A Design and Implementation Method for Elevator Scheduling Problem Using DNA Computing Approach

Mohd Saufee Muhammad, and Osamu Ono

Meiji University, Kanagawa, JAPAN 214-8571

Summary
We present a design and implementation method to solve an elevator scheduling problem using DNA computing in this research. DNA sequences of length directly proportional to the elevator's traveling time are encoded to represent all possible travel path combinations based on certain initial conditions such as present and destination floors, and hall calls from a floor. Parallel overlap assembly is employed for initial pool generation and polymerase chain reaction for amplification. Gel electrophoresis is then performed to separate the sequences according to its length and its image is captured to visualize the optimal path. Experimental result obtained verifies that this approach can be well-suited to solve such real-world problem of this nature.

Key words:
Elevator scheduling problem, DNA computing, gel electrophoresis, optimal path.

Introduction

The practical possibility of using molecules of Deoxyribonucleic Acid or DNA as a medium for computation was first demonstrated in 1994 by Leonard M. Adleman [1]. Using the tools of biomolecular engineering, Adleman successfully solved a directed Hamiltonian Path Problem (HPP) in his experiment.

Instead of the traditional silicon-based computing technologies, DNA computing is a form of computing that uses DNA and molecular biology. Adleman's pioneering work set the new approach for this new field of bio-computing research. Computing with DNA generated a tremendous amount of excitement by offering a brand new paradigm for performing and viewing computations. Adleman's experiment [2] solved a simple instance of the Traveling Salesman Problem (TSP) by manipulating the DNA molecules. This marked the first solution of a mathematical problem with the tools of biology.

Computing with DNA offers many advantages over traditional silicon-based computing due to several reasons. These include massive parallelism and memory capacity. The primary advantage offered by most proposed models of DNA based computation is the ability to handle millions of operation in parallel. DNA computing can reach approximately 10^{20} operations per second compared to today's teraflop supercomputers. Certain operations in DNA computing (for example, hybridization – the bonding of two DNA strands to form the double helix) are over a billion times more energy efficient as compared to conventional computers. Also, DNA stores information at a density of about one bit per nm^3 – about a trillion times as efficiently as videotape.

DNA computation relies on devising algorithms to solve problems using the encoded information in the sequence of nucleotides that make up DNA's double helix strand, breaking and making new bonds between them to reach the answer. Each strand may be viewed as a chain of nucleotides, or bases. An *n*-letter sequence of consecutive bases is known as an *n*-mer or an oligonucleotide of length *n*. The four DNA nucleotides are adenine (A), guanine (G), cytosine (C) and thymine (T). Each strand has, according to chemical convention, a 5' and a 3' end, thus any single strand has a natural orientation. The classical double helix of DNA is formed when two separate strands bond together. Bonding occurs by the pairwise attraction of bases; A bonds with T and G bonds with C. The pairs (A, T) and (G, C) are known as Watson-Crick complementary base pairs [3].

Research on DNA computing approach to solve engineering related problems however has not been very well established. Since DNA computing is very suitable to solve combinatorial problems, an elevator scheduling problem is chosen as a benchmark to be solved using this computing technique. The elevator scheduling problem involves finding an optimal path, or in other words, finding the shortest elevator travel path of a building with certain number of elevators and floors. However, this 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.

There are several research reports on DNA computing techniques for solving shortest path problems of a weighted graph. Nayaranan and Zorbalas [4] proposed a constant proportional length-based DNA computing technique for TSP. Yamamoto *et al*. [5] proposed a concentration-controlled DNA computing to accomplish local search for solving shortest path problem. Lee *et al*. [6] proposed a DNA computing technique based on

Manuscript received June 5, 2006.
Manuscript revised June 25 , 2006.

temperature gradient to solve the TSP problem. Ibrahim *et al*. [7] on the other hand proposed a direct-proportional length-based DNA computing for shortest path problem.

All the methods proposed for computing weighted graph have not been well applied to solve a real word problem. Previously, ideas and implementation methods for solving elevator scheduling problem using DNA computing method had been proposed [8, 9]. In this paper, the DNA computing techniques for solving shortest path problems of a weighted graph have been utilized to solve the elevator scheduling problem. Here, the elevator's traveled paths and traveling time are represented by DNA sequences of specific length designed based on certain initial conditions such as elevator's present and destination floors, and hall calls for an elevator from a floor. Constraints such as node position in the graph, initial pool generation method, sequence amplification and computing output representation are investigated and discussed in detail. Results of an *in vitro* experiment to realize the computing output are presented showing the feasibility and applicability of the computing technique proposed.

2. Elevator Scheduling Problem Overview

Let us consider a typical situation of a building with M elevators and N floors. The present elevator positions, its destinations and hall calls from a floor at a particular instance can be illustrated as shown in Table I.

Table 1: Elevator Situation at a Particular Instance

Floor No	Elevator 1	Elevator 2	…	Elevator $M-1$	Elevator M	Hall Call
N						
N–1		7, 4				↑
N–2				N–4, 5, 2		↓
:	:	:	:	:	:	
3					5, 8, 9	↓
2						↑
1	3, 5, N–2					

These elevator travel paths can be represented using a weighted graph by representing each elevator position at floors 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 weight between each node representing the elevator's travel time between each floor can be formulated as

$$\omega_{|j-i|} = (|j-i|)T_C + T_S \quad \textbf{(1)}$$

where

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

The weighted graph of all possible travel path combinations of one of the elevator can be constructed as shown in Fig. 1.

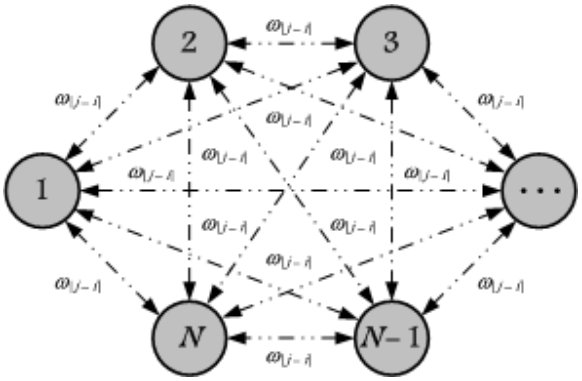

Fig. 1 Weighted graph of all possible travel path combinations of an elevator.

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

$$G(E_x) = \sum \omega_{|j-i|} \quad \textbf{(2)}$$

For a building with M elevators, M similar graphs as shown in Fig. 1 can be duplicated representing all M elevators travel paths. The total traveling time of all the elevators can now be calculated by summing up each of the elevator's traveling time, i.e.

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

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

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

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

Table 2: Elevator Scheduling Problem Example

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

Since the building is 6 floors high, the maximum number of floors that the elevator can travel is $(6-1) = 5$ floors. Now, assume that T_C = 5 s, T_S = 15 s, and representing 5 s of time with 10 units we have from (1)

$$\omega_1 = 1\,(5) + 15 = 20\text{ s} = 40$$
$$\omega_2 = 2\,(5) + 15 = 25\text{ s} = 50$$
$$\omega_3 = 3\,(5) + 15 = 30\text{ s} = 60$$
$$\omega_4 = 4\,(5) + 15 = 35\text{ s} = 70$$
$$\omega_5 = 5\,(5) + 15 = 40\text{ s} = 80$$

A weighted graph representing all possible travel path combinations of elevators *A* and *B* with either elevator answering one or both of the hall calls can now be constructed as shown in Fig. 2. Note that all possible end paths of elevator *A* are joined with the start paths of elevator *B*. This is done in order that the total output of the graph *G* (*A*, *B*) representing the travel path combinations of the elevators can be calculated.

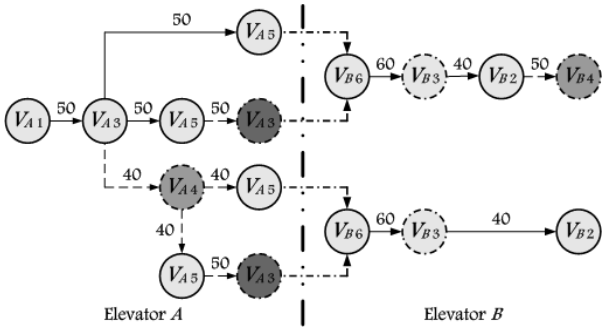

Fig. 2 Weighted graph of all possible travel path combinations.

Since there are two hall calls with two available elevators, it is clearly seen that there are $2^2 = 4$ possible travel path combinations for both of the elevators as tabulated in Table 3. The required solution for the elevator scheduling problem is thus the optimal path weight $G(A, B)_3 = 230 = 115$ s.

Table 3: Total Graph Output Of All Travel Path Combinations

Elevator No	Hall Calls	Elevator Travel Paths	Total Graph Output
A	–	$V_{A1} \rightarrow V_{A3} \rightarrow V_{A5}$	$G(A, B)_1 = 100 + 150$
B	3 , 4	$V_{B6} \rightarrow V_{B3} \rightarrow V_{B2} \rightarrow V_{B4}$	= 250
A	3	$V_{A1} \rightarrow V_{A3} \rightarrow V_{A5} \rightarrow V_{A3} \rightarrow$	$G(A, B)_2 = 150 + 150$
B	4	$V_{B6} \rightarrow V_{B3} \rightarrow V_{B2} \rightarrow V_{B4}$	= 300
A	4	$V_{A1} \rightarrow V_{A3} \rightarrow V_{A4} \rightarrow V_{A5} \rightarrow$	$G(A, B)_3 = 130 + 100$
B	3	$V_{B6} \rightarrow V_{B3} \rightarrow V_{B2}$	= 230
A	3 , 4	$V_{A1} \rightarrow V_{A3} \rightarrow V_{A4} \rightarrow V_{A5}$	$G(A, B)_4 = 180 + 100$
B	–	$\rightarrow V_{A3} \rightarrow V_{B6} \rightarrow V_{B3} \rightarrow V_{B2}$	= 280

3. DNA Computing Approach Design Solution

In order to solve this type of elevator scheduling problem using DNA computing approach, several computing steps are performed that are discussed below.

Step 1. Reconstruct the weighted graph shown in Fig. 2 in order to distinguish between start, intermediate and end nodes, and also to differentiate the nodes of different travel path combinations as depicted in Fig. 3.

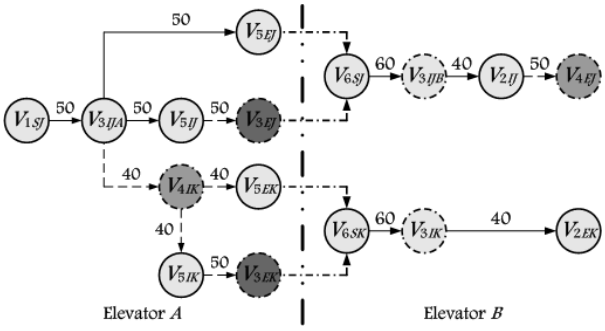

Fig. 3 Weighted graph for DNA computing approach solution showing different node locations and paths.

Note that *S*, *I* and *E* denote start, intermediate and end nodes respectively, while *J* and *K* denotes the different travel paths combinations. This is an important design step since every different node location and path in the graph will be represented with different DNA sequences in order to obtain all the possible travel path combinations that fulfill all the initial conditions and requirements stated.

Step 2. Generate a unique DNA sequence for every node of the graph where each start, intermediate and end node of different travel path combination is assigned with a specific DNA sequence. Using this rule, every DNA sequence assigned to each node will therefore identify its location and travel path in the graph. The sequences are generated using available software for DNA sequence design named DNASequenceGenerator [10], and is shown in Table 4. The GC contents (GC%), melting temperature

(T_m) are also shown in the table, and the sequence complements are shown in Table 5.

Table 4: DNA Sequences for Nodes

Node V_i	20-mer sequences (5' – 3')	GC %	T_m (°C)
V_{1SJ}	CGGCGGTCCACTAAATACTA	50	60.0
V_{3IJ}	CACTCTTTGTGAACGCCTTC	50	60.8
V_{5IJ}	TGAACCGGCCCTTTATATCT	45	60.7
V_{3EJ}	TCATTCGAGTTATTCCTGGG	45	59.9
V_{5EJ}	CTATAAGGCCAAAGCAGTCG	50	59.9
V_{6SJ}	GGACCTGCATCATACCAGTT	50	59.8
V_{2IJ}	AAAGCCCGTCGGTTAAGTTA	45	60.8
V_{4EJ}	GGAATCCATTGATCGCTTTA	40	59.9
V_{4IK}	GTGGGTTAGAGGTAGTCCGG	60	60.8
V_{5IK}	CCGCTGATCCTTGCTAAGTA	50	60.4
V_{3EK}	AAATGACCTTTTTAACGGCA	35	59.4
V_{5EK}	ATGCCTGGCTAAAGTGAGAC	50	59.3
V_{6SK}	TGCACGCAAAACTATTTCAT	35	59.2
V_{3IK}	TCTGCACTGTTAATGAGCCA	45	60.4
V_{2EK}	CTAATTTTAGAAATGGCGCG	40	59.7

Table 5: Complement of DNA Sequences For Nodes

Node $\overline{V_i}$	20-mer sequences (5' – 3')
$\overline{V_{1SJ}}$	TAGTATTTAGTGGACCGCCG
$\overline{V_{3IJ}}$	GAAGGCGTTCACAAAGAGTG
$\overline{V_{5IJ}}$	AGATATAAAGGGCCGGTTCA
$\overline{V_{3EJ}}$	CCCAGGAATAACTCGAATGA
$\overline{V_{5EJ}}$	CGACTGCTTTGGCCTTATAG
$\overline{V_{6SJ}}$	AACTGGTATGATGCAGGTCC
$\overline{V_{2IJ}}$	TAACTTAACCGACGGGCTTT
$\overline{V_{4EJ}}$	TAAAGCGATCAATGGATTCC
$\overline{V_{4IK}}$	CCGGACTACCTCTAACCCAC
$\overline{V_{5IK}}$	TACTTAGCAAGGATCAGCGG
$\overline{V_{3EK}}$	TGCCGTTAAAAAGGTCATTT
$\overline{V_{5EK}}$	GTCTCACTTTAGCCAGGCAT
$\overline{V_{6SK}}$	ATGAAATAGTTTTGCGTGCA
$\overline{V_{3IK}}$	TGGCTCATTAACAGTGCAGA
$\overline{V_{2EK}}$	CGCGCCATTTCTAAAATTAG

Step 3. Synthesize the DNA sequence for every node path of the graph according to the following rules so that the sequence length will directly represent the weight between the nodes:

(i) If i is a start node and j is an intermediate node, synthesize the sequence as

$$V_i(20) + W_{ij}(\omega_{ij} - 30) + V_j(20)$$

(ii) If i is an intermediate node and j is an end node, synthesize the sequence as

$$V_i(20) + W_{ij}(\omega_{ij} - 30) + V_j(20)$$

(iii) If i and j are both intermediate nodes, synthesize the sequence as

$$V_i(20) + W_{ij}(\omega_{ij} - 20) + V_j(20)$$

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 sequences based on the stated rules are shown in Table 6 where capital letters denote the nodes and small letters denote the weight between nodes.

Table 6: Synthesized DNA Sequences for Node Paths

Node V_i	20-mer sequences (5' – 3')
$V_{1SJ} \rightarrow V_{3IJ}$	CGGCGGTCCACTAAATACTAaggtcgtttaaggaagta cgCACTCTTTGTGAACGCCTTC
$V_{3IJ} \rightarrow V_{4IK}$	CACTCTTTGTGAACGCCTTCacgtcgtgtaacgaagtcc tGTGGGTTAGAGGTAGTCCGG
$V_{3IJ} \rightarrow V_{5IJ}$	CACTCTTTGTGAACGCCTTCccgtcggttaagcaagtaa tgtactatgctTGAACCGGCCCTTTATATCT
$V_{3IJ} \rightarrow V_{5EJ}$	CACTCTTTGTGAACGCCTTCgcgtcgcttaccgaagca cgCTATAAGGCCAAAGCAGTCG
$V_{4IK} \rightarrow V_{5IK}$	GTGGGTTAGAGGTAGTCCGGcgctcgttgaagccagt accCCGCTGATCCTTGCTAAGTA
$V_{4IK} \rightarrow V_{5EK}$	GTGGGTTAGAGGTAGTCCGGgcgtcttttaATGCC TGGCTAAAGTGAGAC
$V_{5SJ} \rightarrow V_{3EJ}$	TGAACCGGCCCTTTATATCTacgtgttttacccaagtca gTCATTCGAGTTATTCCTGGG
$V_{5IK} \rightarrow V_{3EK}$	CCGCTGATCCTTGCTAAGTAgcggcgtgtcacgaacta cgAAATGACCTTTTTAACGGCA
$V_{3EJ} \rightarrow V_{6SJ}$	TCATTCGAGTTATTCCTGGGGGACCTGCATC ATACCAGTT
$V_{5EJ} \rightarrow V_{6SJ}$	CTATAAGGCCAAAGCAGTCGGGACCTGCATC ATACCAGTT
$V_{3EK} \rightarrow V_{6SK}$	AAATGACCTTTTTAACGGCATGCACGCAAAA CTATTTCAT
$V_{5EK} \rightarrow V_{6SK}$	ATGCCTGGCTAAAGTGAGACTGCACGCAAA ACTATTTCAT
$V_{6SJ} \rightarrow V_{3IJ}$	GGACCTGCATCATACCAGTTacgtggtttaaggaagta cggtactatgctCACTCTTTGTGAACGCCTTC
$V_{6SK} \rightarrow V_{3IK}$	TGCACGCAAAACTATTTCATccgtgggttaaagaagtc ctgtactctcctTCTGCACTGTTAATGAGCCA
$V_{2IJ} \rightarrow V_{4EJ}$	AAAGCCCGTCGGTTAAGTTAggtcttttaatcaactaat gGGAATCCATTGATCGCTTTA
$V_{3IJ} \rightarrow V_{2IJ}$	CACTCTTTGTGAACGCCTTCacgtcgctgcaagaacta cgAAAGCCCGTCGGTTAAGTTA
$V_{3IK} \rightarrow V_{2EK}$	TCTGCACTGTTAATGAGCCAacgtcttgtcCTACG GATAGGTGTCTGGGA

Step 4. Combine all the synthesized DNA sequences in a test tube for initial pool generation. The initial pool generation uses parallel overlap assembly (POA) method [11] as suggested by Lee *et al*. [12] who demonstrated that POA is a more efficient and economical method for weighted graph problems. Basically, POA operation 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 during denaturation step in which the temperature is increased until the double stranded DNA molecules are separated to become single stranded DNA molecules. An example of the POA process showing the generation of optimal path combination for this elevator scheduling problem is depicted in Fig. 4.

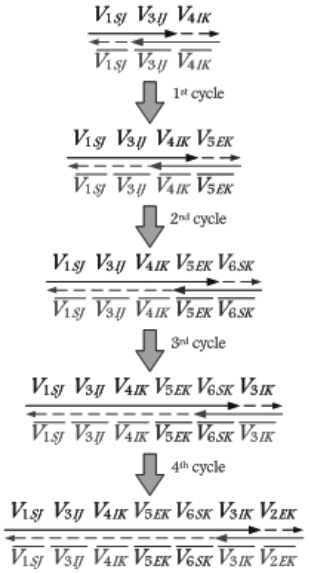

Fig. 4 POA for elevator's optimal path. The continuous arrows represent the synthesized DNA sequence and dotted arrows represent the elongated part during polymerization. The arrowhead indicates the 3' end.

Step 5. The optimal path combinations among many other alternative path combinations have to be filtered from the initial pool solution. This filtering process copies the target DNA duplex exponentially using polymerase chain reaction (PCR) process [13]. PCR proceeds in cycles of 3 steps at different temperatures: denaturation (95°C), involves separation of the double strand DNA molecules, 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 DNA molecules. This sequence is repeated causing an exponential growth in the number of target DNA molecules. For this problem, all the DNA molecules containing start node V_{1SJ} and end node V_{2EK} are amplified exponentially. Numerous amounts of DNA strands that represents the start node V_{1SJ} and end node V_{4EJ} and V_{2EK} passing through all possible travel path combinations will be presented once the PCR operation is accomplished.

Step 6. Finally, gel electrophoresis [14], [15] is performed onto the output solution of the PCR in order to separate all the possible travel path combinations according to its length. The gel electrophoresis image is then captured, where the DNA duplex representing the shortest path starting from V_{1SJ} and end node V_{4EJ} and V_{2EK} could be visualized representing the required optimal path solution of the problem.

4. In Vitro Experiment Setup and Result

An *in vitro* experiment is carried out in order to verify the designed DNA computing approach to solve the elevator scheduling problem. This experiment involves POA for initial pool generation, PCR for DNA sequence amplification and gel electrophoresis to visualize the computation output.

The POA for initial pool generation is performed in a 100 μl solution consisting of 64.0 μl distilled water (Maxim Biotech), 15.5μl oligos (Proligo Primers & Probes, USA), 10 μl dNTP (TOYOBO, Japan), 10 μl 10× KOD dash buffer (TOYOBO, Japan), and 0.5 μl KOD dash polymerase (TOYOBO, Japan). The solution is then subjected to POA reaction of 25 cycles where the different temperatures for each cycle are 94ºC for 30 s, 55ºC for 30 s, and 74ºC for 10 s respectively.

PCR is then performed onto the POA solution for DNA amplification in order to select the paths that begin with node V_{1SJ} and end at nodes V_{4EJ}, and V_{2EK}. PCR is performed in a 25 μl solution consisting of 17.875 μl distilled water (Maxim Biotech), primers V_{1SJ}, $\overline{V_{4EJ}}$, and $\overline{V_{2EK}}$ of 0.5 μl each, 1 μl POA template, 2.5 μl dNTP (TOYOBO, Japan), 2.5 μl 10× KOD dash buffer (TOYOBO, Japan), and 0.125 μl KOD dash polymerase (TOYOBO, Japan). The solution is subjected to PCR reaction of 25 cycles where the different temperatures for each cycle are 94ºC for 30 s, 55ºC for 30 s, and 74ºC for 10 s respectively, the same as POA process.

Finally, the resulting PCR solution is subjected to gel electrophoresis for 30 minutes in order to visualize the computation result. SYBR Gold (Molecular Probes) is used to stain the gel after gel electrophoresis process before the gel image is captured.

The captured image for the POA and PCR process is shown in Fig. 5. Here, lane *M* denotes 20bp ladder while lanes 1 and 2 denote POA and PCR product respectively. It is clearly seen from the POA gel image that the band is blur denoting that all possible travel path combinations are successfully generated. The PCR gel image shows 4 bands indicating all the four possible travel paths, i.e. $G(A, B)_3$ = 230bp, $G(A, B)_1$ = 250bp, $G(A, B)_4$ = 280bp and $G(A, B)_2$ = 300bp. This confirms the expected result that the optimal elevator's travel path is given by $G(A, B)_3$ = 230bp = 115 s.

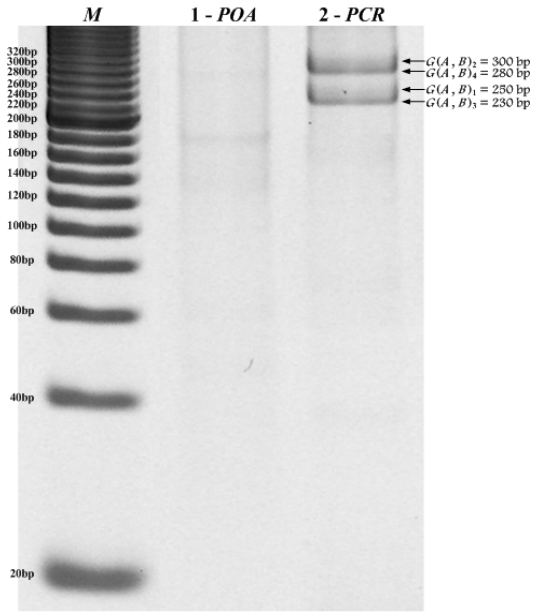

Fig. 5 Captured image of gel electrophoresis showing computing output.

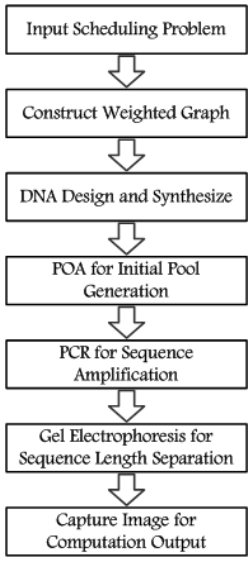

Fig. 6. DNA computing approach algorithm for solving elevator scheduling problem.

5. Conclusions

A new method to solve an elevator scheduling problem using DNA computing approach has been presented and discussed in detail in this paper. An *in vitro* experiment to verify the expected result has successfully been carried out. The design methodology and experimental implementation procedures presented as summarized in Fig. 6 shows that this type of engineering problem is applicable and achievable to be solved using the DNA computing approach. For a larger problem with M elevators, N floors and Y hall calls, all the M^Y travel path combinations can be represented by specific DNA sequences synthesized using the rule stated. POA and PCR can thus be performed to extract the required computing output from the gel electrophoresis image. With this successful design and implementation, the applicability and feasibility of DNA computing approach could therefore be extended into many more complex problems of this type of nature.

Acknowledgments

The first author is very grateful to Universiti Malaysia Sarawak (UNIMAS), Malaysia for granting a study leave at Meiji University, Japan.

References

[1] L.M. Adleman, "Molecular computation of solutions to combinatorial problems," *Science*, vol. 266, pp. 1021-1024, 1994.

[2] L.M. Adleman, "Computing with DNA," *Scientific American*, pp. 34-41, 1998.

[3] J.D. Watson, and F.H.C. Crick, "A Structure for Deoxyribose Nucleic Acid", Nature, Vol. 171, pp. 737-738, 1953.

[4] A. Narayanan, and S. Zorbalas, "DNA algorithms for computing shortest paths," *Proceedings of Genetic Programming*, pp. 718-723, 1998.

[5] Y. Yamamoto, A. Kameda, N. Matsuura, T. Shiba, Y. Kawazoe, and A. Ahochi, "Local search by concentration-controlled DNA computing," *International Journal of Computational Intelligence and Applications*, vol. 2, pp. 447-455, 2002.

[6] J.Y. Lee, S.Y. Shin, S.J. Augh, T.H. Park, and B.T. Zhang, "Temperature gradient-based DNA computing for graph problems with weighted edges," *Lecture Notes in Computer Science*, Springer-Verlag, vol. 2568, pp. 73-84, 2003.

[7] Z. Ibrahim, Y. Tsuboi, O. Ono, and M. Khalid, "Direct-proportional length-based DNA computing for shortest path problem," *International Journal of Computer Science and Applications*, vol. 1, issue 1, pp. 46-60, 2004.

[8] M. S. Muhammad, S. Ueda, O. Ono, and M. Khalid, "DNA-based computing for solving elevator scheduling problem",

3rd International Conference on Computer Applications (ICCA2005), pp. 507-514, 2005
[9] M. S. Muhammad, S. Ueda, O. Ono, J. Watada, and M. Khalid, "Solving Elevator Scheduling Problem Using DNA Computing Approach", *Advances in Soft Computing, Springer*, pp. 359-370, 2005
[10] F. Udo, S. Sam, B. Wolfgang, and R. Hilmar, "DNA sequence generator: A program for the construction of DNA sequences," *Proceedings of the Seventh International Workshop on DNA Based Computers*, pp. 23-32, 2001.
[11] P.D. Kaplan, Q. Ouyang, D.S. Thaler, and A. Libchaber, "Parallel overlap assembly for the construction of computational DNA libraries," *Journal of Theoretical Biology*, vol. 188, issue 3, pp. 333-341, 1997.
[12] J.Y. Lee, H.W. Lim, S.I. Yoo, B.T. Zhang, and T.H. Park, "Efficient initial pool generation for weighted graph problems using parallel overlap assembly," *Preliminary Proceeding of the 10th International Meeting on DNA Computing*, pp. 357-364, 2004.
[13] J. P. Fitch, *Engineering Introduction to Biotechnology*, SPIE Press, 2001.
[14] G. Paun, G. Rozenberg, and A. Salomaa, "DNA computing: New computing paradigms," *Lecture Notes in Computer Science*, Springer-Verlag, vol. 1644, pp. 106-118, 1998.
[15] Y. Yamamoto, A. Kameda, N. Matsuura, T. Shiba, Y. Kawazoe, and A. Ahochi, "A separation method for DNA computing based on concentration control," New Generation Computing, vol. 20, no. 3, pp. 251-262, 2002.

Mohd Saufee Muhammad received his B.Eng (Electronic Computer Systems) from University of Salford, UK in 1996 and MSc in Engineering (Electrical) from Western Michigan University, USA in 2000. He is a staff member of Department of Electronics, Faculty of Engineering, Universiti Malaysia Sarawak, Malaysia since 2000. He is currently on a study leave pursuing his PhD at the Institute of Applied DNA Computing, Meiji University, Kanagawa, Japan. His research interests include DNA computing and its applications, artificial intelligence and image processing. He is a student member of Institute of Electrical and Electronics Engineers (IEEE), IEEE Computational Intelligence Society (ICIS), and IEEE Signal Processing Society (ISPC).

Osamu Ono received his B.Sc., M.Sc. and Ph.D. degrees in electrical engineering from Waseda University, Japan, in 1974, 1976, and 1979, respectively. He was previously a Visiting Professor of the Institute of Automatic Control Technology, Technical University of Munich, Germany, and also a Visiting Professor of National University of Singapore, and Malaysia University of Technology. He is currently a Professor of Electrical and Electronic Engineering at Meiji University, Tokyo, Japan. He is also the Director of Institute of Applied DNA Computing of Meiji University and Chairman of Department of Electrical and Electronic Engineering, Meiji University (2001-2004). He is a Dean and an Editor of *Japan Society for Simulation Technology* (JSST). His research interests include biocomputing and artificial intelligence, evolutional and unconventional computing, biorobotics vision, neural image, signal processing, and humanoid mobile robotics. Currently, his research interest is mainly in the nanobioscience and applications of molecular or DNA computing. He is the Tokyo branch Director of Japan Institute of Electrical Engineers (JIEE). He is also the President of Advanced High Speed Image Processing Technology Inc. of Meiji University Technology Licensing Organization (TLO) Incubation Centre.