DNA Computing and its Application

Junzo Watada

Graduate School of Information, Production and Systems, Waseda University,
2-7 Hibikino, Wakamatsu, Kitakyushu 808-0135 Japan
junzow@osb.att.ne.jp

1 Introduction

The objectives of this chapter are twofold: firstly to introduce DNA computation, and secondly to demonstrate how DNA computing can be applied to solve large, complex combinatorial problems, such as the optimal scheduling of a group of elevators servicing a number of floors in a multi-storey building.

Recently, molecular (or wet) computing has been widely researched not only within the context of solving NP-complete/NP-hard problems that are the most difficult problems in NP, but also implementation by way of digital (silicon-based) computers [21]. We commence with a description of the basic concepts of 'wet computation', then present recent results for the efficient management of a group of elevators.

2 DNA Computing

The main idea behind DNA computing is to adopt a biological (wet) technique as an efficient computing vehicle, where data are represented using strands of DNA. Even though a DNA reaction is much slower than the cycle time of a silicon-based computer, the inherently parallel processing offered by the DNA process plays an important role. This massive parallelism of DNA processing is of particular interest in solving NP-complete or NP-hard problems.

It is not uncommon to encounter molecular biological experiments which involve $6 \times 10^{16}/ml$ of DNA molecules. This means that we can effectively realize 60,000 TeraBytes of memory, assuming that each string of a DNA molecule expresses one character. The total execution speed of a DNA computer can outshine that of a conventional electronic computer, even though the execution time of a single DNA molecule reaction is relatively slow. A DNA computer is thus suited to problems such as the analysis of genome information, and the functional design of molecules (where molecules constitute the input data).

DNA consists of four bases of molecule structure, named adenine (A), guanine (G), cytosine (C) and thymine (T). Moreover, constraints apply to connections between these bases: more specifically, A can connect only with T, and G only with C – this connecting rule is referred to as 'Watson-Crick complementarity'. This property is essential to realize the *separate* operation (discussed later). In other words, it is possible to separate a partial string of characters 'ad' so that a DNA sequence complementary to the DNA denoting 'ad' is marked, input into a test tube, hybridized to form a double strand helix of DNA, then abstracted. Further, this property enables us to randomly create a set of character strings according to some rule.

Since [1] described a method for solving a directed Hamiltonian path problem with 7 cities using DNA molecules, researchers have pursued theoretical studies to realize *general* computation using DNA molecules [for example, [23]. [2] has developed a computational model to realize – via experimental treatment of DNA molecules – operations on multiple sets of character strings, following the encoding of finite alphabet characters onto DNA molecules.

As previously mentioned, DNA molecules can be used as information storage media. Usually, DNA sequences of around 8-20 base-pairs are used to represent bits, and numerous methods have been developed to manipulate and evaluate these. In order to manipulate a wet technology to perform computations, one or more of the following techniques are used as computational operators for copying, sorting, splitting or concatenating the information contained within DNA molecules:

- ligation,
- hybridization,
- polymerase chain reaction (PCR),
- gel electrophoresis, and
- enzyme reaction.

In the following subsection we briefly describe the specific bio-chemical process which serves as the basis of *our* DNA computing approach.

A DNA computer performs wet computation based on the high ability of special molecule recognition executed in reactions among DNA molecules. Molecular computation was first reported in [1], where it was found that a DNA polymerase – which incorporates an enzyme function for copying DNA – is very similar in function to that of a Turing machine. DNA polymerase composes its complementary DNA molecule using a single strand helix of a DNA molecule as a mold. On the basis of this characteristic, if a large amount of DNA molecules are mixed in a test tube, then reactions among them occur simultaneously. Therefore, when a DNA molecule representing data or code reacts with other DNA molecules, this corresponds to super-parallel processing and/or a huge amount of memory in comparison with a conventional (electronic) computer.

2.1 Encoding Scheme

In any DNA computational procedure, the main challenge is to encode each object of interest into a DNA sequence. A correct design is essential in order to ensure optimal results; an incorrect design could result in wrong sequences following the ligation process.

Fig. 1. Droppers for spoiding and hybridizing

Ligation and Hybridization

When DNA sequences are dropped in a test tube using a micro pipettor (Figure 1), the DNA sequences recombine with each other by means of some enzyme reaction, this process being referred to as 'ligation'. All DNA sequences to be used in the experiment – together with their complements – are mixed together in a single test tube. Normally the oligonucleotide or DNA mixture is heated to 95^o centigrade (celsius) and cooled to 20^oC at 1^oC per minute for hybridization, as indicated in Figure 1. The reaction is then subjected to a ligation. At the end of this process, a certain DNA sequence will ligate together with another DNA sequence in order to produce a new sequence.

Polymerase Chain Reaction (PCR)

Polymerases perform several functions, including the repair and duplication of DNA. PCR is a method for amplifying DNA *in vitro*. PCR is a process that quickly amplifies the amount of specific DNA molecules in a given solution, using primer extension by polymerase. Each cycle of the reaction doubles the quantity of this molecule, leading to an exponential growth in the number of sequences. It consists of the following key processes:

1. Initialization: a mix solution of template, primer, dNTP and enzyme is heated to $94 - 98°C$ for $1 - 9$ minutes to ensure that most of the DNA template and primers are denatured;
2. Denaturation: heat the solution to $94 - 98°C$ for $20 - 30$ seconds for separation of DNA duplexes;
3. Annealing: lower the temperature enough (usually between $50 - 64°C$) for $20 - 40$ seconds for primers to anneal specifically to the ssDNA template;
4. Elongation/Extention: raise temperature to optimal elongation temperature of *Taq* or similar DNA polymerase ($70 - 74°C$) for the polymerase adds dNTP's from the direction of $5'$ to $3'$ that are complementary to the template;
5. Final Elongation/Extention: after the last cycle, a $5 - 15$ minutes elongation may be performed to ensure that any remaining ssDNA is fully extended.

Step 2 to 4 is repeated for $20 - 35$ times; less cycles results less product, too many cycles increases fraction of incomplete and erroneous products. PCR is a routine job in the laboratory that can be performed by an apparatus named *thermal cycler*.

Denaturation Temperature Gradient PCR

Denaturation temperature gradient PCR (DTG-PCR) is a modified PCR method that the denaturation temperature changes with cycle [20]. In DTG-PCR, conventional PCR is performed where the temperature of the denaturation step (the step 2 of PCR procedure above mentioned) is gradually increased in cycles.

Quantitative PCR

Quantitative PCR (Q-PCR) is a modification of the PCR used to rapidly measure the quantity of DNA, complementary DNA (cDNA) or RNA present in a sample. It may be used to determine a DNA sequence is presented in a sample, and the number of its copies produced in PCR.

Affinity Separation

The objective of the affinity separation process is to verify whether each of the data includes a certein sequence. This process permits single strands containing a given subsequence v to be filtered out from a heterogeneous pool of other sequences. After synthesizing strands complementary to v and attaching them to magnetic beads, the heterogeneous solution is passed over the latter. Those strands containing v anneal to the complementary sequence and are retained; those strands not containing v pass through and are discarded.

Normally, in this process a double stranded DNA is incubated with the Watson-Crick complement of data that is conjugated to magnetic beads. A bead is attached to a fragment complementary to a substring then a magnetic field is the used to pull out all of the DNA fragments containing such sequence. The process is then repeated.

Fig. 2. Electrophoresis

Gel Electrophoresis

Gel electrophoresis is an important technique for sorting DNA strands by their size [4]. Electrophoresis enables charged molecules to move in an electric field, as illustrated in Figure 2. Basically, DNA molecules carry negative charge. Thus, when we place them in an electrical field, they tend to migrate towards a positive pole. Since DNA molecules have the same charge per unit length, they all migrate with the same force in an electrophoresis process. Smaller molecules therefore migrate faster through the gel, and can be sorted according to size (usually agarose gel is used as the medium here). At the end of this process the resultant DNA is photographed, as indicated in Figure 3.

3 Comparison with Conventional Computing

Now DNA computing employs completely different tactics when allocating an independent letter code (such as ATCG, GTAC or CAAC) to each sample. Next, DNA sequences corresponding to the number of possible combinations are prepared. After they are hybridized in super parallel fashion, the remaining DNA fragments are amplified to obtain an answer sequence – note that this procedure is carried out only once [15].

Fig. 3. Camera

The main benefit of using DNA computation to solve complex problems is that all possible solutions are created *concurrently* – in other words, it offers massively parallel processing. By contrast, humans – as well as most electronic computers – solve problems in a step-by-step manner (in other words, sequentially). DNA provides other benefits, including low cost, and energy efficiency [**?**].

The main steps in DNA computing are:

1. *Separate* (T, s)*:* this operation separates a given set T into the set $+(T, s)$ of characters, including character string s and the set $-(T, s)$ of character strings that do not contain character string s. This operation corresponds to abstract experimentation on DNA molecules in a test tube.

2. *Mix:* this operation mixes sets T_1 and T_2 into the union set $T_1 \cup T_2$. This operation corresponds to mixing test tubes T_1 and T_2.
3. *Detect* (T)*:* this operation returns 'YES' if the test tube T is not empty, and 'NO' if it is empty. The operation corresponds to an experimental procedure that detects the existence of DNA molecules by the electrophoretical fluorescent method.
4. *Amplify* (T)*:* this operation corresponds to creating multiple sets T_1 and T_2 with the same contents as the given set T. This corresponds to an experimental treatment that amplifies the amount of molecules using polymerase chain reaction (PCR).

Now from the perspective of DNA computing, the most important characteristic of a DNA molecule is its Watson-Crick complementarity.

In Adleman's model, a set of character strings – computed using hybridization – is computed according to the four steps described above. Using this computation, an NP-complete problem can be solved using an algorithm based on production-detection PCR. DNA computers can be used to solve real-world problems using this method.

4 Applications of DNA Computing

The theory of DNA computing is discussed in [1, 26, 28]. DNA computing has been applied to various fields, including nanotechnology, combinatorial optimization [24, 25], boolean circuit development [28], and of particular relevance to the present section, scheduling [21, 23, 16, 18, 28, 31].

5 Approaches to Optimization and Scheduling

We have a long history of a mathematical way to solve optimization problems. But there is limitation in a mathimatical method where many problems are left unsolved. Beyond such mathematical methods, "problem solving" tries to mimic such a human or empirical way as a rule-of-thumb method. It is most prevailed after a von Neuman computer was invented since 1945. It is named aritificial intelligence that brought a gold era in late 1970s and early 1980s, even if logic approaches including a production system, a predicate logic system, a semantic network and a frame system should face combinatorial explosion that provides challanges for us. On the other hand, in order to mimic a human thinking way many methods were proposed such as fuzzy systems, genetic algorithms, chaotic systems, neural networks and so on which are named as a soft computing. If we intend to solve a real scale of problems we have to overcome the combinatorial explosion. Espeically, NP-complete probems cannot be solved by a present silicon-based computer.

Genetic algorithm

A genetic algorithm (GA) is a kind of soft computing with genetic mechanism in organisms, searches optimal values, when it assumes number of control patterns in repeating GA simulation. Genetic algorithm is employed to group control of elevators [8, 13]. The setting of parameters in a group control system of elevators is also hard to manage on manual basis. Cortes et al. proposed genetic algorithm to select good setting of parameters [9].

Neural network

In the field of neural networks, the optimization of an evaluation function is pursued employing back-propagation learning depending on past transactions [27, 30]. In fuzzy logic method, The easiness of updating rules is welcome in comparison with GAs and neural networks [14, 17].

Fuzzy logical computation and others

It is not only limited in the fields of AI, GAs, NNs and Fuzzy Logical Computation but also such soft computing methods as enhanced learning [10] evolutional strategy [7] and genetic network programming [11, 12] are employed in optimalizing group control of elevators.

New demands

Recently, a new generation type of elevator systems such as an elevator group supervisory control system(EGSCS), [5, 6] is developed to satisfy the various needs of users and enable the verticalization of buildings [3]. According such new types of elevator systems place more additional constraints on the group management. When searching a large number of alternatives, they require fast processing and large computer overhead.

Therefore, the development of group management systems have intensively studied for the improvement of elevator's transportation efficiency and convenience. The usage condition of an elevator system is changed depending on time and customers. Business people don't like to wait so many minutes but people at a hotel do not like a crowded elevator even if it is fast transported. The elevetor system is required to fulfill such passengers' different preferences. On the other hand, in DNA computing, since the computing mimics DNA copy mechanism in chemical reaction, processing is inherently (massively) parallel.

6 Elevator Management System

Multiple elevators are commonly used in high-rise buildings ('skyscrapers'). Effective control of such multiple elevators is essential. The overall aim in controlling a group of elevators is to satisfy the time constraints of all passengers,

at the same time providing the most efficient system. The basic problem is to decide *which* elevator should stop at a particular floor where passengers are waiting to go up(down).

Even in peak (rush) hours, it is possible to find *all* elevators moving in the same direction, or alternatively all elevators arriving simultaneously at the same floor. In order to resolve such situations, all elevators need to be optimally assigned to passengers, regardless of the latter's changing arrival times at the various floors in the multi-storey building.

The group control system selects elevator movement patterns according to random changes in traffic volumes and/or driving management, or in the case of an accident. Such group control realizes comfortable, safe and economical management of elevators.

Suppose that a building has N floors and m elevators. Table 1 shows the current position of each elevator, the destinations of passengers in each elevator, together with the intended travel direction of passengers waiting on each floor. Figure 4 illustrates this situation using a graphical expression. In Figure 4, the graph in the left side shows an up-going movement and the graph in the right side a down-going movement, respectively. Elevators 1, 2 and M stop at floors 2, 3 and $N-1$, respectively. On the other hand there are people on floor 1 who direct upword, people on floors $N-1$ and N who direct downward. For example, at time t, Elevator 1 at floor 2 has passengers who are going to floors 3, 4 and $N-1$. On the other hand, there are passengers on floors N, $N-1$ and 1 who are going *down*, *down* and *up*, respectively.

Table 1. Elevator information at time-t

Floor	Queuing	Elevator-1	Elevator-2	$\cdots$	Elevator-M
N	$\downarrow$				
N - 1	$\downarrow$		(1,3,5)		
$\vdots$					
3					(4, N-1,N)
2		(3,4,N-1)			
1	$\uparrow$				

In this research, as indicated, the input information to the elevator management system is as follows:

1. The present position of each elevator;
2. The destination floors of each elevator. Required floor numbers where passengers in each elevator are going to in are input to the system; and
3. The floors from which each elevator has been called. People on each floor can indicate their direction but they cannot input their destinating floor.

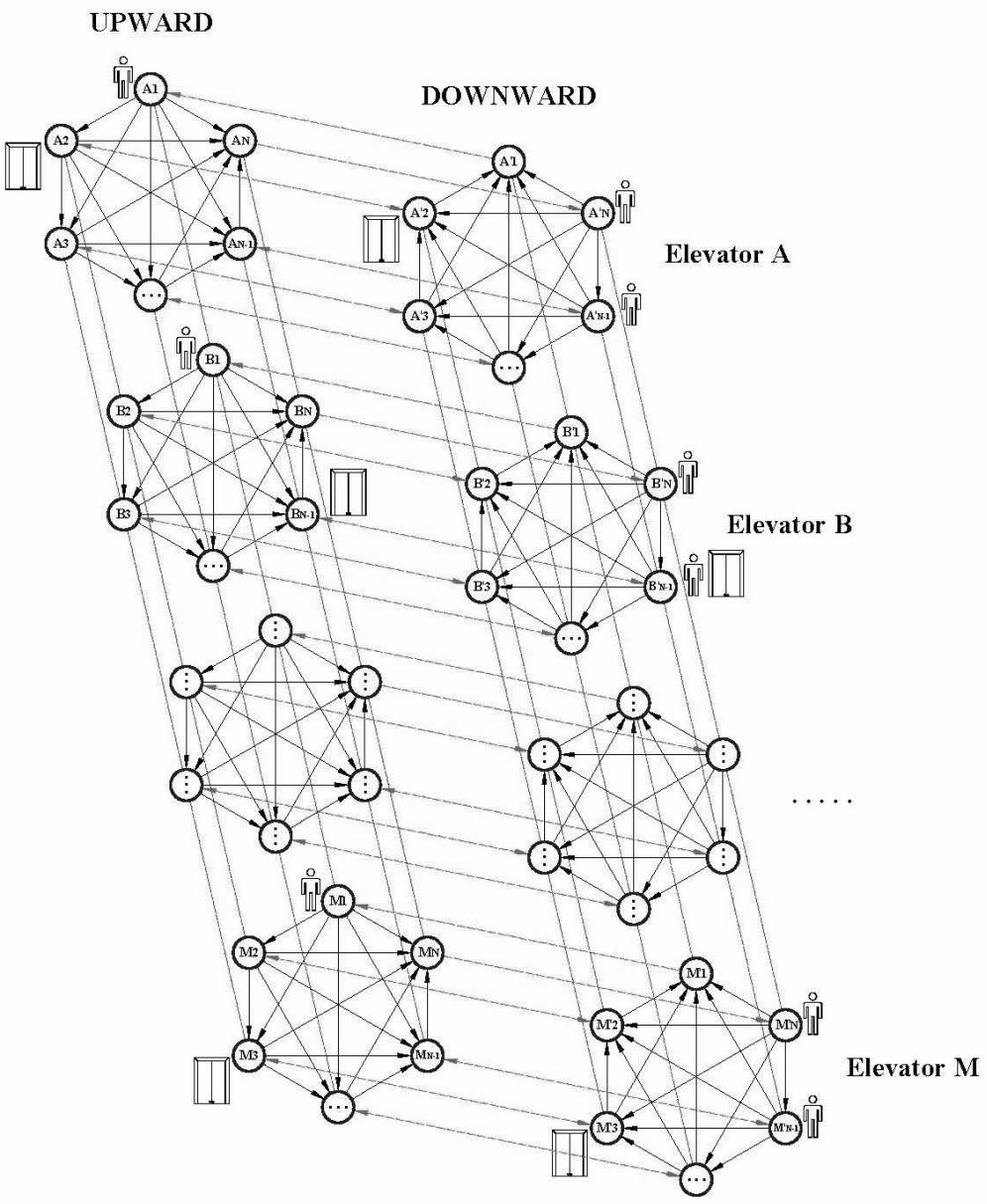

Fig. 4. Whole paths of M elevators

Table 2 shows that Elevator-1 is at Floor 2 and has passengers who are going to floors 3, 4 and $N-1$. There are people who are going up on Floor 1, down on Floor $N-1$ and down on Floor N.

Based on this information, the problem is to efficiently manage *all* elevators.

6.1 Restrictions on Elevator Movements

The problem is to determine the optimal scheduling of *all* M elevators – in other words to provide the shortest overall wait time – given the wait queue and the initial position of each elevator at any time t. Let us denote the following variables:

$|d(m) - o(m)|$ is the total number of floor elevator moves
C_t is the time costs for elevator traveling between adjacent floors
C_s is the time costs for elevator stops

The elevator moves between floors, denoted m, must be consecutive, *i.e.*,

$$d(m_i) = o(m_{i+1}) \vee 1 \leq i < N \tag{1}$$

where

$o(m)$: original floor
$d(m)$: destination floor

The traveling time T between floors can be represented as

$$T(|d(m) - o(m)|) = \begin{cases} [|d(m) - o(m)|]C_t + C_s, & m\text{=destinated;} \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

The output of the graph G given by the sum of the costs thus represents the total time of traveling of the elevator E, *i.e.*,

$$G(E) = \sum_{m=1}^{N} T(|d(m) - o(m)|) \tag{3}$$

For a building with M elevators, the graph of a single elevator movements shown in Table 2 can be duplicated M times in representing the whole paths of elevator traveling. The total traveling time of M elevators can thus be calculated by summing up the traveling time of each single elevator as

$$G(E_1, E_2, \ldots, E_{M-1}, E_M) = \sum_{k=1}^{M} G(E_k) \tag{4}$$

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

$$O = min\{G(E_1 E_1, E_2, \ldots, E_{M-1}, E_M)\} \tag{5}$$

6.2 Elevator Scheduling

Now since we have m elevators, we can duplicate m graphs and connect between all vertices in the one graph. Figure 4 shows the case where $m = 2$.

The left-side graph in the Figure 4 illustrates all paths which are going upward and the right-side graph shows all paths are going downward. Elevator A at floor 1 can move to all floors from 2 to n. But when Elevator A is at floor j, it can mover to all floors from $j + 1$ to n but cannot mover to the downward such as to floor 1 to i. The right-side graph shows the same situation concerning the downward movement of the same Elevator-i. The connections between both the graphs show the changing of the direction from downward to upward or from upward to downward. These changes of the direction happen on all floors. The connections among elevators on the same floor show that passengers can move from one elevator to another elevator. But these movements are not considered in the computations because such the movements are pursued depending on the preference of passengers.

It is therefore sufficient that one of the m. elevators may reach the calling floor on the line of graph A, B, $\cdots$, m. Let us construct a graph for each case where elevator-A or elevator-B, $\cdots$, and elevator-m reach a floor where the button has been pushed.

Considering all combinations, let us calculate the shortest path of each graph A, B, $\cdots$, m. Suppose that $f(1, 2, \cdots, m)$ denotes the largest value of graphs A, B, $\cdots$, m. By calculating all combinations $f_x(A, B, \cdots, m)$, we can obtain the optimal allocation of elevators by selecting the minimum value of $f_x(A, B, \cdots, m)$, where x denotes the number of combinations. For example, when the number of elevators is 2 and the number of calling floors is 3, the number of combinations is 2^3.

It is possible to represent elevator-B by a graph, just as we did in the case of elevator-A. Figure 5 illustrates the graph of all paths of the two elevators A and B.

Table 2. Elevator management information

Floor	Calling	Elevator-A	Elevator-B
6			(2,3)
5	↓		
4	↑		
3	↓		
2			
1		(3,5)	

Now, elevator-A is currently at floor-1 and its destination floors are 3 and 5. The destination floors of elevator-B at floor-6 are 2 and 3. There are also

upward calls. Figure 5. shows the floors with an elevator mark where one or both elevators should stop.

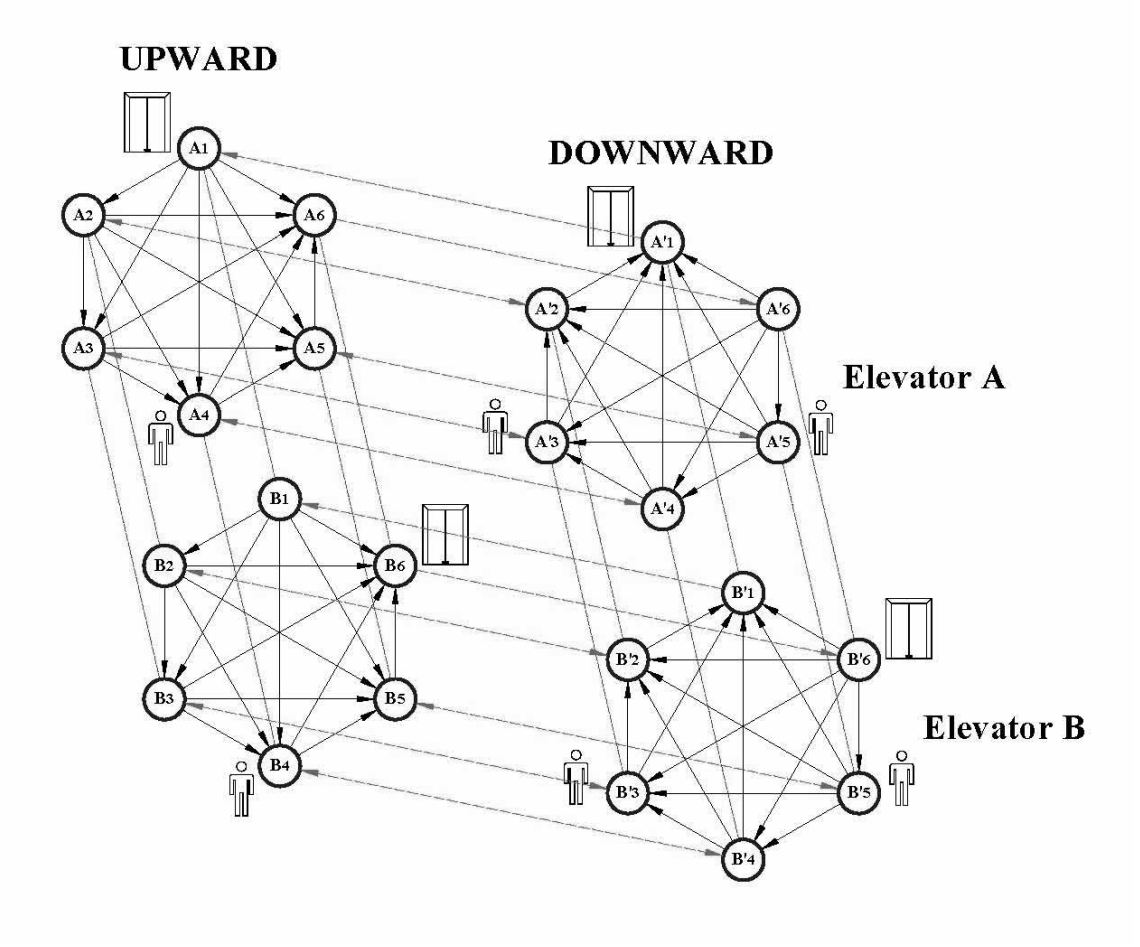

Fig. 5. Whole paths of two elevators

In this case, if the destination floor has been decided for elevator A or B, then the other one is automatically assigned to the destination floor. As a result, it is necessary to calculate the optimal paths for *two* kinds of graphs for elevators A and B. The larger value for both elevators A and B is denoted by $f_x(A, B)$.

The problem here is to select the smallest value – $\min(f(A, B), (x = 1, 2, \cdots, 8))$. The obtained schedule which gives the smallest value is the opti-

mal solution for elevators A and B. There are 16 kinds of graph which can be obtained from 8 combinations. The next Section shows how to calculate the shortest path schedule for both elevators.

7 Bio-soft Computing Based on DNA Length

In DNA computation, each base sequence is assigned to a floor, as shown in Table 3.

Table 3. Correspondence between floors and base sequences

1	2	3	4	5	6
$AAAA$	$CCCC$	$TTTT$	$ATAT$	$GAGA$	$GGGG$
1'	2'	3'	4'	5'	6'
$CACA$	$TCTC$	$TGTG$	$GCGC$	$CAGT$	$GATC$

Let us denote the connection between two base sequences corresponding to each floor as the movement between two floors.

Let us assign an appropriate length of DNA sequence to the edge weight:

$$\begin{aligned} rll\psi_k &= f(k) + T_E \\ \psi_1 &= f(1) + T_E \text{ string of two random characters}(ZZ) \\ &\vdots \\ \psi_5 &= f(5) + T_E \text{ string of ten random characters} \\ &\qquad (YYYYYYYYYYYYYYYYYY) \end{aligned} \tag{6}$$

$$\begin{aligned} rlf(A,B) &= \max(4+4+4+4+4, 4+2+4+4+4+2+4+4+4+4) \\ &= \max\{20, 36\} = 36 \end{aligned} \tag{7}$$

These values show the time spent moving between two floors, which are combined in the base sequence.

In this problem all movements between floors comprise 30 roots. Let us produce DNA sequences corresponding to these roots in Table 3.

Let us produce DNA fragments corresponding to a floor in Table 1 and DNA fragments corresponding to a root in Table 2. Next, place these DNA fragments and combining polymerase in the same test tube and store the test tube at an appropriate temperature; all combinations will be automatically created.

Table 4. Representation of roots by DNA sequence (edge DNA oligonucleotides)

$1 \to 2$	$AAZZCC$	$6' \to 5'$	$GAZZGT$
	$TTEEGG$		$CTEECA$
$1 \to 3$	$AAWWWWTT$	$6' \to 4'$	$GAWWWWCG$
	$TTFFFFAA$		$CTFFFFGC$
$1 \to 4$	$AAVVVVVVAT$	$6' \to 3'$	$GAVVVVVVTG$
	$TTHHHHHHTA$		$CTHHHHHHAG$
$1 \to 5$	$AAXXXXXXXXGA$	$6' \to 2'$	$GAXXXXXXXXTC$
	$TTIIIIIIIICT$		$CTIIIIIIIIAG$
$1 \to 6$	$AAYYYYYYYYYYGG$	$6' \to 1'$	$GAYYYYYYYYYYCA$
	$TTJJJJJJJJJJCC$		$CTJJJJJJJJJJGT$
$2 \to 2'$	$CCTC$	$5' \to 5$	$CAZZGA$
	$GGAG$		$GTEECT$
$2 \to 3$	$CCZZTT$	$5' \to 4'$	$CAZZCG$
	$GGEEAA$		$GTEEGC$
$2 \to 4$	$CCWWWWAT$	$5' \to 3'$	$CAWWWWTG$
	$GGFFFFTA$		$GTFFFFAC$
$2 \to 5$	$CCVVVVVVGA$	$5' \to 2'$	$CAVVVVVVTC$
	$GGHHHHHHCT$		$GTHHHHHHAG$
$2 \to 6$	$CCXXXXXXXXGG$	$5' \to 1'$	$CAXXXXXXXXCA$
	$GGIIIIIIIICC$		$GTIIIIIIIIGT$
$3 \to 3'$	$TTTG$	$4' \to 4$	$GCAT$
	$AAAC$		$CGTA$
$3 \to 4$	$TTZZAT$	$4' \to 3'$	$GCZZTG$
	$AAEETA$		$CGEEAC$
$3 \to 5$	$TTWWWWGA$	$4' \to 2'$	$GCWWWWTC$
	$AAFFFFCT$		$CGFFFFAG$
$3 \to 6$	$TTVVVVVVGG$	$4' \to 1'$	$GCVVVVVVCA$
	$AAHHHHHHCC$		$CGHHHHHHGT$
$4 \to 4'$	$ATCG$	$3' \to 3$	$TGTT$
	$TAGC$		$ACAA$
$4 \to 5$	$ATZZGA$	$3' \to 2'$	$TGZZTC$
	$TAEECT$		$ACEEAG$
$4 \to 6$	$ATWWWWGG$	$3' \to 1'$	$TGWWWWCA$
	$TAFFFFCC$		$ACFFFFGT$
$5 \to 5'$	$GACA$	$2' \to 2$	$TGCC$
	$CTGT$		$CTGG$
$5 \to 6$	$ATZZGG$	$2' \to 1'$	$TGZZCA$
	$TAEECC$		$CTEEGT$
$6 \to 6'$	$GGGA$	$1' \to 1$	$CAAA$
	$CCCT$		$GTTT$

Various DNA sequences are automatically created by combining fragments shown as each floor in Table 2 and filaments shown as each movement root in Table 2. These DNA sequences correspond to combinations of feasible solutions. In order to solve the schedule, DNA sequences which have 'AA' (the former two characters upward at the first floor) at the start, and 'GA' (the latter two characters upward at the fifth floor) at the end are detected, out of the many DNA sequences, using various polymerase.

As we know the floors where elevators should stop, only DNA sequences with $AA * TTTT * GA$ are selected – in other words those where the DNA sequences start with 'AA', pass through '$TTTT$', and terminate at 'GA'. Then, the shortest DNA sequence shows the optimal solution which starts at the 1st floor, stops at the 3rd floor, and reaches the 6th floor. This procedure can be abstracted by the weight of a DNA sequence, since long DNA sequences are heavy and short DNA sequences are light. At the end we check the DNA sequence and convert it to a floor number. The shortest roots for graphs 5 through 12 contain the following DNA sequences, and the length of these DNA sequences can be calculated.

The shortest sequence is the schedule where elevator-A stops at the 4th floor, and elevator-B stops at the 3rd and 5th floors. Therefore, the optimal schedule for elevators-A and B is obtained for the present state of elevators and calling floors. If this computation is pursued to obtain the optimal schedule whenever buttons are pushed at a calling floor, the schedule with the shortest wait time will always be obtained.

8 Bio-soft Computing with Fixed-length DNA

For the elevator dispatching problem, an N-story building equipped with M identical elevator cars given by up hall calls, down hall calls, and car calls. The optimal route is given by $min\{G(1, 2, \cdots, N)\}$.

The key factor to cost sequence design is the Tm of a DNA strand. The concept is to design the DNA sequences that have heavier weights with higher Tm then those lighter weights. DNA amplification and detection techniques often depend on oligonucleotide Tm. The Tm of a DNA duplex is defined as the temperature where one-half of the nucleotides are paired and one-half are unpaired [32]. The Tm indicates the transition from double helical to random coil formation and is related to the DNA `GC` base content [**?**]. Usually expressed as a percentage, it is the proportion of `GC`-base pairs in the DNA molecule or genome sequence being investigated. `GC`-pairs in the DNA are connected with three hydrogen bonds instead of two in the `AT`-pairs, which makes the `GC`-pair stronger and more resistant to denaturation by high temperatures. In our encoding scheme, the DNA sequences that represent floor nodes are fixed length, and costs are distinguished by Tm of the given DNA strands. This

design makes an oligonucleotide with lighter weight, which represent more economical path tend to have a lower Tm.

All the possible solutions are randomly generated by DNA hybridization and ligation with the oligonucleotides representing floors, edges, and costs. To satisfy the condition of an elevator dispatching problem, the route must begin and end at a specified node, and the route must pass by each consecutive floor until reaches the final destination.

The PCR can be applied to test the former requirement, which it is a technique for amplifying DNA that rapidly synthesize many copies of a specific DNA segment by providing specific complementary sequences (primers) and enzymes (DNA polymerases, for example, *Pfu* and *Taq*). The DNA strands corresponding to original floor and complement of final destination floor are used as two primers in two successive PCRs to reproduce the routes.

To test the latter requirement, agarose gel electrophoresis is applied. Agarose gel electrophoresis is a method to separate DNA strands by size, and to determine the size of the separated strands by comparison to strands of known length. All PCR products are sieved by agarose gel electrophoresis, and the unreasonable lengths are excluded. To verify the DNA strands pass by every consecutive floor, the product from the above step is affinity-purified with a biotin-avidin magnetic beads system.

To solve the elevator routing problem with our molecular algorithm, note that all possible end paths of elevator i are jointed with the start path of elevator $i+1$ so that the total output of the graph $G(1, 2, \cdots, N)$ representing the travel route of all elevators can be calculated.

DTG-PCR is a specified PCR protocol that modifies the denaturation temperature profile. If the denaturation temperature is decreased to a certain level in PCR, the DNA strands with denaturation temperatures lower than that temperature will be denatured and amplified. As the denaturation temperature is increased cycle by cycle in PCR, other DNA strands with higher denaturation temperature will also be amplified. However, the economical paths that have lighter weights will be amplified more and will occupy the major part of the solution and hence can be easily detected [20]. Based on the electrophoretic mobility of DNA strands in different Tm, the TGGE is applied to detect the the most economical route among other possibilities resulting from DTG-PCR.

The proposed bio-soft computing algorithm for solving the elevator dispatching problem is summarized following:

Step 0 Design the fixed-length DNA sequences with thermodynamic control in weight;
Step 1 Generate a random pool of solution by hybridization and ligation;
Step 2 Retrieve the strand that satisfy the condition of elevator dispatching problem by PCR and gel electrophoresis;
Step 3 Verify the strands that pass by every consecutive floor by affinity purification;

Step 4 Joint the strands of elevators by ligation;
Step 5 Sieve the economical path by DTG-PCR;
Step 6 Detect the most economical path by TGGE;
Step 7 Readout the most optimal route by DNA sequencing.

8.1 Empirical Study

A six-story building equipped with two identical elevator cars, A and B, is considered in the empirical study. As illustrated in Table 2, elevator A is currently at the 1st floor upward answers the car calls on 3rd and 5th floor. Elevator-B is at the 6th floor downward answers the car calls on 2nd and 3rd floor. In addition, hall calls are requested on 3rd, 4th, and 5th floor for down, up, and down, respectively. The objective is to find the optimal route for all elevators that fulfill all initial conditions and requirements defined. The optimal route will be given by $min\{G(A, B)\}$.

As listed in Table 5, each floor node is randomly associated with a 20-mer sequence of ssDNA, denoted F_i, which has a similar melting temperature due to node sequences should contribute equally to the thermal stability of paths. Weight sequences are designed to have different Tm depending on its weight. The lighter in weight, the lower the Tm is. In other words, more economical path has lower Tm. The edge between consecutive floors is generated by partial beginning node, cost sequence, and partial ending node. For each floor movement (edge) $i \rightarrow j$ in the graph, an oligonucleotide $F_{i\rightarrow j}$ is created that is the $3'$ 10-mer complement of F_i followed by the cost sequence of path length, and then the $5'$ 10-mer complement of F_j. The edge from floor 1 to floor 2, for example, the $3'$ 10-mer complement of F_1: '`CATGACAACG`' followed by the cost sequence of C_t: '`ATCTTGGATTTATTACCAAG`', then the $5'$ 10-mer complement of F_2: '`TGGCTACATG`', as illustrated in Fig. 6.

In this study, nearest-neighbor (N-N) model is applied to calculate the Tm, which it is the most accurate method for predicting the Tm of oligonucleotide DNA through interactions between neighboring bases. The enthalpy ($\triangle H$) and entropy ($\triangle S$) of adjacent bases is considered in the formula [29]. The Tm in this study was calculated with the initial concentration of $1nM$ oligonucleotide and $50mM$ salt.

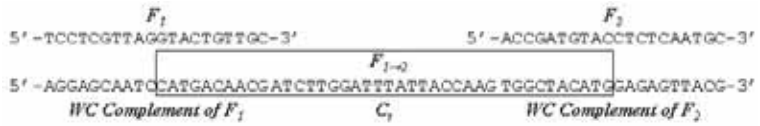

Fig. 6. Example of the encoding scheme. An oligonucleotide $V_{1\rightarrow 2}$ is created that is the $3'$ 10-mer of V_1 followed by the weight sequence of path length, and then the $5'$ 10-mer of V_2.

Table 5. DNA sequences associated with each floor node and cost for the six-floor elevator routing problem.

	DNA Sequence ($5' \rightarrow 3'$)	Tm ($^\circ C$)	GC Content (%)
Floor Nodes			
F_1	TCCTCGTTAGGTACTGTTGC	46.02	50
F_2	ACCGATGTACCTCTCAATGC	46.40	50
F_3	TGGTCAGCTAATGACGTGAG	46.42	50
F_4	GCGGTTCTAAATTCCGTCAC	46.51	50
F_5	ATTGGACCCAGATGCAAAGG	46.92	50
F_6	GTTAGACCTCGCGTTGCTAT	46.97	50
F_1'	GCGTAATCGTATCCGTGAGA	46.58	50
F_2'	TAGCCTTACGTACCGGCTTA	46.84	50
F_3'	CCGTAACGTATAGCGATGGA	46.22	50
F_4'	GACGGTATTGCGTAATTCGG	46.48	50
F_5'	ATCGGAATCGATCCGTATGC	46.88	50
F_6'	AGCTGGGATAAGGCATACCA	46.76	50
Costs			
C_t	ATCTTGGATTTATTACCAAG	36.98	30
C_s	GAGCCGACCAGCGACACCCA	55.84	70

The proposed fixed-length DNA based algorithm for solving the elevator dispatching problem began from generating of a random pool of possible routes by the hybridization of DNA strands that represent the floors and edges. All possible paths of the elevator dispatching problem was generated simultaneously under the massive parallelism of DNA molecules. For each F_i and for edge $i \rightarrow j$ were mixed in a single ligation reaction. Per [20], the added amount of an edge was varied according to weight, where as the weight increased, the amount was decreased. The oligonucleotide mixture was heated to $95^\circ C$ and cooled to $20^\circ C$ at $2^\circ C/min$ for hybridization. The reaction mixture was then subjected to a ligation.

The conventional gel electrophoresis excluded the unreasonable length of DNA strands from candidate pool. Then, the DNA strands that were not passing by every floor nodes between origin and destination was excluded by affinity separation. The complement of F_1 was conjugated to magnetic beads so that only those ssDNA molecules which contained the sequence F_1 annealed to the bound were retained. This process was repeated until each floor node was verified.

In DTG-PCR, the denaturation temperature started at low temperature in $70^\circ C$ in the beginning cycles of PCR, which is lower than the Tm of template strands. Then, the denaturation temperature was gradually increased until reached $95^\circ C$ and maintain at the same temperature for the remaining cycle. After this process, one main band was observed in the gel which contained two different DNA strands of the possible routes as shown in Table 6. These

strands, however, were of the same length which cannot be separated by the conventional gel electrophoresis.

Nevertheless, from the algorithm design, the weights have their distinct behaviors in Tm and thus the more economical path would have a lower Tm. Thus, TGGE can be used to filter the DNA strands that have lowest Tm among other strands of the same length. Based on the correlation of the melting characteristic of a DNA strand to its electromigration, the DNA strand of the most economical route would travel fastest in gel; hence, it can be distinguished from other possible routes. The Tm and its GC content among those possible routes are shown in Fig. 7, which the DNA strands corresponded to the route for elevator A: '$1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5$' that answers the hall call at 4th floor and 5th floor for up and down, respectively, and elevator B: '$6 \rightarrow 5 \rightarrow 4 \rightarrow 3 \rightarrow 2$' that answers the hall call at 3rd floor for down shown the optimal solution.

Table 6. Result from DTG-PCR. Two different DNA strands represent the possible routes of the same length which cannot be separated by conventional gel electrophoresis.

DNA Sequence ($5' \rightarrow 3'$)	Oligo Length (Bases)
TCCTCGTTAGGTACTGTTGCATCTTGGATTTATTACCAAG	
ACCGATGTACCTCTCAATGCGAGCCGACCAGCGACACCCA	
TGGTCAGCTAATGACGTGAGGAGCCGACCAGCGACACCCA	
GCGGTTCTAAATTCCGTCACGAGCCGACCAGCGACACCCA	
ATTGGACCCAGATGCAAAGGAGCTGGGATAAGGCATACCA	360
GAGCCGACCAGCGACACCCAATCGGAATCGATCCGTATGC	
ATCTTGGATTTATTACCAAGGACGGTATTGCGTAATTCGG	
GAGCCGACCAGCGACACCCACCGTAACGTATAGCGATGGA	
GAGCCGACCAGCGACACCCATAGCCTTACGTACCGGCTTA	
TCCTCGTTAGGTACTGTTGCATCTTGGATTTATTACCAAG	
ACCGATGTACCTCTCAATGCGAGCCGACCAGCGACACCCA	
TGGTCAGCTAATGACGTGAGGAGCCGACCAGCGACACCCA	
GCGGTTCTAAATTCCGTCACGAGCCGACCAGCGACACCCA	
ATTGGACCCAGATGCAAAGGAGCTGGGATAAGGCATACCA	360
ATCTTGGATTTATTACCAAGATCGGAATCGATCCGTATGC	
ATCTTGGATTTATTACCAAGGACGGTATTGCGTAATTCGG	
GAGCCGACCAGCGACACCCACCGTAACGTATAGCGATGGA	
GAGCCGACCAGCGACACCCATAGCCTTACGTACCGGCTTA	

9 Conclusion

"It is not the world that attracts attention now and that a usual physical law sways in the minute (nano) world but the world of a quantum-mechanics-

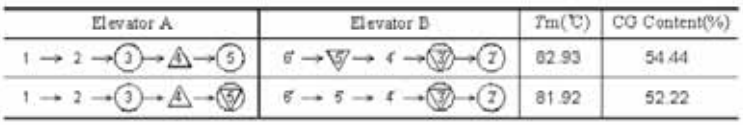

Fig. 7. The distinct behavior in melting temperature among the possible optimal solutions. ◯: car call; △: up hall call; ▽: down hall call.

law." This is described in famous 'uncertainty principle' which Heisenberg in Germany discovered in 1927.

Although at present computers are built on the model of a deterministic Turing machine, a new idea of a quantum Turing machine is not built into a model yet. The computer based on this idea is called a 'quantum computer', and the idea is required to go further beyond the present computer. Although this type of computer has yet to be produced in the real world, a present computer (von Neumann type computer) using the semiconductor should face the wall of combinatorial problems.

There are also many problems including the following:

- 'Preparation' and 'extraction' take too much time, and
- errors occur in copying DNA.

Although there are problems which must be solved to realize a DNA computer, it is expected as modern technology which will replace with the present von Neumann-type computer.

References

1. Adleman LM (1994) Molecular computation of solutions to combinatorial Problems, *Science*, 266: 1021-1024.
2. Adleman LM (1998) Computing with DNA, *Scientific American*, 279(2): 54-61.
3. Amano, M. Yamasaki and H. Ikejima (1995) The Latest Elevator Group Control System, In G.C.Barney (ed) *Elevator Technology* 6 *Proc. Of ELEVCON'95*, March 13-16, 1995, Hong Kong: 88-95.
4. Amos M, Paun G, Rozenberg G, Salomaa A(2002) Topics in the Theory of DNA Computing, *J. Theoretical Computer Science*, 287(1): 3-38.
5. Barney GC, dos Santos S (1985) *Elevator Traffic Analysis, Design and Control (2nd ed)*, IEEE Computer Society, USA.
6. Barney GC (2003) *Elevator Traffic Handbook: Theory and Practice-US-*, Spon Press, USA.
7. Beielstein T, Ewald T-P,Markon S (2003) Optimal elevator group control by evolution strategies, In Cantu-Paz E, Foster JA (ed) *Genetic and Evolution Computation-Gecco 2003, Proc. Of Genetic and Evolutionary Computation, Conf.* July 12-16, 2003, Chicago, Springer-Verlag, Berlin,1963-1974.

8. Bi X, Zhu C, Ye Q (2004) A GA-based approach to the multi-objective optimization problem in elevator group control system, *Elevator World*, June: 58-63.
9. Cortes P, Larraneta L, Onieva L (2004) Genetic algorithm for controllers in elevator groups: analysis and simulation during lunchpeak traffic, *Applied Soft Computing*, 4: 159-174.
10. Crites R, Barto A (1998) Elevator group control using multiple reinforcement learning agents, *Machine Learning*, 33: 235-262.
11. Eguchi T, Hirasawas K, Hu J, Markon S (2004) Elevator group supervisory control system using genetic network programming, In: Eberhart RC, Shi Y (ed) *Proc. of IEEE Congress on Evolutionary Computation 2004, CEC2004*, 19-23 June 2004, IEEE, USA: 1661-1667.
12. Eguchi T, Hirasawa K, Hu J, Markon S (2006) Elevator group supervisory control system using genetic network programming with functional localization, *J. Advanced Computational Intelligence and Intelligent Informatics*, 10(3): 385-394.
13. Fujino A, Tobita T, Segawa K, Yoneda K,Togawa A (1997) An elevator group control system with floor-attribute control method and systems optimization using genetic algorithms," *IEEE Trans. Industrial Electronics*, 44(4): 546-552.
14. Gudwin R, Gomide F and Netto M (1998) "A fuzzy elevator group controller with linear context adaptation," In: *Proc. Of Fuzzy-IEEE98, WCCI'98-IEEE World Conf. Computational Intelligence*, Anchorage, Alaska 1998: 481-486.
15. Ito Y, Fukusaki E (2004) DNA as a 'Nanomaterial', *J Molecular Catalysis B: Enzymatic* 28(4-6): 155-166.
16. Jeng D J-F, Watada J, Kim I (2007) Solving a real time scheduling problem based on DNA computing, *Soft Computing J.* (in press).
17. Kim C, Seong K, Lee-Kwang H, Kim JO (1998) Design and implementation of a fuzzy elevator group control system," *IEEE Trans. System, Man and Cybernetics - PART-A*, 28(3): 277-287.
18. Kim I, Jeng D J-F, Watada J (2006) Redesigning subgroups in a personnel network based on DNA computing *Int J. Innovative Computing, Information and Control*, 2(4): 885-896.
19. Lee JY, Zhang B-T, Park TH (2003) Effectiveness of denaturation temperature gradient-polymerase chain reaction for biased DNA algorithms, *Pre-Proc. 9th Internaional Meeting on DNA Based Computers*, Madison: 208.
20. Lee JY, Shin S-Y, Park TH, Zhang B-T (2004) Solving traveling salesman problems with DNA molecules encoding numerical values, *Biosystems* 78(1): 39-47.
21. Lipton RJ (1995) DNA solution of hard computational problems, *Science*, 268: 542-545
22. Marmur J, Doty P (1962) Determination of the base composition of deoxyribonucleic acid from its thermal denaturation temperature, *J. Molecular Biology*, 5: 109-118.
23. van Noort D (2004) Towards a re-programmable DNA computer, *DNA9, LNCS 2943*, Springer-Verlag, Berlin Heidelberg: 190-196.
24. Ouyang Q, Kaplan PD, Liu S, Libchaber A (1997) DNA solution of the maximal clique problem, *Science*, 278: 446-449.
25. Owenson GG, Amos M, Hodgson DA, Gibbsons A (2001) DNA-based logic, *Soft Computing*, 5(2): 102-105.
26. Paun GH, Rozenberg G, Salomaa A (1999) *DNA Computing: New Computing Paradigms*, Yokomori T (Translated Ed) Springer (in Japanese).

27. Powell BA, Sirag DJ, Witehall BL (2001) Artificial neural networks in elevator dispatching, In: *Lift Report* 27(2): 14-19.
28. Rohani BAB, Watada J, Pedrycz W (2006) A DNA computing approach to data clustering based on mutual distance rrder, In: Watada J (ed) *Proc. 9th Czech-Japan Seminar* August 18-22, 2006, Kitakyusyu & Nagasaki: 139-145
29. SantaLucia JJr (1998) A unified view of plymer, dumbbell, and olygonucleotide DNA nearest-neighbor thermodynamics, *Proc. National Academy of Sciences of U.S.A.*, 95: 1460-1465.
30. Wan H, Liu C, Liu H (2002) NN. Elevator Group-Control Method, In: *Elevator World* 2002(2): 148-154.
31. Watada J, Kojima S, Ueda S, Ono O (2006) DNA computing approrch to optimal decision problem, *Int. J. Innovative Computing, Information and Control*, 2(1): 273-282.
32. Wetmur JG (1991) DNA probes: applications of the principles of nucleic acid hybridization, *Critical Reviews in Biochemistry and Molecular BIology*, 26(3): 227-259.
33. Winfree E, Lin F, Wenzler LA, Seeman NC (1998) Design and self-assembly of two-dimensional DNA crystals, *Nature*, 394(6693): 539-549.