence for node, W denotes the DNA sequence for weight, ω denotes the weight value, and '+' denotes a 'join' between the DNA sequence. All the synthesized oligos based on the stated rules are shown in Table 4.

Table 4. DNA sequence for path between nodes

Node Path	DNA Sequence (5' – 3')		
	V_i	W_{ij}	V_j
$V_1 \rightarrow V_4$	TCATCCTCCC GTCATTAACT	30	CAATACTGCG
$V_2 \rightarrow V_4$	AAGTCGGTAG	20	CAATACTGCG CGAATGTTAC
$V_3 \rightarrow V_4$	GCTCTAAGCT AGTATCGCGG	10	CAATACTGCG
$V_3 \rightarrow V_6$	GCTCTAAGCT AGTATCGCGG	30	ATAGGGGGGA
$V_4 \rightarrow V_5$	CGAATGTTAC	10	AAATACCAAA AACATGCCGT
$V_4 \rightarrow V_5$	CGAATGTTAC	20	AAATACCAAA
$V_5 \rightarrow V_{5'}$	AACATGCCGT	0	GAAGGGGCTC AAAGTCATAA
$V_6 \rightarrow V_8$	CATATCCAAT	20	AATTTGGGTG GACCGTAGTA
$V_6 \rightarrow V_8$	CATATCCAAT	30	AATTTGGGTG
$V_8 \rightarrow V_{8'}$	GACCGTAGTA	10	ACGGAGTCAA
$V_8 \rightarrow V_{5'}$	GTGAATAGCC	30	GAAGGGGCTC AAAGTCATAA
$V_6 \rightarrow V_{5'}$	CACATAGACT GGGGGTTACC	10	GAAGGGGCTC
$V_6 \rightarrow V_{3'}$	CACATAGACT GGGGGTTACC	30	CAATATGCTT
$V_5 \rightarrow V_{3'}$	AAAGTCATAA	30	CAATATGCTT
$V_3 \rightarrow V_{2'}$	TCCGGCTTAT	10	ATCCCAATTA TGGGTCTCAA
$V_3 \rightarrow V_{2'}$	TCCGGCTTAT	20	ATCCCAATTA
$V_2 \rightarrow V_2$	TGGGTCTCAA	10	TTGGCTAAGG

Step 6. All the synthesized oligos are then poured into a test tube for initial pool generation. POA is used to for the initial pool generation as suggested by Lee *et al.* [11] who demonstrated that POA is a more efficient and economical initial pool generation method for weighted graph problems. POA operation is similar to PCR; the only difference is that POA operates without the use of primers. As PCR, one cycle consists of three steps: hybridization, extension, and denaturation. During the annealing step, the temperature is decreased slowly so that partial hybridization is allowed to occur at respective locations. The extension on the other hand is applied with the presence of polymerase enzyme and the polymerization can be done from 5' to 3' direction. The generated double stranded DNA molecules are then separated by denaturation step. This can be done by increasing the temperature until the double stranded DNA molecules are separated to become single stranded DNA molecules.

From the graph of Fig. 6, the total output of the graph representing the travel path of each elevator A, B and C with either elevator A, B and C answering the hall calls can be calculated. The calculations performed verifies that the optimal path is $V_{A1} \rightarrow V_{A4} \rightarrow V_{A5} \rightarrow V_{A5'}$ for elevator A, $V_{B6'} \rightarrow V_{B3'} \rightarrow V_{B2'}$ for elevator B and $V_{C3} \rightarrow V_{C6} \rightarrow V_{C8}$ for elevator C with a total output $G(A, B, C) = 340$.

Fig. 7 illustrates the oligos involved in the generation of this optimal path. Note that at the same time, all other combinations of travel paths are also generated in the same manner.

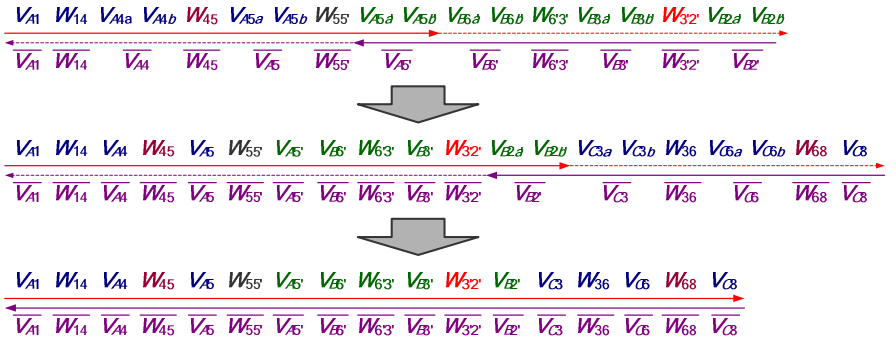

Fig. 7. DNA duplex based on POA method representing elevator's optimal path $V_{A1} \rightarrow V_{A4} \rightarrow V_{A5} \rightarrow V_{A5'} \rightarrow V_{B6'} \rightarrow V_{B3'} \rightarrow V_{B2'} \rightarrow V_{C3} \rightarrow V_{C6} \rightarrow V_{C8}$. The 3' end is indicated by the arrowhead.

Step 7. At this stage, an initial pool of solution is produced. The optimal path combinations among many other alternative path combinations of the problem have to be filtered. This filtering process copies the target DNA duplex exponentially using the PCR process by amplifying all the DNA molecules containing start node V_{A1} and end node V_{C8}. Numerous amount of DNA strands representing the start node V_{A1} and end node V_{C8} passing through all possible travel path combinations will be presented once the PCR operation is accomplished. Finally, gel electrophoresis is then performed onto the output solution of the PCR. The DNA molecules will be separated according to its length during this operation. The bands of gel electrophoresis are then analyzed, and the DNA duplex representing the shortest path starting from V_{A1} and end node V_{C8} will be extracted to represent the required solution of the problem.

5 Conclusions

In this paper, ideas and implementation methods to solve an elevator scheduling problem using DNA computing has been presented and discussed in details. DNA computing application towards solving this type of engineering problem has been shown to be achievable and applicable. It is expected from experimental results that the shortest DNA sequence length will represent the required optimal path for the elevator scheduling problem. With the successful confirmation of the expected result, the applicability of DNA computing could be extended into many more complex problems of this type of nature. Hence, the applicability of DNA computing could be extended into greater fields of other engineering related problems.

References

1. Adleman, L.M.: Molecular Computation of Solutions to Combinatorial Problems. Science, Vol. 266 (1994) 1021-1024
2. Adleman, L.M.: Computing with DNA. Scientific American (1998) 34-41
3. Narayanan, A., Zorbalas, S.: DNA Algorithms for Computing Shortest Paths. Proceedings of Genetic Programming, (1998) 718-723

4. Yamamoto, Y., Kameda, A., Matsuura, N., Shiba, T., Kawazoe, Y., Ahochi, A.: Local Search by Concentration-Controlled DNA Computing. International Journal of Computational Intelligence and Applications, Vol. 2 (2002) 447-455
5. Lee, J.Y., Shin, S.Y., Augh, S.J., Park, T.H., Zhang, B.T.: Temperature Gradient-Based DNA Computing for Graph Problems with Weighted Edges. Lecture Notes in Computer Science, Springer-Verlag, Vol. 2568 (2003) 73-84
6. Ibrahim, Z., Tsuboi, Y., Ono, O., Khalid, M.: Direct-Proportional Length-Based DNA Computing for Shortest Path Problem. International Journal of Computer Science and Applications, Vol. 1, Issue 1 (2004) 46-60
7. Fitch, J. P.: Engineering Introduction to Biotechnology. SPIE Press (2001)
8. Zucca, M.: DNA Based Computational Models. PhD Thesis, Politecnico Di Torino, Italy (2000)
9. Stemmer, W.P.: DNA Shuffling by Random Fragmentation and Reassembly: In Vitro Recombination for Molecular Evolution. Proc. Natl. Acad. Sci. U.S.A., Vol. 91 (1994) 10747-10751
10. Kaplan, P.D., Ouyang, Q., Thaler, D.S., Libchaber, A.: Parallel Overlap Assembly for the Construction of Computational DNA Libraries. Journal of Theoretical Biology, Vol. 188, Issue 3 (1997) 333-341
11. Lee, J.Y., Lim, H.W., Yoo, S.I., Zhang, B.T., Park, T.H.: Efficient Initial Pool Generation for Weighted Graph Problems Using Parallel Overlap Assembly. Preliminary Proceeding of the 10th International Meeting on DNA Computing (2004) 357-364
12. Paun, G., Rozenberg, G., Salomaa, A.,: DNA Computing: New Computing Paradigms. Lecture Notes in Computer Science, Springer-Verlag, Vol. 1644 (1998) 106-118
13. Amos, M.: DNA Computation. PhD Thesis, The University of Warwick, UK (1997)
14. Yamamoto, Y., Kameda, A., Matsuura, N., Shiba, T., Kawazoe, Y., Ahochi, A.: A Separation Method for DNA Computing Based on Concentration Control. New Generation Computing, Vol. 20, No. 3 (2002) 251-262
15. Udo, F., Sam, S., Wolfgang, B., Hilmar, R.: DNASequenceGenerator: A Program for the Construction of DNA Sequences. Proceedings of the Seventh International Workshop on DNA Based Computers (2001) 23-32