Computer Communications 25 (2002) 1536–1547

www.elsevier.com/locate/comcom

A genetic algorithm for solving dual-homing cell assignment problem of the two-level wireless ATM network

D.-R. Din[a], S.S. Tseng[b,*]

[a]Department of Computer Science and Information Management, Hung-Kuang Institute of Technology, Taichung 433, Taiwan, ROC
[b]Department of Computer and Information Science, National Chiao-Tung University, 1001 Ta Hsueh Road, Hsinchu 30050, Taiwan, ROC

Received 3 September 2001; revised 11 March 2002; accepted 27 March 2002

Abstract

In this paper, we investigate the optimal assignment problem, which assigns cells in Personal Communication Service to switches on Asynchronous Transfer Mode network in an optimum manner. The cost has two components: one is the cost of handoffs that involve two switches, and the other is the cost of cabling. This problem is model as dual-homing cell assignment problem, which is a complex integer programming problem. Since finding an optimal solution of this problem is NP-hard, a stochastic search method, based on a genetic approach, is proposed to solve this problem. In this paper, domain-dependent heuristics are encoded into crossover operations, mutations of genetic algorithm (GA) to solve this problem. Simulation results show that GA is robust for this problem. © 2002 Elsevier Science B.V. All rights reserved.

Keywords: Cell assignment; Dual-homing; Wireless ATM; Genetic algorithm; Design of algorithm

1. Introduction

Recently there has been some interest in extending Asynchronous Transfer Mode (ATM) technology to the wireless environment [1–10]. The motivation behind this extension (termed *wireless ATM*) includes the desire for seamless interconnection of wireless and ATM networks, and the need to support emerging mobile mutilated services. However, due to some inherent differences between these two types of networks, the introduction of ATM into the wireless environment presents many interesting challenges [7,10]. These include support for an end-to-end ATM connection with user mobility, handling high error rate performance of wireless links, grouping cells into LAs (location area or cluster), and assigning clusters to switches in an optimum manner.

The *cell assignment problem* in Personal Communication Service (PCS) network was proposed by Merchant and Sengupta in Ref. [7], which assigns each cell in PCS network to *only one* switch on ATM network such that the total (the sum of cabling and handoff) cost can be minimized. The problem was formulated as an integer programming problem and a heuristic algorithm was proposed to solve it. They [7] also discussed the *dual-homing cell assignment problem* in PCS network by considering that the calling patterns at different times of the day could be different. In the dual-homing cell assignment problem, each cell will be assigned to two switches and they allowed one to reduce the cost of handoff by increasing the cost of cables.

If the ATM backbone network is integrated with the PCS network, the handoff cost considered in Ref. [7] which only depends on the frequency of handoff between two switches is not realistic. Since the switch of ATM backbone is widely spread, the communication cost between two switches should be considered in calculating the handoff cost. In Refs. [11–14], the cell assignment problem considered in Ref. [1] was extended to cell assignment problem in wireless ATM environment. A solution model [12,14] consists of five three-phase heuristic algorithms (NSF, LHWF, GHWF, MLCF, and MCMLCF), two genetic algorithms (GA) [11] (SGA and EGA), and a simulated annealing algorithm [13] were proposed to solve the cell assignment problem in wireless ATM environment. Experimental results showed that these algorithms have good efficiency.

In the designing of traditional network, robustness of networks, the ability to perform required communications

* Corresponding author. Tel.: +886-3-5731966; fax: +886-3-5721490.
E-mail addresses: sstseng@cis.nctu.edu.tw (S.S. Tseng), deron@aho.cis.nctu.edu.tw (D.R. Din).

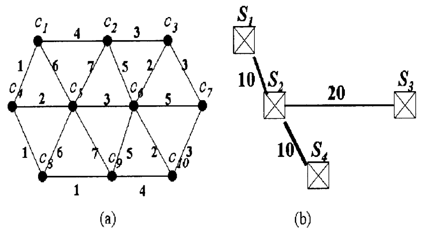

Fig. 1. An example of PCS network and ATM network, (a) PCS network $CG(C, L)$, (b) ATM network $G(S, E)$.

after a specified set of components becomes unavailable [15], may be enhanced by various technical solutions. However, it cannot be fully met without having a sufficient level of structural redundancy. The most effective and straightforward means for structural redundancy would be provided more than one path between any pair of cell and switches. There exists some network design studies in which each terminal (user) node is to be provided with redundant connections to the main hub network for backup services, this can be viewed as variants of the *facility location problem*. Prikul et al. [16] have investigated the *concentrator location problem* where each terminal should be connected to two different concentrators. Tang et al. [17] and Narasimhan [18] have dealt with the concentrator location problem where each terminal has multiple connections to several concentrators.

In this paper, we are dealing with the topological design of the two-level hierarchical network, where the upper network is the ATM backbone network, and the lower level is the PCS network. The problem of *multi-homing cell assignment problem* is assigning each cell in PCS network to more than one switches on ATM network. In such a problem, each cell c_i will be assigned to K_i ($K_i \geq 1$ and integer) switches and the corresponding K_i disjoint links will be created. If K_i is set as 1 for all cells, the problem is reduced to the cell assignment problem [11,12]. In this paper, we are concerned about the dual-homing cell assignment problem where K_i was fixed at 2 for all cells. That is, there are two assignments of each cell in PCS network, one is the *primary assignment* and the other is the *secondary assignment*.

Under normal circumstances, each cell uses its primary assignment to communicate. If that connection between cell and the corresponding switch of the primary assignment becomes unavailable, the secondary assignment is used. Such additional switches assignment (hence, reliability and availability) comes at a cost. The increased cost arises and also due to the additional switch capacity due to the additional cabling costs of connecting the cells to their assigning switches.

In this paper, we are given a group of cells in PCS network and a group of switches in an ATM network (whose locations are fixed and known). The problem is to assign cells to switches in the ATM network in an optimum manner. We consider the topological design of a two-level hierarchical network. The objective cost has two components. One is the inter-switch handoff cost that involves two different switches and the other is the cost of cabling that connects cells to switches of ATM network. We try to assign cells to switches such that the total cost can be minimized under some assumptions which will be described in Section 2.

We first present a mathematical programming formulation for the dual-homing cell assignment problem in wireless ATM network environment. Since this problem is an NP-hard problem, this implies that a solution procedure searching for the optimal solution may not terminate within a reasonable amount of computing time. In order to deal with problems of significant size, it is worthwhile developing an effective GA to solve this problem.

The organization of this paper is as follows. In Section 2, we formally define the problem. In Section 3, we describe the backgrounds of GAs. In Section 4, we describe the details of solution algorithm. The experimental results are presented in Section 5. Finally, conclusions are given in Section 6.

2. Problem formulation

This section first provides an overview of various terms and notations used to explain the concepts outlined in the subsequent sections. Let n cells in PCS network $CG(C, L)$ be assigned to m switches in ATM network $G(S, E)$. We assume that the location of cells and switches, the structure of ATM network $G(S, E)$ are fixed and known. Assume G is connected, s_k, s_l in S, and (s_k, s_l) in E. Let (X_{s_k}, Y_{s_k}) be the coordinate of switch s_k, $k = 1, 2, \ldots, m$; (X_{c_i}, Y_{c_i}) be the coordinate of cell c_i, $i = 1, 2, \ldots, n$; and d_{kl} be the minimal communication cost between the switches s_k and s_l. Let f_{ij} be the frequency of handoff per unit time that occurs between cells c_i and $c_j \in C$, $(i, j = 1, 2, \ldots, n)$ is fixed and known. We assume that all edges in CG are undirected and weighted; and assume cells c_i and c_j in C are connected by an edge $(c_i, c_j) \in L$ with weight w_{ij}, where $w_{ij} = f_{ij} + f_{ji}$, $w_{ij} = w_{ji}$, and $w_{ii} = 0$ [11,12]. Let l_{ik} be the cost of cabling per unit time and between cell c_i switch s_k, $(i = 1, 2, \ldots, n$; $k = 1, 2, \ldots, m)$ and assume l_{ik} is the function of Euclidean distance between cell c_i and switch s_k, that is, $l_{ik} = \sqrt{(X_{c_i} - X_{s_k})^2 + (Y_{c_i} - Y_{s_k})^2}$.

Assume the number of calls that can be handled by each cell per unit time is equal to 1. Let Cap_k be the number of cells that can be assigned to switch s_k. Our objective is to assign each cell in C to two switches so as to minimize (total cost) the sum of cabling cost and handoffs cost per unit time of whole system.

Example 1. Consider the graph shown in Fig. 1. There are 10 cells in C which should be assigned to *four* switches in S. The weight of edge between two cells is the frequency of

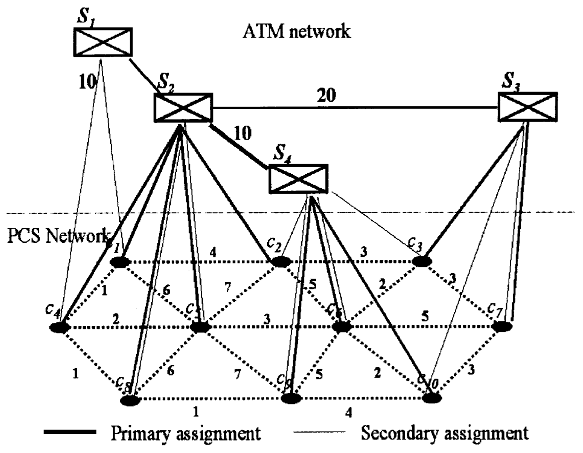

Fig. 2. A possible dual-homing assignment of Example 1.

handoffs per unit time that occurs between them. Four switches are positioned at the center of the cell: c_1, c_2, c_4, and c_6.

Assume the matrix CS of the distance between a cell and a switch is as follows:

$$CS = \{l_{ik}\}_{10\times 4} = \begin{array}{c|cccc} & s_1 & s_2 & s_3 & s_4 \\ \hline c_1 & 0 & 1 & \sqrt{7} & 2 \\ c_2 & 1 & 1 & \sqrt{3} & \sqrt{3} \\ c_3 & 2 & \sqrt{3} & 1 & 2 \\ c_4 & 1 & 1 & 3 & \sqrt{3} \\ c_5 & 1 & 0 & 2 & 1 \\ c_6 & \sqrt{3} & 1 & 1 & 1 \\ c_7 & \sqrt{7} & 2 & 0 & \sqrt{3} \\ c_8 & \sqrt{3} & 1 & \sqrt{7} & 1 \\ c_9 & 2 & 1 & \sqrt{3} & 0 \\ c_{10} & \sqrt{7} & \sqrt{3} & 1 & 1 \end{array}.$$

A possible dual-homing cell assignment of Example 1 is shown in Fig. 2.

To formulate this problem, let us define the following

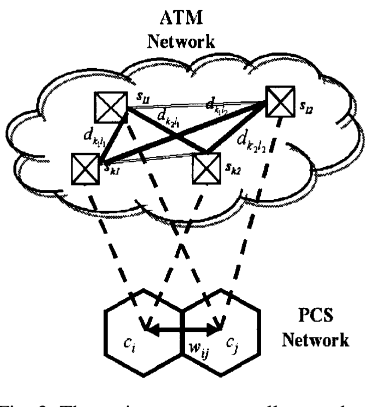

Fig. 3. The assignments two cells c_i and c_j.

variables. Let $x_{ik_1} = 1$ if the primary assignment of cell c_i is assigned to switch s_{k_1}, $s_{k_1} \in S$; $x_{ik_1} = 0$, otherwise. Let $x_{ik_2} = 1$ if the secondary assignment of cell c_i is assigned to switch s_{k_2}, $s_{k_2} \in S$; $x_{ik_2} = 0$, otherwise. It is important to note that if a cell is to be connected to the same switch in both primary and secondary assignments, its cabling cost should not be double. In order to ensure that the cabling costs are not double in the event that a cell is connected to the same switch in both assignments, variables x_{ik}, $i = 1,2,\dots,n$, $k = 1,2,\dots,m$ are defined as: $x_{ik} = x_{ik_1} \vee x_{ik_2}$, for $i = 1,2,\dots,n$ and $k = 1,2,\dots,m$, where the ' ∨ ' symbol means the 'or' operation [1]. Since each cell should be assigned to at most two switches, we have the constraints

$$\sum_{k_1=1}^{m} x_{ik_1} = 1, \quad \text{for } i = 1,2,\dots,n$$

and

$$\sum_{k_2=1}^{m} x_{ik_2} = 1, \quad \text{for } i = 1,2,\dots,n.$$

Further, since we allow the two assignments of the cell can be the same, the constraint on the call handling capacity of switch is

$$\sum_{i=1}^{n} x_{ik_1} + \sum_{i=1}^{n} x_{ik_2} \le Cap_k, \quad \text{for } k = k_1 = k_2 = 1,2,\dots,m.$$

Thus, the sum of cabling costs can be formulated as:

$$\sum_{i=1}^{n}\sum_{k=1}^{m} l_{ik}x_{ik} = \sum_{i=1}^{n}\sum_{k=1}^{m} l_{ik}(x_{ik_1} \vee x_{ik_2}).$$

To formulate the handoff cost, define variables $z_{ijk_1} = x_{ik_1}x_{jk_1}$, for $i,j = 1,2,\dots,n$ and $k_1 = 1,2,\dots,m$. Thus, z_{ijk_1} equals 1 if both the primary assignments of cells c_i and c_j are connected to a common switch s_{k_1}; otherwise it is zero. Further, let

$$y_{ij} = \sum_{k_1=1}^{m} z_{ijk_1}, \quad i,j = 1,2,\dots,n.$$

Thus, y_{ij} takes a value of 1, if both the primary assignments of cells c_i and c_j are connected to a common switch; 0, otherwise. With this definition, it is easy to see that the cost of handoffs per unit time between the primary assignment of cell c_i and c_j is given by

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k_1=1}^{m}\sum_{l_1=1}^{m} w_{ij}(1-y_{ij})x_{ik_1}x_{jl_1}d_{k_1l_1}.$$

The formulation of handoff cost described above is directly derived from the cell assignment problem in Refs. [11,12]. In the dual-homing cell assignment problem, since each cell is assigned to two switches in ATM network, the computation of handoff cost should consider more complex model. As shown in Fig. 3, assume cell c_i is assigned to s_{k_1} and s_{k_2}, c_j is assigned to switches s_{l_1} and s_{l_2}, respectively.

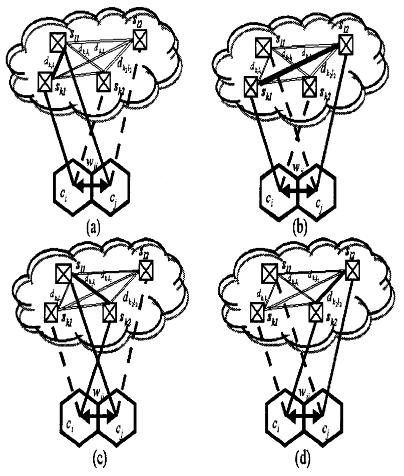
Fig. 4. Four cases of computing multi-homing handoff cost.

There are four possible communication cases between cells c_i and c_j as shown in Fig. 4. They are: (a) from s_{k_1} to s_{l_1}, (b) from s_{k_1} to s_{l_2}, (c) from s_{k_2} to s_{l_1}, and (d) from s_{k_2} to s_{l_2} as shown in Fig. 4(a)–(d), respectively. Thus, the handoff cost should be computed by the summation (or average) of these cases. To formulate the handoff cost, several variables are introduced as follows:

- $z_{ijk_1} = x_{ik_1}x_{jl_1}$, for $i,j = 1,2,\ldots,n$ and $k = k_1 = l_1 = 1,2,\ldots,m$.
- $y_{ij} = \sum_{k_1=1}^{m} z_{ijk}$, $i,j = 1,2,\ldots,n$.
- $z'_{ijk} = x_{ik_1}x_{jl_2}$, for $i,j = 1,2,\ldots,n$ and $k = k_1 = l_2 = 1,2,\ldots,m$.
- $y'_{ij} = \sum_{k=1}^{m} z'_{ijk}$, $i,j = 1,2,\ldots,n$.
- $z''_{ijk} = x_{ik_2}x_{jl_1}$, for $i,j = 1,2,\ldots,n$ and $k = k_2 = l_1 = 1,2,\ldots,m$.
- $y''_{ij} = \sum_{k=1}^{m} z''_{ijk}$, $i,j = 1,2,\ldots,n$.
- $z'''_{ijk} = x_{ik_2}x_{jl_2}$, for $i,j = 1,2,\ldots,n$ and $k = k_2 = l_2 = 1,2,\ldots,m$.
- $y'''_{ij} = \sum_{k=1}^{m} z'''_{ijk}$, $i,j = 1,2,\ldots,n$.

With these definitions, it is easy to see that the cost of handoffs per unit time for the best case is given by

$$\text{Handoff} = \alpha\left\{\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k_1=1}^{m}\sum_{l_1=1}^{m} w_{ij}(1-y_{ij})x_{ik_1}x_{jl_1}d_{k_1l_1} + \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k_1=1}^{m}\sum_{l_2=1}^{m} w_{ij}(1-y'_{ij})x_{ik_1}x_{jl_2}d_{k_1l_2} + \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k_2=1}^{m}\sum_{l_1=1}^{m} w_{ij}(1-y''_{ij})x_{ik_2}x_{jl_1}d_{k_2l_1} + \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k_1=1}^{m}\sum_{l_2=1}^{m} w_{ij}(1-y'''_{ij})x_{ik_2}x_{jl_2}d_{k_2l_2}\right\}.$$

This, together with our earlier statement about the sum of cabling costs, gives us the objective function:

$$\text{Minimize}: \sum_{i=1}^{n}\sum_{k=1}^{m} l_{ik}x_{ik} + \text{Handoff},$$

where α is the ratio of the cost between cabling and network costs. That is, our objective is to assign each cell to two switches so as to minimize (total cost) the sum of cabling costs and handoffs cost per unit time.

3. Background of genetic algorithms

The *Genetic Algorithm* was developed by John Holland at the University of Michigan [19]. GAs are search techniques for global optimization in a complex search space. As the name suggests, GA employs the concepts of natural selection and genetic. Using past information, GA directs the search with expected improved performance. The concept of GA is based on the theory of adoption in natural and artificial systems [19]. In artificial adaptive systems, adaptation starts with an initial set of structures (possible solutions). These initial structures are modified according to the performance of their solution by using an adaptive plan to improve the performance of these structures. It has been proved by Holland that repeatedly applying this adaptive plan to input structures results in optimal or near optimal solutions [19]. The traditional methods of optimization and search do not fare well over a broad spectrum of problem domains [20]. Some are limited in scope because they employ local search techniques (e.g. calculus-based methods). Others, such as enumerative schemes, are not efficient when the practical search space is too large.

3.1. Concept of GA

The search space in GA is composed of possible solutions to the problem. A solution in the search space is represented by a sequence of 0*s* and 1*s*. This solution string is referred as a chromosome in the search space. Each chromosome has an associated objective function called the *fitness*. A good chromosome is the one that has a high/low fitness value, depending upon the nature of the problem (maximization/minimization). The strength of a chromosome is represented by its *fitness value*. Fitness values indicate which chromosomes are to be carried to the next generation. A set of chromosomes and associated fitness values is called the *population*. This population at a given stage of GA is referred to as a *generation*. The general GA proceeds as follows:

```
Genetic Algorithm( )

Begin
Initialize population;
while (not terminal condition) do
    Begin
```

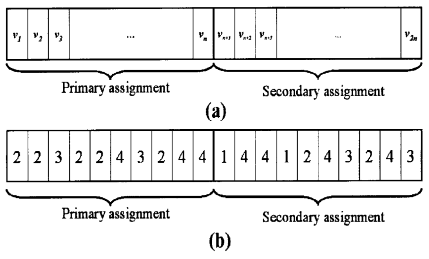

Fig. 5. (a) Cell-oriented representation of chromosome structure. (b) Cell-oriented representation of Example 1.

```
        choose parents from population; /* Selection */
        construct offspring by combining parents; /* Crossover */
        optimize (offspring); /* Mutation */
        if suited (offspring) then
        replace worst fit (population) with better offspring;
        /* Survival of the fittest */
    End;
End.
```

There are three main processes in the while loop for GA:

1. The process of selecting good strings from the current generation to be carried to the next generation. This process is called *selection/reproduction*.
2. The process of shuffling two randomly selected strings to generate new offspring is called *crossover*. Sometimes, one or more bits of a chromosome are complemented to generate a new offspring. This process of complementation is called *mutation*.
3. The process of replacing the worst performing chromosomes based on the fitness value.

The population size is finite in each generation of GA, which implies that only relatively fit chromosomes in generation (i) will be carried to the next generation ($i+1$). The power of GA comes from the fact that the algorithm terminates rapidly to an optimal or near optimal solution. The iterative process terminates when the solution reaches the optimum value. The three genetic operators, namely, selection, crossover and mutation, are discussed in Sections 3.2–3.4.

3.2. Selection/reproduction

Since the population size in each generation is limited, only a finite number of good chromosomes will be copied in the *mating pool* depending on the fitness value. Chromosomes with higher fitness values contribute more copies to the mating pool than those with lower fitness values do. This can be achieved by assigning proportionately a higher probability of copying a chromosome that has a higher fitness value [20]. Selection/reproduction uses the fitness values of the chromosome obtained after evaluating the objective function. It uses a biased roulette wheel [20] to select chromosomes, which are to be taken in the mating pool. It ensures that highly fit chromosomes (with high fitness value) will have a higher number of offspring in the mating pool. Each chromosome (i) in the current generation is allotted a roulette wheel slot sized in proportion (p_i) to its fitness value. This proportion p_i can be defined as follows. Let Of_i be the actual fitness value of a chromosome (i) in generation (j) of g chromosomes, $\mathrm{Sum}_j = \sum_{i=1}^{g} Of_i$ be the sum of the fitness values of all the chromosomes in generation j, and let $p_i = Of_i/\mathrm{Sum}_j$.

When the roulette wheel is spun, there is a greater chance that a better chromosome will be copied into the mating pool because a good chromosome occupies a larger area on the roulette wheel.

3.3. Crossover

This phase involves two steps: first, from the mating pool, two chromosomes are selected at random for mating, and second, crossover site c is selected uniformly at random in the interval $[1,n]$. Two new chromosomes, called *offspring*, are then obtained by swapping all the characters between positions $c+1$ and n. This can be shown using two chromosomes, say P and Q, each of length $n=6$ bit positions

chromosome P: 111|000;
chromosome Q: 000|111.

Let the crossover site be 3. Two substrings between 4 and 6 are swapped, and two substrings between 1 and 3 remain unchanged; then, the two offspring can be obtained as follows:

chromosome R: 111|111;
chromosome S: 000|000.

3.4. Mutation

Combining the reproduction and crossover operations may sometimes result in losing potentially useful information in the chromosome. To overcome this problem, mutation is introduced. It is implemented by complementing a bit (0 to 1 and vice versa) at random. This ensures that good chromosomes will not be permanently lost.

4. Genetic algorithm for dual-homing cell assignment problem

In this section, we discuss the details of GA developed to solve the dual-homing assignment problem of optimum assignment of cells in PCSs to switches in the ATM network. The development of GA requires: (1) a chromosomal coding scheme, (2) genetic crossover operators, (3) mutation operators, (4) fitness function and penalty function definitions, (5) a replacement strategy, and (6) termination rules.

4.1. Chromosomal coding

Since our problem involves representing connections between cells and switches, we employ a coding scheme that use positive integer numbers. Cells are labeled from 1 to n (the total number of cells), and switches are labeled from 1 to m (the total number of switches). The *cell-oriented representation* of chromosome structure shown in Fig. 5(a) consists of two parts. The first part is the primary assignment of cells, where the ith cell belongs to the v_ith switch, and the second part is the secondary assignment, where the ith cell belongs to the $v_{(i+n)}$th switch. Since each cell will be assigned to two switches on ATM network, we use $2 \times n$ array to represent the assignment of cells. If cell c_i is assigned to switches s_{k_1}, and s_{k_2}, then $v_i = k_1$ and $v_{(i+n)} = k_2$. For example, the chromosome of the Example 1 shown in Fig. 2 is shown in Fig. 5(b). It is worth noting that, the cell-oriented representation of chromosome structure can be divided into two sets which represent the primary assignment and the secondary assignment of cells, respectively.

4.2. Genetic crossover operator

Four types of genetic operators were used to develop this algorithm:

1. *partial single point crossover*,
2. *global single point crossover*,
3. *partial cell-exchanging operator*, and
4. *global cells-exchanging operator*.

4.2.1. Partial single point crossover

The partial single point crossover is randomly selecting two chromosomes (say P_1 and P_2) for crossover from previous generations and then by using a random number generator, an integer value i is generated in the range $[1, 2n]$. This number i is used as the crossover site. All characters between $i+1$ and $2n$ of two parents are swapped and offspring chromosomes O_1 and O_2 are generated. The following example provides the detailed description of partial single point crossover operation: (assume crossover site $i = 5$),

parent P_1:

2 2 3 3 3 | **4 3 2 4 4** || **1 4 4 1 2 1 4 4 1 2**,

parent P_2:

1 4 4 1 2 | **1 4 4 1 2** || **2 2 3 2 2 4 3 2 4 4**.

Two substrings between 6 and 20 are swapped, we have:

offspring O_1:

2 2 3 3 3 | **1 4 4 1 2** || **2 2 3 2 2 4 3 2 4 4**,

offspring O_2:

1 4 4 1 2 | **4 3 2 4 4** || **1 4 4 1 2 1 4 4 1 2**.

4.2.2. Global single point crossover

The global single point crossover is slightly different from the partial single point crossover, which randomly selects two chromosomes (say P_1 and P_2) for crossover from previous generations and then by using a random number generator, an integer value i is generated in the range $[1, n]$. Two numbers i and $(i+n)$ are used as the crossover sites. All characters between $i+1$ and n, $(i+n+1)$ and $2n$ of two parents are swapped and offspring chromosomes O_1 and O_2 are generated. The following example provides the detailed description of global single point crossover operation: (assume crossover sites are $i = 5$ and 15),

parent P_1:

2 2 3 3 3 | **4 3 2 4 4** || 1 4 4 1 2 | **1 4 4 1 2**,

parent P_2:

1 4 4 1 2 | **1 4 4 1 2** || 2 2 3 2 2 | **4 3 2 4 4**.

First, two substrings between 6 and 10 are swapped, then two substrings between 16 and 20 are swapped, we have:

offspring O_1:

2 2 3 3 3 | **1 4 4 1 2** || 1 4 4 1 2 | **4 3 2 4 4**,

offspring O_2:

1 4 4 1 2 | **4 3 2 4 4** || 2 2 3 2 2 | **1 4 4 1 2**.

4.2.3. Partial cell-exchanging operator

In partial cell-exchanging operator, two sites of chromosome in range $[1, 2n]$ are randomly selected, and the primary assignment of two cells are exchanged.

For example, before exchanging, we have:

2 2 **3^* 3 3 4 3 2^* 4 4** || **1 4 4 1 2 1 4 4 1 2**.

(Assume that the two cells c_3 and c_8 are selected) After exchanging:

2 2 **2^* 3 3 4 3 3^* 4 4** || **1 4 4 1 2 1 4 4 1 2**.

4.2.4. Global cell-exchanging operator

In global cell-exchanging operator, two cells in C of a chromosome $[l, n]$ (say c_i and c_j) are randomly selected, the primary and secondary assignments of two cells c_i, and c_j are exchanged. That is, v_i and v_j, $v_{(i+n)}$ and $v_{(j+n)}$ are exchanging. For example, before exchanging, we have:

$$2\ 2\ \mathbf{3}^*\ \mathbf{3\ 3\ 4\ 3\ 2}^*\ \mathbf{4\ 4} \parallel \mathbf{1\ 4\ 4}^*\ \mathbf{1\ 2\ 1\ 4\ 4}\ 1\ 2.$$

(Assume that the two cells c_3 and c_8 are selected) After exchanging:

$$2\ 2\ \mathbf{2}^*\ \mathbf{3\ 3\ 4\ 3\ 3}^*\ \mathbf{4\ 4} \parallel \mathbf{1\ 4\ 4}^*\ \mathbf{1\ 2\ 1\ 4\ 4}^*\ \mathbf{1\ 2}.$$

4.3. Mutations and heuristic mutations

Five types of mutations are used to develop this algorithm and the active probabilities of mutations are the same.

Traditional Mutation (TM). Randomly select a cell of vector v_i, where i in $[1, 2n]$ and transform it to a random number q between 1 and m. The following example provides the detailed description of traditional mutation: (assume cell c_8 is randomly selected and $q = 4$) before mutating, we have

$$2\ 2\ 3\ 3\ 3\ 4\ 3\ \mathbf{2}^*\ \mathbf{4\ 4} \parallel \mathbf{1\ 4\ 4\ 1\ 2\ 1\ 4\ 4\ 1\ 2}.$$

After mutating, we have:

$$2\ 2\ 2\ 3\ 3\ 4\ 3\ \mathbf{4}^*\ \mathbf{4\ 4} \parallel \mathbf{1\ 4\ 4\ 1\ 2\ 1\ 4\ 4\ 1\ 2}.$$

Multiple Cells Mutation (MCM). Randomly select two random numbers k, l between 1 and m, transform the value of cells value is k to l and l to k. The following example provides the detailed description of multiple cells mutation: (assume random number $k = 3$ and $l = 2$) before mutating, we have

$$2^+\ 2^+\ 3^*\ 3^*\ 3^*\ 4\ 3^*\ 2^+\ 4\ 4 \parallel 1\ 4\ 4\ 1\ 2^+\ 1\ 4\ 4\ 1\ 2^+.$$

After mutating, we have

$$3^*\ 3^*\ 2^+\ 2^+\ 2^+\ 4\ 2^+\ 3^*\ 4\ 4 \parallel 1\ 4\ 4\ 1\ 3^*\ 1\ 4\ 4\ 1\ 3^*.$$

Heaviest Weight First Preference (HWFP) [11]. Since the handoff cost involving only one switch is negligible, two cells can be assigned to the same switch so as to reduce the handoff cost between these cells. Two cells with higher weight w_{ij} should have a higher probability of being assigned to the same switch. Thus, if we consider two connected cells c_i and $c_j \in C$, then the probability of mutation from v_i of cell c_i to the value v_j of cell c_j is as follows

$$P_{(i,j)} = \frac{w_{ij}}{\sum_{i=1}^{n} \sum_{j=1}^{\mathrm{degree}(c_i)} w_{ij}}, \quad \text{for } i,j = 1, 2, \ldots, n,$$

where degree(c_i) is the number of cells connected to cell c_i in CG.

Minimal Cabling Cost First Preference (MCCFP) [11]. To reduce the cabling costs between cells and switches, we prefer to assign each cell to the nearer switch rather than the farther one. Cell c_i and switch s_k with lower cabling cost l_{ik} should result in higher probability that c_i will be assigned to s_k. Thus, if we consider the randomly selected cell c_i, then the probability of mutation from v_i of cell c_i to the value v_k is

$$P_{(i,k)} = \frac{L_{\max} - l_{ik}}{\sum_{l=1}^{m} (L_{\max} - l_{il})},$$

where $L_{\max} = \max_{l=1}^{m} \{l_{il}\}$.

Unique Switch First Mutation (USFM). It is important to note that if a cell is to be connected to the same switch in both primary and secondary assignments, its cabling cost should not be double. That is, if the two assignments of the cell are assigned to the same switch, the cabling cost will be reduced. If we randomly selected cell c_i, then the value of v_i can be mutated to $v_{(i+n)}$ or $v_{(i+n)}$ can be mutated to v_i with equal probabilities. The following example provides the detailed description of unique switch first mutation: (assume cell c_8 is randomly selected) before mutating, we have

$$2\ 2\ 3\ 3\ 3\ 4\ 3\ \mathbf{2}^*\ \mathbf{4\ 4} \parallel \mathbf{1\ 4\ 4\ 1\ 2\ 1\ 4\ 4}^*\ \mathbf{1\ 2}.$$

After mutating, assume the primary assignment is mutated to the secondary assignment, we have

$$2\ 2\ 2\ 3\ 3\ 4\ 3\ \mathbf{4}^*\ \mathbf{4\ 4} \parallel \mathbf{1\ 4\ 4\ 1\ 2\ 1\ 4\ 4}^*\ \mathbf{1\ 2}.$$

4.4. Fitness function definition

Generally, GAs use fitness functions to map objectives to costs to achieve the goal of an optimally designed two-level wireless ATM network. If cell c_i is assigned to switch s_{k_1}, and s_{k_2}, then v_i in the chromosome is set as k_1 and $v_{(i+n)}$ is set as k_2. Let $d_{(v_i,v_j)}$, $d_{(v_i,v_{(j+n)})}$, $d_{(v_{(i+n)},v_j)}$, and $d_{(v_{(i+n)},v_{(j+n)})}$ be the minimal communication cost between switches s_{k_l} and s_{l_1}, s_{k_1} and s_{l_2}, s_{k_2} and s_{l_1}, s_{k_2} and s_{l_2} in G, respectively. An objective function value is associated with each chromosome, which is the same as the fitness measure. If $v_i = v_{(i+n)}$ then $q_i = 0$, otherwise $q_i = 1$. We use the following objective function: minimize

$$\mathrm{OBJ} = \sum_{i=1}^{n} [l_i v_i + q_i l_i v_{(i+n)}] + \alpha \sum_{i=1}^{n} \sum_{k=1}^{m} w_{ij} \{ d_{(v_i,v_j)} + d_{(v_i,v_{(j+n)})} + d_{(v_{(i+n)},v_j)} + d_{(v_{(i+n)},v_{(j+n)})} \}.$$

Note that if assumptions $\sum_{k_1=1}^{m} x_{ik_1} = 1$, $\sum_{k_2=1}^{m} x_{ik_2} = 1$, for $i = 1, 2, \ldots, n$, and $\sum_{k_1=1}^{m} x_{ik_1} + \sum_{k_2=1}^{m} x_{ik_2} = 1 \le Cap_k$, $k = 1, 2, \ldots, m$ are considered with this objective function, we have a complete problem formulation. While breeding chromosomes, the GA does not require the chromosome to reflect a feasible solution. Thus, we need to attach a penalty to the fitness function in the event the solution is infeasible. Let n_k be the number of cells assigned to switch s_k, define $p_k = n_k - Cap_k$ if $n_k \ge Cap_k$; $p_k = 0$, otherwise. We

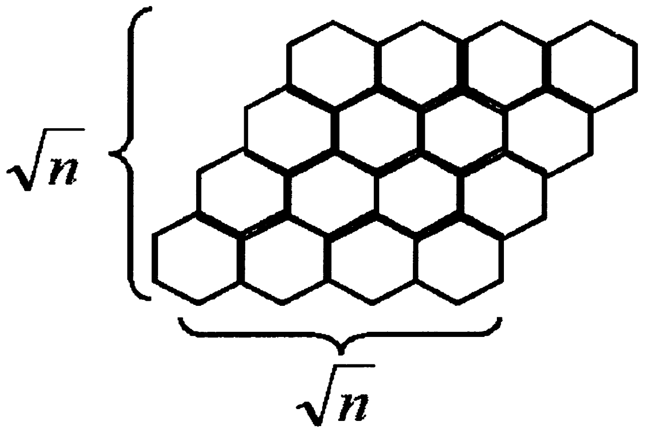

Fig. 6. H-mesh of the experiments.

rewrite the formulation above in an unconstrained form:

minimize cost $=$ OBJ $+ \Pi$,

where $\Pi = \beta(\sum_{k_1=1}^{m} p_k)$ is the penalty measure associated with a chromosome, and, β is the penalty weight. Since the best-fit chromosomes should have a probability of being selected as parents that is proportional to their fitness, they need to be expressed in a maximization form. This is done by subtracting the objective from a large number $C_{\max}$. Hence, the fitness function becomes

maximize $C_{\max} - [\mathrm{OBJ} + \Pi]$,

where $C_{\max}$ denotes the maximum value observed, so far, of the cost function in the population. Let cost be the value of the cost function for the chromosome; $C_{\max}$ can be calculated by the following iterative equation

$C_{\max} = \max\{C_{\max}, \mathrm{cost}\}$,

where $C_{\max}$ is initialized to zero.

4.5. Replacement strategy

This subsection discusses a method used to create a new generation after crossover and mutation is carried out on the chromosomes of the previous generation. Several replacement strategies have been proposed in the literature, and a good discussion can be found in Ref. [20]. The algorithm developed here combines to both the concepts maintained above. Each offspring generated after crossover is added to the new generation if it has a better objective function value than both of its parents. If the objective function value of an offspring is better than that of only one of the parents, then we select a chromosome randomly from the better parent and the offspring. If the offspring is worse than both parents, then each of the parents is selected at random for the next generation. This ensures that the best chromosome is carried to the next generation, while the worst is not carried to the succeeding generations.

Table 1
Test sets of the wireless ATM network

Set No.	No. of cells (n)	No. of switches	Cap	α	β
1	16	8	8	1.0	100 000
2	100	20	20	0.8	100 000

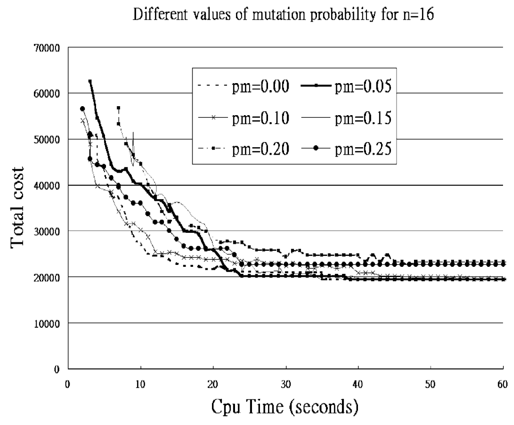

Fig. 7. Experimental results of different values of mutation probability for Set 1 ($n = 16$).

4.6. Termination rules

Execution of GA can be terminated using any one of the following rules:

R1: when the average and maximum fitness values exceed a predetermined threshold;
R2: when the average and maximum fitness values of strings in a generation become the same; or
R3: when the number of generations exceeds an upper bound specified by the user.

The best value for a given problem can be obtained from a GA when the algorithm is terminated using R2 [20].

5. Experimental results

In this section, we present the experimental results of the GA. The description of the experimental results is divided into four subsections. First, in Section 5.1, we describe that the effect of the mutation probability. In Section 5.2, we describe the effect of the crossover probability. In Section 5.3, we describe the effect of population size. Finally in Section 5.4, the effect of the heuristic mutations is examined.

In all the experiments, the implementation language is C, and all experiments were run on Windows NT with a Pentium II 450 MHZ CPU and 256MB RAM. We simulated a hexagonal system in which cells were configured as an H-mesh as shown in Fig. 6. We assumed that the antenna for each cell was at the center of the cells, and was also assumed to be at the center of the cells. Switches are located at the same position of cells which are randomly selected from the

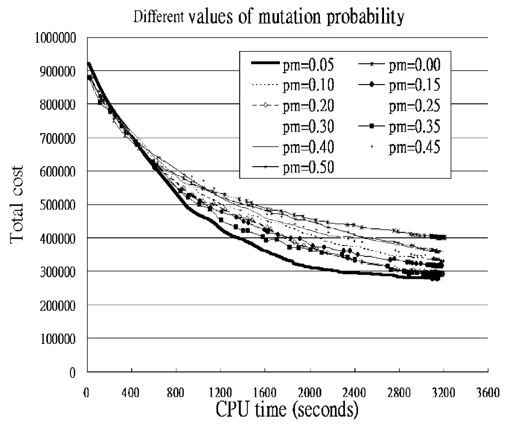

Fig. 8. Experimental results of different values of mutation probability for Set 2 ($n = 100$).

cells. The cabling cost between a switch and a cell was taken to be proportional to the geometric distance between the two. The topology of the backbone network is assumed to be a complete graph and the communication cost between two switches is assumed to be proportional to the geometric distance. The handoff frequency f_{ij} for each border was generated from a normal random number with mean 100 and variance 20. Two different set of cells and switches shown in Table 1 are used to test the performance of the algorithm.

5.1. Effect of mutation probability

To examine the effect of the mutation probability of GAs, we set population size (popsize) = 100, crossover probability (P_c) = 1.0, maximum number of generations = 3000, and mutation probability is selected from {0.00, 0.05, 0.10, 0.15, 0.20, 0.25, 0.2, 0.25}. The results shown in Figs. 7 and 8 are the best solution of different values of mutation probability after 10 runs of Set 1 and Set 2, respectively. In Fig. 7, we found that when the mutation probability is lower than (0.15), GA can find the best solution. If the mutation probability is higher (0.20 or 0.25), the GA may get trapped in local minima. In Fig. 8, for

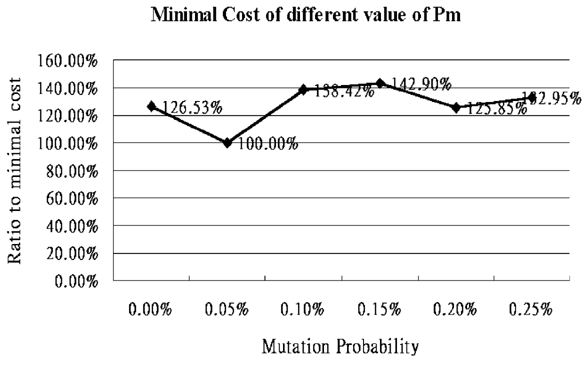

Fig. 9. The total cost ratio of different values of mutation probability for Set 2 ($n = 100$).

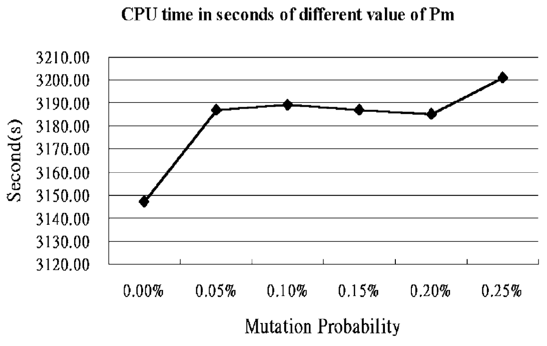

Fig. 10. The CPU time ratio of different values of mutation probability for Set 2 ($n = 100$).

the Set 2, we found that when the mutation probability is (0.05), GA can find the best solution.

Let C_{pm} be the average result of the GA running in mutation probability P_m in 10 runs, and let the known best solution total cost be the OPT. The total cost ratio computed by C_{pm}/OPT% of Set 2 is shown in Fig. 9. The CPU time of different values of mutation probability for Set 2 is shown in Fig. 10. From Figs. 9 and 10, we can conclude, for the Set 2, if P_m is set as 0.05, GA have the best result and efficiency.

5.2. Effect of crossover probability

To examine the effect of the crossover probability of GAs, we set population size (popsize) = 100, mutation probability (P_m) is 0.05, maximum number of generations = 3000, and crossover probability is selected from {1.00, 0.95, 0.90, 0.85, 0.80, 0.70, 0.60, 0.50}. The results shown in Figs. 11 and 12 are the best solutions of different values of crossover probability after 10 runs of Set 1 and Set 2, respectively. In Fig. 11, we found that when the crossover probability is greater than (0.85), GA can find the best solution. If the crossover probability is lower (0.50), the GA may get trapped in local minima. In Fig. 12, we

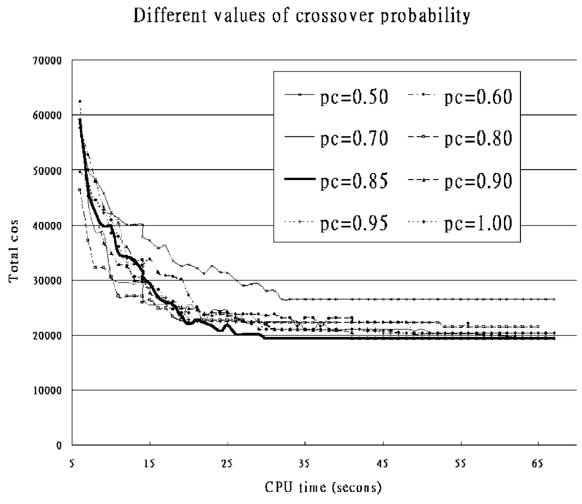

Fig. 11. Experimental results of different values of crossover probability for Set 1 ($n = 16$).

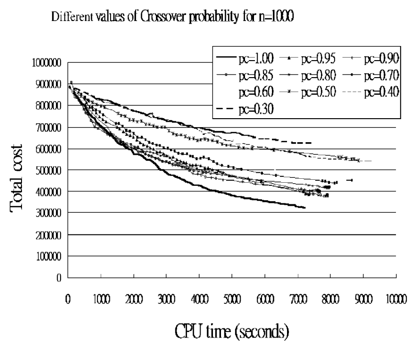

Fig. 12. Experimental results of different values of crossover probability for Set 2 ($n = 100$).

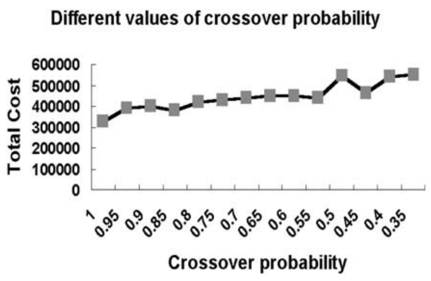

Fig. 13. The total cost ratio of different values of crossover probability for Set 2 ($n = 100$).

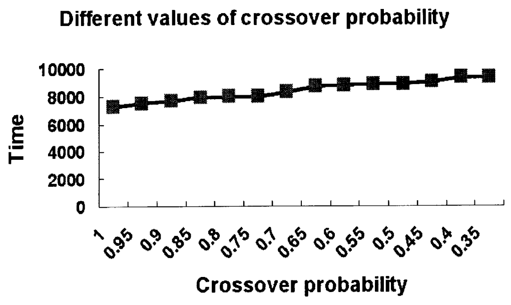

Fig. 14. The CPU time ratio of different values of crossover probability for Set 2 ($n = 100$).

found that when the crossover probability is (1.00), GA can find the best solution.

The comparison of different values of crossover probability for Set 2 is shown in Fig. 13. The CPU time of different values of crossover probability for Set 2 is shown in Fig. 14. We found that when $P_c = 1.0$, GA can find the best solution. From Figs. 13 and 14, we can conclude, for the Set 2, if P_c is set as 1.00, GA have the best result and efficiency.

5.3. Effect of population size

To examine the effect of population size of GAs, we set crossover probability (P_c) is 1.00, mutation probability (P_m) is 0.05, maximum number of generations = 3000, and population size is selected from {100, 200, 300, 400, 500,

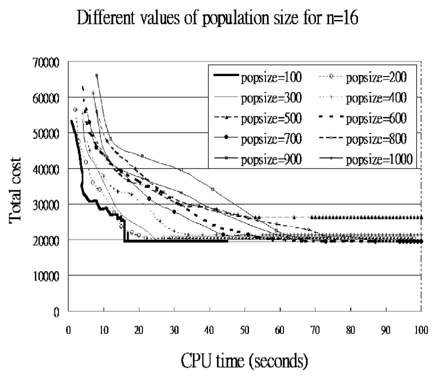

Fig. 15. Experimental results of different values of population size for Set 1 ($n = 16$).

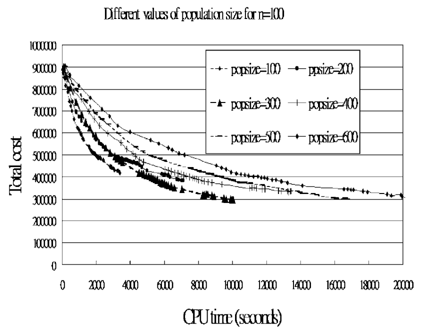

Fig. 16. Experimental results of different values of population size for Set 2 ($n = 100$).

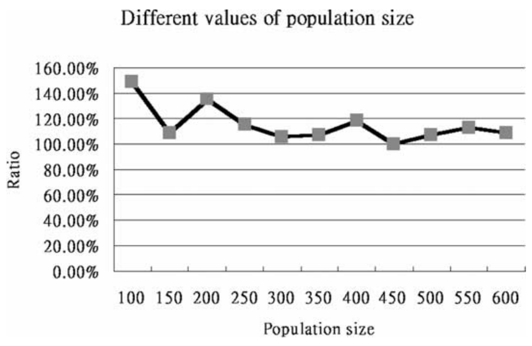

Fig. 17. The total cost ratio of different values of population size for Set 2 ($n = 100$).

600, 700, 800, 900, 1000 }. The result shown in Figs. 15 and 16 are the best solution of different values of population size after 10 runs of Set 1 and Set 2, respectively. In Fig. 15, we found when the population size is greater than 100 (except 500), GA can find the best solution. In Fig. 15, for Set 2, we found that when the population size is (300), GA can find the best solution.

The comparison of different values of population size for Set 2 is shown in Fig. 17. The CPU time of different values of population size for Set 2 is shown in Fig. 18. We found that when popsize = 300, GA can find the best solution. From Figs. 17and 18, we can conclude, for the Set 2, if P_c is set as 300, GA have the best result and efficiency.

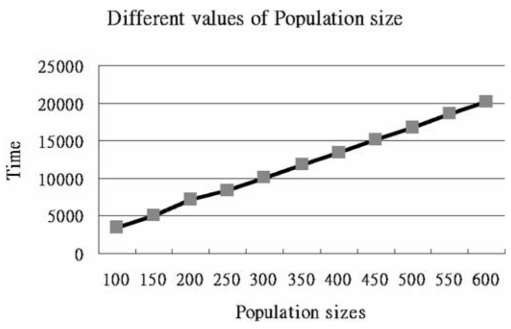

Fig. 18. The CPU time ratio of different values of population size for Set 2 ($n = 100$).

5.4. Effect of heuristic mutation

To evaluate the effect of the heuristic mutations described in Section 4.3, we simply constructed a trivial GA called TGA (trivial GA). The crossover operations of TGA were the same as our genetic algorithm (OGA) but the mutation operations were not. The mutation operation used in TGA is only the traditional mutation (TM) described in Section 4.3. The network Set 2 shown in Table 1 was used in experiments and the parameter of two GAs are set as $P_m = 0.05$, $P_c = 1.0$, and popsize = 300. Fig. 19 shows the best results in 10 runs of two GA (OGA and TGA). Observe the results shown in Fig. 19, the total cost of our GA decreases rapidly and is better than TGA. That is, heuristic mutations in OGA can decrease the total cost and keep the cost from getting trapped a local minima.

6. Conclusions

In this paper, we investigate the dual-homing cell assignment problem which optimally assigns each cell in PCS to two switches on ATM network. This problem is currently faced by designers of mobile communication service and in the future, it is likely to be faced by designers of PCS. Since finding an optimal solution of this problem is NP-hard, a stochastic search method based on a genetic approach is proposed to solve it. Simulation results showed that GA is robust for this problem. In our methods, cell-oriented representation is used to represent the cell assignment; three general genetic operators, selection, crossover, and mutation were employed. Four types of operators (partial single point crossover, global single point crossover, partial cell-exchanging crossover and global cell-exchanging crossover) and five types of mutations (TM, MCM, LHWFP, MCCFP, and USFM) are employed in our method. Experimental results indicate that the algorithm can run efficiently.

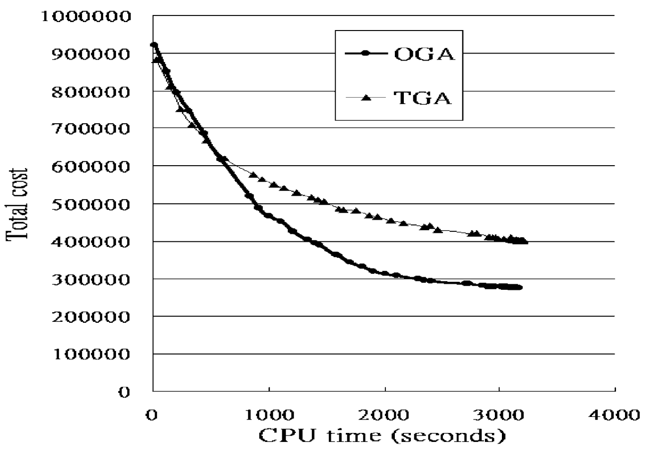

Fig. 19. Comparison of OGA and TGA Set 2 ($n = 100$).

Acknowledgments

This work was supported in part by MOE program of Excellence Research under Grant 90-E-FA04-1-4.

References

[1] M. Cheng, S. Rajagopalan, L.F. Chang, G.P. Pollini, M. Barton, PCS mobility support over fixed ATM networks, IEEE Commun. Mag. 35 (11) (1997) 82–91.
[2] D.C. Cox, Personal communication—a viewpoint, IEEE Commun. Mag. 11 (8) (1990) 8–20.
[3] T. Hattori, Personal communication—concept and architecture, International Conference on Communications (ICC'90), Georgia World Congress Center, Altanta, Georgia, April, 1990, pp. 333.4.1–335.4.7.
[4] R. Jain, Y.B. Lin, C. Lo, S. Mohan, A caching strategy to reduce network impacts of PCS, IEEE JSAC 12 (8) (1994) 1434–1444.
[5] Y.B. Lin, Determining the user locations for personal communications services networks, IEEE Trans. Veh. Technol. 43 (3) (1994) 466–473.
[6] Y.B. Lin, S.Y. Hwang, Comparing the PCS location tracking strategies, IEEE Trans. Veh. Technol. 45 (2) (1996) 114–121.
[7] A. Merchant, B. Sengupta, Assignment of cells to switches in PCS networks, IEEE/ACM Trans. Network. 3 (5) (1995) 521–526.
[8] D. Raychaudhuri, N. Wilson, ATM based transport architecture for multiservices wireless personal communication network, IEEE JSAC 12 (8) (1994) 1401–1403.
[9] R. Steele, Deploying personal communication networks, IEEE Commun. Mag. 4 (1990) 6–11.
[10] U. Varshney, Supporting mobile using wireless ATM, IEEE Comput. 30 (1) (1997) 131–133.
[11] D.-R. Din, S.S. Tseng, Genetic algorithms for optimal design of two-level wireless ATM network, Proc. Natl Sci. Council, ROC Part A: Phys. Sci. Engng 25 (3) (2001) 151–162.
[12] D.-R. Din, S.S. Tseng, Heuristic algorithm for optimal design of two-level wireless ATM network, J. Information Sci. Engng 17 (2001) 647–665.
[13] D.-R. Din, S.S. Tseng, Simulated annealing algorithm for optimal design of two-level wireless ATM network, Journal of Engineering Intelligent Systems. Submitted for publication.
[14] D.-R. Din, S.S. Tseng, A solution model for optimal design of two-level wireless ATM network, International Computer Symposium 2000 (ICS 2000) Chia-Yia, Taiwan R.O.C. Dec. 6–8, 2000 pp 120–128. Submitted for publication.

[15] G.G. Brush, N.A. Malrow, Assuring the dependability telecommunications networks and services, IEEE Network Mag. 4 (1) (1990) 29–34.
[16] H. Prikul, S. Narasimhan, P. De, Locating concentrators for primary and secondary coverage in a computer communication network, IEEE Trans. Commun. COM-36 (1988) 450–458.
[17] D.T. Tang, L.S. Woo, L.R. Bahl, Optimization of teleprocessing networks with concentrators and multiconnected terminals, IEEE Trans. Comput. c-27 (1978) 594–604.
[18] S. Narasimhan, The concentrator location problem with variable coverage, Comput. Networks ISDN Syst. 19 (1990) 1–10.
[19] J.T. Richardson, M.R. Palmer, G.L. Liepins, M. Hilliard, Some guidelines for genetic algorithms with penalty functions, in: J.D. Schaffer (Ed.), Proceedings of the Third International Conference on Genetic Algorithms, George Mason University, 1989.
[20] L. Davis (Eds.), Handbook of Genetic Algorithms, Van Nostrand Reinhold, New York, NY, 1991.