

Heuristic and simulated annealing algorithms for solving extended cell assignment problem in wireless ATM networks

Der-Rong Din¹ and S. S. Tseng^{2,*†}

¹*Department of Computer Science and Information Management, Hung-Kuang Institute of Technology,
Taichung 433, Taiwan R.O.C.*

²*Department of Computer and Information Science, National Chiao-Tung University,
Hsinchu 300, Taiwan R.O.C.*

SUMMARY

In this paper, we investigate the *extended cell assignment problem* which optimally assigns new adding and splitting cells in Personal Communication Service (PCS) to switches in a *wireless* Asynchronous Transfer Mode (ATM) network. Given cells in a PCS network and switches on an ATM network (whose locations are fixed and known), we would like to do the assignment in an attempt to minimize a cost criterion. 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 modeled as a complex integer programming problem, and finding an optimal solution to this problem is *NP-hard*. A heuristic algorithm and a simulated annealing algorithm are proposed to solve this problem. The heuristic algorithm, Extended Assignment Algorithm (EAA), consists of two phases, *initial assigning phase* and *cell exchanging phase*. First, in the initial assigning phase, the initial assignments of cells to switches are found. Then, these assignments are improved by performing cell exchanging phase in which two cells are repeatedly exchanged in different switches with great reduction of the total cost. The simulated annealing algorithm, ESA (enhanced simulated annealing), generates constraint-satisfied configurations, and uses three configuration perturbation schemes to change current configuration to a new one. Experimental results indicate that EAA and ESA algorithms have good performances. Copyright © 2002 John Wiley & Sons, Ltd.

KEY WORDS: wireless ATM; PCS; optimization; heuristic algorithm; simulated annealing; cell assignment problem

1. INTRODUCTION

The rapid worldwide growth of digital wireless communication services motivates a new generation of mobile networks to serve as infrastructure for such services. Mobile networks deployed in the next few years should be capable of smooth migration to future broadband

*Correspondence to: S. S. Tseng, Department of Computer and Information Science, National Chiao-Tung University, Hsinchu 300, Taiwan R.O.C.

†E-mail: sstseng@cis.nctu.edu.tw

Contract/grant sponsor: MOE Program of Excellence Research; contract/grant number: 90-E-FA04-1-4

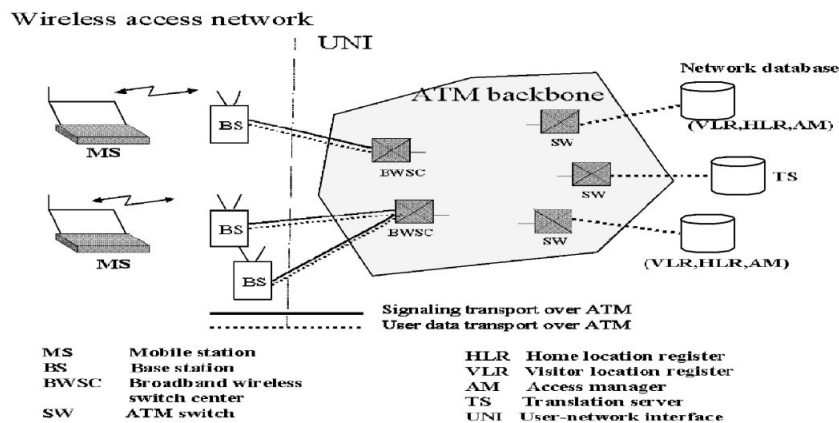


Figure 1. Architecture of wireless ATM PCS.

services based on high-speed wireless access technologies, such as *wireless asynchronous transfer mode (wireless ATM)* [1]. In the architecture presented in Reference [1] (as shown in Figure 1), the base station controllers (BSCs) in traditional personal communication service (PCS) network are omitted, and the base stations (BSs or cells) are directly connected to the ATM switches. The mobility functions supported by the BSCs will be moved to the BSs and/or the ATM switches. In this paper, we address the problem that is currently faced by designers of mobile communication service and in the future, it is likely to be faced by designers of PCS.

In the designing process of PCS network systems, first, the telephone company determined the global service area according to the usage of the mobile users, and divided the global service area into some smaller coverage areas which are covered by cells. Second, the cellular system and base stations are established and setup, BSs are connected to the switches on the ATM network to form the topology of wireless ATM. This topology may be out of date, since more and more users may use the PCS communication system. Some areas, which have not been covered in the original global service area, may now have mobile users to serve. The services requirement of some areas, which were originally covered by some BSs may be increased and exceed the capacities provided by the original BSs and switches. Nonetheless, the wireless ATM system must be extended so that the system can provide higher quantity of services to the mobile users. Two methods can be used to extend the capacities of the system and provide higher quantity of services. The first one is: adding new cells to the wireless ATM network so that the non-covered areas can be covered by new cells. The other is: reducing the size of the cells so that the total number of channels available per unit cell and the capacity of a system can be increased. In practice, this can be achieved by using *cell splitting* [2] process. The cell splitting process establishes new BSs at specific points in the cellular pattern and reduces the cell size by a factor of 2 (or more) as shown in Figure 2.

In this paper, we are given a two-level wireless ATM network as shown in Figure 3. In the PCS network of Figure 3, cells are divided into two sets. One is the set of cells, which are built originally, and each cell in this set has been assigned to a switch on the ATM network (e.g. cells c_1 , c_2 are assigned to switch s_2 , and cells c_3 , c_4 , and c_5 are assigned to switch s_4 in Figure 3). The other is the set of cells, which are newly added (e.g. c_6, c_7, c_8) or established by performing the cell splitting process (e.g. $c_9, c_{10}, c_{11}, c_{12}, c_{13}$, and c_{14}). Moreover, the locations of cells and

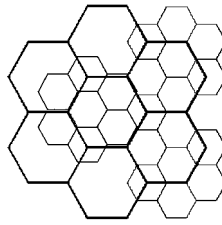


Figure 2. Cell splitting.

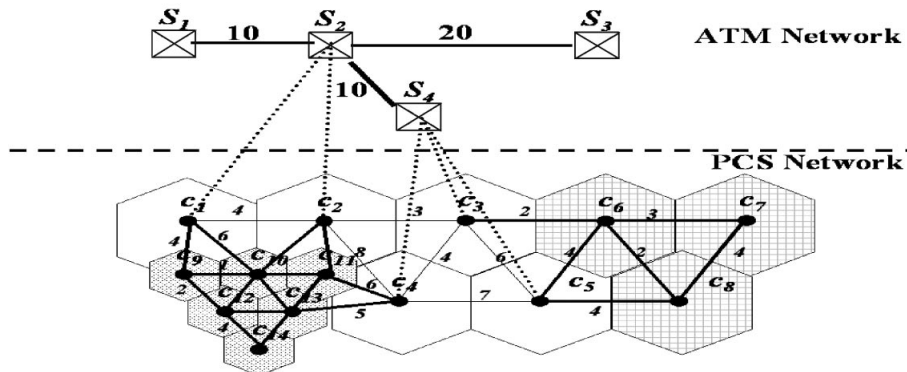


Figure 3. Two-level wireless ATM network.

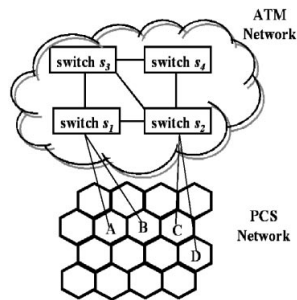


Figure 4. Handoff from B to C is more expensive than from B to A.

switches are fixed and known. To simplify the discussion, we assumed that the number of cells and switches are fixed. The problem is to assign new adding and splitting cells in the PCS to switches on the ATM network in an optimum manner. We would like to do the assignment in an attempt to minimize a cost criterion. The cost has two components: one is the cost of handoffs that involve two switches, and the other is the cost of *cabling* (or *trucking*) [3–5].

Consider the example shown in Figure 4, cells *A* and *B* are connected to switch s_1 , and cells *C* and *D* are connected to switch s_2 . If the subscriber moves from cell *B* to cell *A*, switch s_1 will perform a handoff for this call. This handoff is relatively simple and does not involve any

location update in the databases that record the position of the subscriber. The handoff also does not involve any network entity other than switch s_1 . Now let us suppose that the subscriber moves from cell B to cell C . Then the handoff involves the execution of a fairly complicated protocol between switches s_1 and s_2 . In addition, the location of the subscriber in the databases must be updated. There are two types of handoffs: one involves only one switch and the other involves two switches. The handoffs that occur between two cells connected to different switches consume much more network resources (therefore, are much more costly) than those between cells connected to the same switch [3–5]. Based on the discussion given in the previous paragraph, we assume that the cost of handoffs involving only one switch is negligible. Throughout this paper, we assume each cell to be connected to only one switch.

Merchant and Sengupta [5] considered the *cell assignment problem* which assigns cells to switches in PCS network. They formulated the problem and proposed a heuristic algorithm to solve it so that the total cost can be minimized. The total cost consists of cabling and location update. The location update cost considered in Reference [5], which depends only on the frequency of handoff between two switches, is not practical. Since the switch of the ATM backbone is wide spread, the communication cost between two switches should be considered in calculating the location update cost. In References [3,4], this model was extended to solve the problem that grouped cells into clusters and assigned these clusters to switches on the ATM network in an optimum manner by considering the communication cost between two switches. The problem considered in References [3,4] assume that the connections between cells and switches have not yet been constructed. A three-phase heuristic algorithm and two genetic algorithms are proposed to solve the *cell assignment problem* in wireless ATM network, respectively. In this paper, we follow the objective function, which was formulated in References [3,4], and assume that the original set of cells in PCS have been assigned to switches on ATM network. The goal is to find an assignment of the new adding and the splitting cells to the switches on the ATM network so that the total cost can be minimized. This problem is defined as *extended cell assignment problem*.

The organization of this paper is as follows. In Section 2, we formally define the problem. In Sections 3 and 4, we describe details of the heuristic and the simulated annealing algorithms. The experimental results are presented in Section 5. Finally, a conclusion is given in Section 6.

2. PROBLEM FORMULATION

Let $CG(C, L)$ be the PCS network, where C is a finite set of cells with $|C| = n$ and L is the set of edges such that $L \subseteq C \times C$. We assume that $C^{\text{new}} \cup C^{\text{old}} = C$, $C^{\text{new}} \cap C^{\text{old}} = \emptyset$, C^{new} is the set of new adding and splitting cells where $|C^{\text{new}}| = n'$, cells in C^{new} have not yet been assigned to switches on the ATM, and C^{old} the set of original cells where $|C^{\text{old}}| = n - n'$. Without loss of generality, we assume that cells in C^{old} and C^{new} are indexed from 1 to $n - n'$ and $n - n' + 1$ to n , respectively. If cells c_i and c_j in C are assigned to different switches, then a handoff cost is incurred. Let f_{ij} be the frequency of handoff per unit time that occurs between cells c_i and c_j , ($i, j = 1, \dots, n$) and 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 $w_{ii} = 0$ [3,4]. Let $G(S, E)$ be the ATM network, where S is the set of switches with $|S| = m$, $E \subseteq S \times S$ is the set of edges, s_k, s_l in S and (s_k, s_l) in E and G is

connected. We assume that the locations of cells and switches are fixed and known. The topology of the ATM network $G(S, E)$ is also fixed and known. Let (X_{s_k}, Y_{s_k}) be the co-ordinate of switch s_k , $k = 1, 2, \dots, m$, (X_{c_i}, Y_{c_i}) be the co-ordinate of cell c_i , $i = 1, 2, \dots, n$; and d_{kl} be the minimal communication cost between the switches s_k and s_l . Let l_{ik} be the cost of cabling per unit time and between cell c_i switch s_k , ($i = 1, \dots, n; k = 1, \dots, m$) and assume l_{ik} is the function of Euclidean distance between cell c_i and switch s_k , that is,

$$l_{ik} = \sqrt{(X_{c_i} - X_{s_k})^2 + (Y_{c_i} - Y_{s_k})^2} \quad (1)$$

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 the remaining cells that can be used to assign cells to switch s_k . Our objective is to assign cells in C^{new} to switches so that the total cost (sum of cabling cost and handoffs cost) per unit time of whole system can be minimized.

To formulate this problem, let us define the following variables. Let $x_{ik} = 1$ if cell $c_i \in C$ is assigned to switch s_k ; $x_{ik} = 0$, otherwise. Since each cell should be assigned to only one switch, we have the constraint $\sum_{k=1}^m x_{ik} = 1$, for $i = 1, \dots, n$. Further, the constraint on the capacity is

$$\sum_{i=n-n'+1}^n x_{ik} \leq \text{Cap}_k, \quad k = 1, \dots, m \quad (2)$$

Also, the sum of cabling costs is

$$\sum_{i=1}^n \sum_{k=1}^m l_{ik} x_{ik} \quad (3)$$

To formulate handoff cost, variables $z_{ijk} = x_{ik} x_{jk}$, for $i, j = 1, \dots, n$ and $k = 1, \dots, m$ are defined in Reference [5]. Thus, z_{ijk} equals 1 if both cells c_i and c_j are connected to a common switch k ; otherwise it is zero. Further, let

$$y_{ij} = \sum_{k=1}^m z_{ijk}, \quad i, j = 1, \dots, n \quad (4)$$

Thus, y_{ij} takes a value of 1 if both cells c_i and c_j are connected to a common switch and 0 otherwise. With this definition, it is easy to see that the cost of handoffs per unit time is given by Der-Rong Din and Tseng [3,4]

$$\sum_{i=1}^n \sum_{j=1}^n \sum_{k=1}^m \sum_{l=1}^m w_{ij} (1 - y_{ij}) x_{ik} x_{jl} d_{kl} \quad (5)$$

This, together with our earlier statement about the sum of cabling costs, gives us the objective function [3,4]:

$$\text{minimize : } \sum_{i=1}^n \sum_{k=1}^m l_{ik} x_{ik} + \alpha \sum_{i=1}^n \sum_{j=1}^n \sum_{k=1}^m \sum_{l=1}^m w_{ij} (1 - y_{ij}) x_{ik} x_{jl} d_{kl} \quad (6)$$

where α is the ratio of the cost between cabling and handoff costs.

The following assumptions will be satisfied:

(1) We assume that the number of cells in C^{new} is less or equal to $\sum_{k=1}^m \text{Cap}_k$. That is, there is no need for adding new switches into the ATM network.

- (2) The structures and locations of the ATM network and the PCS network are fixed and known.
- (3) Each cell in the PCS network will be directly assigned and connected to only one switch in ATM network.
- (4) To simplify the discussion, we assume that $\text{Cap}_k > 0$, for $k = 1, \dots, m$.

Example 1

Consider the two graphs shown in Figure 3. There are 14 cells in CG which should be assigned to four switches in S . In CG , cells are divided into two sets, one is the set C^{old} of cells which are built originally, and cells in C^{old} have been assigned to switches in the ATM network (e.g. $\{c_1, c_2, c_3, c_4, c_5\}$ in Figure 3). The other is the set C^{new} of cells which are new adding cells (e.g. $\{c_6, c_7, c_8\}$) or splitting cells (e.g. $\{c_9, c_{10}, c_{11}, c_{12}, c_{13}, c_{14}\}$). The edge weight between two cells is the frequency of handoffs per unit time that occurs between them. Four switches are positioned at the centre of cells c_1 , c_2 , c_4 , and c_6 . Assume that the matrix CS of the distance between cells and switches is

$$CS = \{l_{ik}\}_{14 \times 4} = \begin{matrix} & \begin{matrix} s_1 & s_2 & s_3 & s_4 \end{matrix} \\ \begin{matrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \\ c_6 \\ c_7 \\ c_8 \\ c_9 \\ c_{10} \\ c_{11} \\ c_{12} \\ c_{13} \\ c_{14} \end{matrix} & \left[\begin{array}{cccc} 0 & 1 & 3 & \sqrt{3} \\ 1 & 0 & 2 & 1 \\ 2 & 1 & 1 & 2 \\ \sqrt{3} & 1 & \sqrt{3} & 0 \\ \sqrt{7} & \sqrt{3} & 1 & 1 \\ 3 & 2 & 0 & \sqrt{3} \\ 4 & 3 & 1 & \sqrt{7} \\ \sqrt{13} & \sqrt{7} & 1 & 2 \\ \sqrt{3}/3 & 2\sqrt{3}/3 & 2\sqrt{21}/3 & \sqrt{37}/4 \\ \sqrt{21}/6 & \sqrt{21}/6 & \sqrt{237}/6 & \sqrt{17}/4 \\ 2\sqrt{3}/3 & \sqrt{3}/3 & \sqrt{39}/3 & \sqrt{3}/3 \\ \sqrt{155}/12 & \sqrt{219}/12 & \sqrt{1115}/12 & \sqrt{203}/12 \\ \sqrt{219}/12 & \sqrt{155}/12 & \sqrt{795}/12 & 5\sqrt{3}/12 \\ \sqrt{21}/3 & \sqrt{21}/3 & 5\sqrt{3}/3 & 2\sqrt{3}/3 \end{array} \right] \end{matrix}$$

3. HEURISTIC ALGORITHM

It is well known that finding an optimal solution to this problem is *NP-hard*. In this section, a two-phase heuristic algorithm described below is proposed to solve this problem.

(1) *Initial assigning phase*: Construct an initial assignment of cells in C^{new} by considering some basic constraints.

(2) *Cell exchanging phase*: Improve the initial assignment by exchanging cells in different switches.

Initial assigning phase. First, we introduce some notations required in the following. Given m sets of cells P_l , $l = 1, 2, \dots, m$, we assume $P_1 \cup P_2 \cup \dots \cup P_m = C^{\text{old}}$ and $P_i \cap P_j = \emptyset$, where $i \neq j$, $i, j = 1, 2, \dots, m$. For example, in Figure 3, $P_2 = \{c_1, c_2\}$, $P_4 = \{c_3, c_4, c_5\}$, $P_3 = P_1 = \emptyset$. Without loss of generality, we assume that the cells in set P_j is assigned to the switch s_j , $j = 1, 2, \dots, m$, and this definition can be extended to include the set C^{new} . Let $\text{sid}(c_i) = l$ if c_i is in P_l , l is called the sid of cell c_i . Let $\text{LUCS}(i, l) = \sum_{c_j \in P_l} w_{ij}$ be the sum of the location update costs between cell c_i and all cells in switch s_k , and $D(i, l) = \text{LUCS}(i, l) - \text{LUCS}(i, \text{sid}(c_i))$. Therefore, for a given partition P , the location update (handoff) cost of the partition is

$$\alpha \sum_{c_i \in C} \sum_{s_l \in S} (\text{LUCS}(i, l) d_{\text{sid}(c_i)l}) \quad (7)$$

For example, as seen in Figure 3, $c_1 \in P_2$ and $c_2 \in P_2$, thus $\text{sid}(c_1) = \text{sid}(c_2) = 2$, $\text{LUCS}(2, 2) = \sum_{c_j \in P_2} w_{2j} = w_{21} + w_{22} = 4 + 0 = 4$, $\text{LUCS}(2, 4) = \sum_{c_j \in P_4} w_{2j} = w_{23} + w_{24} + w_{25} = 3 + 8 + 0 = 11$. $D(2, 4) = \text{LUCS}(2, 4) - \text{LUCS}(2, 2) = 11 - 4 = 7$.

To evaluate the effect of cell c_i in C^{new} being assigned to switch s_k , we must compute the cabling and location update costs derived from this event. By the definition described above, the cabling cost is l_{ik} . The location update cost has two components, one is the location update costs between cells c_i and cells in C^{old} ; the other is the location update costs between cell c_i and the other cells in C^{new} . Since cells in C^{old} have been assigned to switches on the ATM, if cell c_i in C^{new} is assigned to s_k , the location update cost between the cell c_i and the cell c_j in C^{old} is fixed and can be computed by

$$A_{ik} = \alpha \sum_{s_l \in S, l \neq k} (\text{LUCS}(i, l) d_{kl}), \quad (i = n - n' + 1, n - n' + 2, \dots, n; k = 1, 2, \dots, m) \quad (8)$$

Consider the example shown in Figure 3, if c_6 is assigned to s_3 and $\alpha = 1$, then $A_{63} = \text{LUCS}(6, 1) \times d_{31} + \text{LUCS}(6, 2) \times d_{32} + \text{LUCS}(6, 4) \times d_{34} = 0 \times 30 + 0 \times 20 + (2 + 4) \times 30 = 180$. Since cells in C^{new} have not yet been assigned to switches, the location update cost between cell c_i and the other cells in C^{new} cannot be computed by a determinative formula. To estimate the location update cost, let $\text{avg DIST}_k = \sum_{l=1}^m d_{kl} / (m - 1)$ be the average distance between switch s_k and the other switch, $\text{avg LU}_i = \sum_{j=1}^{n'} w_{ij} / n$ be the average location update cost between c_i and the other cell c_j in C^{new} . It is worth noting that if two cells are assigned to the same switch, then the handoff cost between these two cells is ignored. If cell c_i in C^{new} is assigned to the switch s_k and the capacity of switch s_k is Cap_k , i.e. if all cells are assigned to switches, then at most $n' - \text{Cap}_k$ cells should be computed in considering the location update cost. Let NL_i be the number of cells in C^{new} which the frequency of handoff between c_i and c_j is greater than zero. The total location update cost between c_i which assigned to s_k and the other cells in C^{new} assigned to other switches can be estimated by

$$B_{ik} = \begin{cases} \alpha \times (n' - \text{Cap}_k) \times \text{avg LU}_i \times \text{avg DIST}_k & \text{if } \text{Cap}_k \leq NL_i \\ 0 & \text{otherwise} \end{cases} \quad (9)$$

If $\text{Cap}_k > NL_i$, then NL_i cells can be assigned to the same switch s_k , that is, B_{ik} is set to 0.

For example, if $c_6 \in C^{\text{new}}$ is assigned to switch s_3 in Figure 3, then $\text{avg DIST}_3 = 23.33$, $\text{avg LU}_6 = (w_{67} + w_{68})/9 = 0.556$. Assume $\text{Cap}_3 = 2$, at most 7 cells in C^{new} should be computed by considering the location update cost with cell c_6 . Thus, $B_{63} = \alpha \times 7 \times 0.556 \times 23.33 = 90.728$.

After computing the cabling and the location update costs between cells, it is time to assign cells in C^{new} to switches according to these costs. First, we transformed the original problem to a *minimal weighted matching problem*. Then the famous and efficient algorithms for the minimal weighted matching problem in the literature can be used to assign cells to switches. The transformation process is shown as follows. First, a *bipartite graph* $BG(C^{\text{new}}, S^{\text{cap}})$ is constructed; let $S^{\text{cap}} = \{s_{11}, s_{12}, \dots, s_{1\text{cap}_1}, s_{21}, s_{22}, \dots, s_{2\text{cap}_2}, \dots, s_{k1}, s_{k2}, \dots, s_{k\text{cap}_k}, \dots, s_{m1}, s_{m2}, \dots, s_{m\text{cap}_m}\}$, i.e. for each switch s_k in S , Cap_k nodes, $s_{k1}, s_{k2}, \dots, s_{k\text{cap}_k}$, are constructed and assume that each node in S^{cap} can only be assigned one cell. Let C^{new} be the set of cells that have not yet been assigned to switches. Then, for each cell in C^{new} , there is one-edge connected cell to each node in S^{cap} . The weight of edge which connected cell c_i in C^{new} to $s_{kp} \in S^{\text{cap}}$, $p = 1, 2, \dots, \text{Cap}_k$ is set to be

$$l_{ik} + A_{ik} + B_{ik} \quad (10)$$

The transforming result of Example 1 is shown in Figure 5.

Assume $\text{Cap}_1 = 4$, $\text{Cap}_2 = 2$, $\text{Cap}_3 = 4$, and $\text{Cap}_4 = 1$. After transformation, the resulting bipartite graph of Example 1 is shown in Figure 5. Two dummy cells c_{15} and c_{16} are introduced into the bipartite graph. For example, the weight of edge which connected cell c_6 to s_{3*} (e.g. s_{31}, s_{32}) is $l_{63} + A_6 + B_6 = 0 + 180 + 90.728 = 270.728$. Obviously, the subproblem can be formulated as a minimal weighted matching problem on bipartite graph, which is known as the *assignment problem* [6]. Therefore, Hitchcoch Algorithm [6] can be applied to find the optimal solution of assignment problem in $O(\max\{|C^{\text{new}}|, |S^{\text{cap}}|\}^3) = O(\max\{n', \sum_{k=1}^m \text{Cap}_k\}^3) = O((\sum_{k=1}^m \text{Cap}_k)^3)$ time.

Cell exchanging phase. The goal of cell exchanging phase is to select two cells in C^{new} which have been assigned to different switches and exchange them in order to reduce the total cost. The basic idea of cell exchanging phase is borrowed from Kernighan–Lin [7] algorithm which only works on traditional 2-way graph partition problems. In our two-level cell-switch wireless ATM network environment, location update and cabling costs must be considered simultaneously. Hence, we modified Kernighan–Lin’s algorithm to exchange cells in different switches by selecting the ‘most preferable’ cells to exchange instead of arbitrarily exchange two cells.

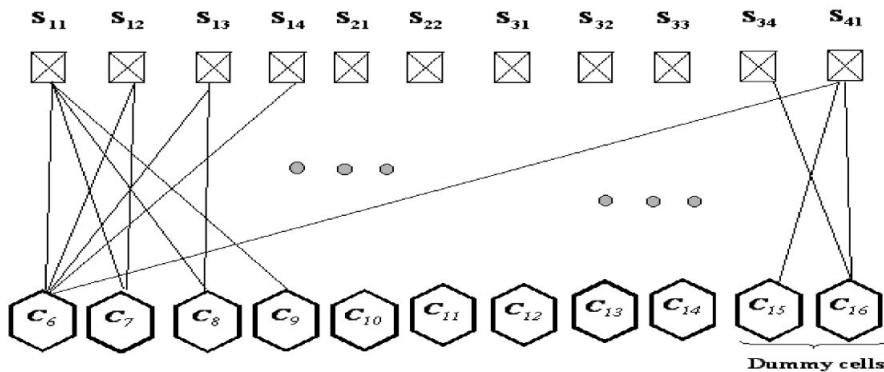


Figure 5. Transformation of initial assignment phase to Example 1.

We have the following lemmas:

Lemma 1[4]

Ignoring the capacity restriction, if cell $c_i \in C^{\text{new}}$ in k is moved to l , then

- (1) the cabling cost is reduced by $d_{ik} - d_{il}$,
- (2) the location update cost is reduced by $\alpha(\text{LUCS}(i, l) - \text{LUCS}(i, k)) d_{kl}$.

If cell c_i and c_j are exchanged, the gain can be computed from the following lemma:

Lemma 2[4]

If cell $c_i \in C^{\text{new}}$ in switch s_k and $c_j \in C^{\text{new}}$ in switch s_l are exchanged, then

- (1) The reduced cabling cost is: $RC(i, j) = d_{ik} - d_{il} + d_{jl} - d_{jk}$
- (2) The reduced location update cost is

$$\begin{aligned} \text{RLU}(i, j) = & (D(i, l) - D(j, k) - 2w_{ij}) \times d_{kl} \times \alpha + (d_{kz} - d_{lz}) \\ & \alpha \times (\text{LUCS}(i, s) - \text{LUCS}(j, z)) \end{aligned}$$

- (3) The reduced total cost is: $\text{exchange}(i, j) = RC(i, j) + \text{RLU}(i, j)$.

After cells $c_i \in C^{\text{new}}$ and $c_j \in C^{\text{new}}$ have been selected and exchanged, the matrix LUCS should be updated so that the algorithm can run effectively and avoid sequential searching.

Lemma 3[4]

After cells $c_i \in C^{\text{new}}$ in switch s_k and $c_j \in C^{\text{new}}$ in switch s_l are exchanged, the LUCS values can be updated as follows:

- (1) $\text{LUCS}^{\text{new}}(i, k) = \text{LUCS}(i, k) + w_{ij}$
- (2) $\text{LUCS}^{\text{new}}(i, l) = \text{LUCS}(i, l) - w_{ij}$
- (3) $\text{LUCS}^{\text{new}}(j, k) = \text{LUCS}(j, k) - w_{ij}$
- (4) $\text{LUCS}^{\text{new}}(j, l) = \text{LUCS}(j, l) + w_{ij}$

For any cell a in switch s_z

- (5) $\text{LUCS}^{\text{new}}(a, k) = \text{LUCS}(a, k) + w_{aj} - w_{ai}$, if $(s_z = k)$ and $(a \neq i)$
- (6) $\text{LUCS}^{\text{new}}(a, l) = \text{LUCS}(a, l) - w_{aj} + w_{ai}$, if $(s_z = k)$ and $(a \neq i)$
- (7) $\text{LUCS}^{\text{new}}(a, k) = \text{LUCS}(a, k) + w_{aj} - w_{ai}$, if $(s_z = l)$ and $(a \neq j)$
- (8) $\text{LUCS}^{\text{new}}(a, l) = \text{LUCS}(a, l) - w_{aj} + w_{ai}$, if $(s_z = l)$ and $(a \neq j)$
- (9) $\text{LUCS}^{\text{new}}(a, k) = \text{LUCS}(a, k) + w_{aj} - w_{ai}$, if $(s_z \neq k \text{ and } s_z \neq l)$ and $(a \neq i \text{ and } a \neq j)$
- (10) $\text{LUCS}^{\text{new}}(a, l) = \text{LUCS}(a, l) - w_{aj} + w_{ai}$, if $(s_z \neq k \text{ and } s_z \neq l)$ and $(a \neq i \text{ and } a \neq j)$

Given an initial assignment, the total cost can be reduced by reassigning a cell in current switch to another switch, or exchanging two cells which can be assigned to different switches. A set DM of $\sum_{k=1}^m \text{Cap}_k - |C^{\text{new}}|$ dummy cells are introduced into the CG to ensure that reassigning one cell in one step is possible. Cell exchanging phase consecutively selects two cells in $C^{\text{new}} \cup DM$ which can be assigned to different switches to exchange. At each iteration, two cells c_a and $c_b \in C^{\text{new}} \cup DM$ are selected which maximize the reduced exchanging cost $\text{exchange}(a, b)$ where

$$\text{exchange}(a, b) = \max_{(c_i, c_j) \in \{C^{\text{new}} \cup DM\} \times \{C^{\text{new}} \cup DM\}} \text{exchange}(i, j) \quad (11)$$

The iteration continues if $\text{exchange}(a, b) \geq 0$.

Algorithm: Cell exchanging

Step 1: For each cell in $CG \cup DM$ and each switch in G , compute values of matrices LUCS (i, l) and $D(i, l)$. Let $C = C^{\text{new}} \cup DM$.

Step 2: Find two cells c_a and c_b in B assigned different switches such that

$$\text{exchange}(a, b) = \max_{(c_i, c_j) \in \{C^{\text{new}} \cup DM\} \times \{C^{\text{new}} \cup DM\}} \text{exchange}(i, j)$$

Step 3: If $\text{exchange}(a, b) > 0$, then exchange cells c_a and c_b ; and delete c_a and c_b from B .

Step 4: Update LUCS (i, l) and $D(i, l)$ for each $c_i \in B$ and s_l in L .

Step 5: If B is go to Step 2.

Step 6: If $\text{exchange}(c_a, c_b) > 0$ then go to Step 1; otherwise terminate the algorithm.

4. SIMULATED ANNEALING ALGORITHM

Simulated annealing is a stochastic computational technique derived from statistical mechanics for finding near globally-minimum-cost solutions to large optimization problems. Kirkpatrick *et al.* [8] were the first to propose and demonstrate the application of simulation techniques from statistical physics to the problem of combinatorial optimization. Due to the complexity of the extended cell assignment problem in a two-level wireless ATM network, the provision of an optimal solution in reasonable time is not guaranteed. In this respect, the usual step is to devise an approximate algorithm for solving this problem. The simulated annealing (SA) technique is applied to solve the extended cell assignment problem in this section.

In the design of simulated annealing algorithm, if the traditional-SA approach is used to solve the extended cell assignment problem, it may generate a significant number of configurations, but only a small fraction of these are indeed constraint-satisfied (10 per cent or 20 per cent). Thus, the performance of the traditional-SA algorithm is not promising. In this paper, we attempt to develop an *enhanced-SA* approach to solve the extended cell assignment problem by generating configurations, which are constraint-satisfied. The key elements in a simulated annealing algorithm are a cost function, a configuration space, a perturbation mechanism, and a cooling schedule. In our case, the solution method is shown as follows.

4.1. Configuration space and perturbation mechanism

The objective of the extended cell assignment problem in a two-level wireless ATM network is to find an optimal assignment of new adding and splitting cells to switches so that the object function value is minimized. To do this, the configuration space is designed to be the set of possible solutions, which is defined as a binary $\sum_{k=1}^m \text{Cap}_k \times m$ matrix X . Define x_{ik} in X to be 1 if the cell c_i is assigned to switch s_k ; $x_{ik} = 0$, otherwise. Consider a possible assignment of the example shown in Figure 3, the configuration matrix of the assignment is shown in Figure 6(a). In this example, two dummy cells c_{15} and c_{16} are introduced into the configuration matrix (configuration).

It is worth noting that the configuration can be divided into two parts, fixed and variable parts. The first part of matrix, which represents the assigning status of cells in C^{old} , is fixed in the running of SA. Thus, the first part of the matrix can be ignored since it is never changed during experiments. For the purpose of easy understanding, the fixed part of the matrix is still kept in configuration in the rest of the paper.

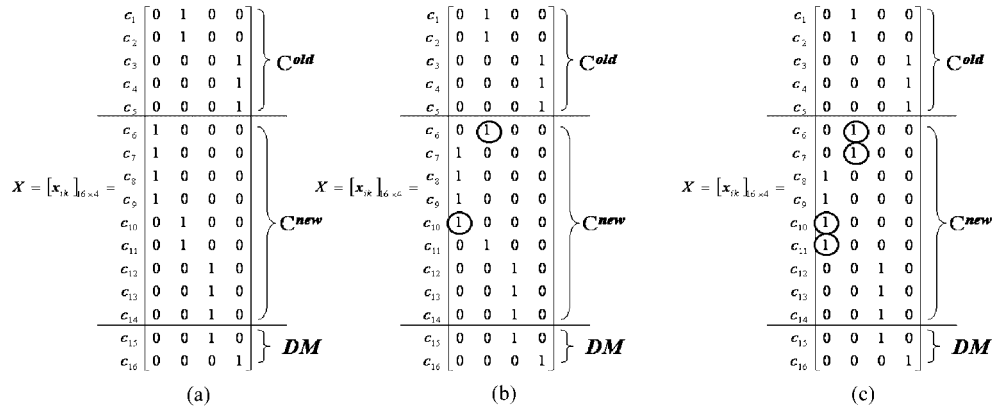


Figure 6. Possible feasible configuration for Example 1, (b) Cell exchanging for Example 1 and (c) Multiple cells exchanging for Example 1.

If the initial configuration is randomly generated or new configuration is formed by changing the assigning status of cells by way of randomly choosing cells and switches, then, there is a large chance that the generated configuration is not a constraint-satisfied one. To avoid this, algorithms, which are guaranteed to generate constraint-satisfied configurations and perturbation schemes, must be constructed. To generate the constraint-satisfied initial configuration, we propose the following algorithm:

Algorithm: Initial configuration generating algorithm(ICGA)

Step 1. Let $A = S$, $B = C^{new} \cup DM$.

Step 2. Repeat Steps 2.1, 2.2 and 2.3 until B is empty.

Step 2.1. Randomly select a switch s_a from A .

Step 2.2. Randomly assign $|Cap_k|$ cells in B to switch s_a and remove these cells from B .

Step 2.3. Remove s_a from A .

In the traditional-SA algorithm, the search process may get stuck at a local minimum due to the small change moves, particularly as the barrier is high and the temperature is low. Hence, we introduce an idea of *large perturbation schema* whose function is the same as the *mutation operation* in *genetic algorithm*. The main idea of large perturbation schema is to leap over the barrier during a search and to explore another region of the search space. This can be achieved by applying a certain number of moves, consecutively, special problem domain perturbations, or local search heuristic perturbation. Three types of perturbations are introduced to the enhanced-SA algorithm, which are shown as follows.

- *Cells exchanging schema:* First, the cells exchanging schema randomly selects two cells c_i and c_j in $C^{new} \cup DM$, which have been assigned to different switches s_k and s_l , respectively. Then, exchange the cell c_i to the switch s_l and the cell c_j to the switch s_k . For example, assume cells c_6 and c_{10} are randomly selected; after performing cells exchanging perturbation, the configuration matrix is shown in Figure 6(b).
- *Multiple cells exchanging schema:* First, the multiple cells exchanging schema randomly selects two switches s_k and s_l from S . Then, reassigns cells in the switch s_k to the switch s_l and *vice versa*. Since the original configuration is constraint-satisfied, after perturbation,

Table I. Nine cases for consideration in multiple cells exchanging.

	$n_k > \text{Cap}_l$	$n_k = \text{Cap}_l$	$n_k < \text{Cap}_l$
$n_l > \text{Cap}_k$	Contradiction	Contradiction	Case 1
$n_l = \text{Cap}_k$	Contradiction	Directed exchange	Directed exchange
$n_l < \text{Cap}_k$	Case 2	Directed exchange	Directed exchange

the resulting configuration must be a feasible one. If we reassign cells in two switches directly, the resulting configuration may violate the constraints. It can be proven that several cases need to be considered as shown in Table I (contradiction means that this case will not appear). As seen in Table I, we found that the following strategies can be used to guarantee that the final configuration is constraint-satisfied. Let COS_k be the set of cells which are currently assigned to switch s_k . In case 1, if $n_l > \text{Cap}_k$ and $n_k < \text{Cap}_l$, then Cap_k cells are randomly selected from the set COS_l and reassigned to switch s_k ; at the same time, all cells in COS_k are reassigned to the switch s_l . In case 2, if $n_k > \text{Cap}_l$ and $n_l < \text{Cap}_k$, Cap_l cells are randomly selected from the set COS_k and reassigned to the switch s_k ; at the same time, all cells in COS_k are reassigned to the switch s_l . Otherwise, all cells being assigned to the switch s_k are directly reassigned to the switch s_l and *vice versa*. For example, assume switches s_1 and s_2 are randomly selected; after performing multiple cells exchanging, the configuration matrix is shown in Figure 6 (c) (assume cells c_6 and c_7 are randomly selected from the switch s_1).

- *Local search heuristic schema*: The local search heuristic schema used here is trying to find the local optimum by exchanging cells in different switches. The cell exchanging heuristic described in Section 4.2 can be used to find the local optimum of current configuration.

Algorithm: Local Search heuristic perturbation (LSHP)

Step 1: For each cell in CG and each switch in G , compute values of matrices $\text{LUCS}(i, l)$ and $D(i, l)$. Let $B = C^{\text{new}} \cup DM$.

Step 2: Find two cells c_a and c_b in B in different switches such that

$$\text{exchange}(a, b) = \max_{(c_i, c_j) \in B \times B} \text{exchange}(i, j)$$

Step 3: If $\text{exchange}(a, b) > 0$, then exchange cells c_a and c_b .

It is worth noting that if the probabilistic decision mechanism of SA is disabled, and three perturbation schemes are used to perturb the initial configuration to generate a new one, then, it is easy to prove that all possible feasible configurations can be reached by applying a sequence of perturbations. In the experiment, let p_1 , p_2 , and p_3 be the probabilities of transforming current configuration to a new one by applying cell exchanging schema, multiple cells exchanging schema, and local search heuristic perturbation, respectively. We assume $p_1 + p_2 + p_3 = 1$ and the values of p_1 , p_2 , and p_3 will be empirically determined and described in later sections.

4.2. Cooling schedule

One of the most important problems involved in the simulated annealing algorithm implementation is the definition of a proper cooling schedule, which is based on the choice of

the following parameters: starting temperature, final temperature, length of Markov chains, the way of decreasing temperature. A correct choice of these parameters is crucial because the performances of the algorithm strongly depend on it. In the literature many cooling schedules are maintained [9]. These cooling schedules lead to a polynomial time execution of the simulated annealing, but it cannot give any guarantee for the deviation in cost between the final solution obtained by the algorithm and the optimal cost [9,10]. The different parameters of the cooling schedule are determined based on the statistics calculated during the search. In the following we describe these parameters.

Initial value of the control parameter: The rule used in our enhanced-SA is that the starting temperature c_0 is determined by calculating the average increase in cost, $\overline{\Delta C^+}$, for 50 random transitions and solve c_0 from $c_0 = \overline{\Delta C^+} / \ln(\chi_0^{-1})$, where *accepted ratio* χ_0 is defined as the number of accepted transitions divided by the number of proposed transitions. In this paper, the accepted ratio χ_0 is empirically set to 0.99.

Decrement of the control parameter: The decreasing rate of the temperature needs to be small enough to reach thermal equilibrium for each temperature value. As the temperature is decreased, the accepted ratio is lowered. When no solution that increases the objective function can be found, the system is 'frozen' and converged to a certain solution. The speed of coverage of the simulated annealing algorithm depended on the decreasing rate of the temperature and the length of the Markov chain. As mentioned in Reference [9], the decrement is chosen such that small Markov chain lengths suffice to re-establish quasi-equilibrium after the decrement of the temperature is applied. The decrement rule in enhanced-SA is defined as follows: $T_{k+1} = \gamma T_k$, where γ is empirically set to 0.99.

The final value of the control parameter: The iterative procedure is terminated when there is no significant improvement in the solution after a pre-specified number of iterations. It can also be terminated when the maximum number of iterations is reached.

The length of Markov chains: In References [9,10], it is concluded that the decrement function of the control parameter requires only a 'small' number of trial solutions to rapidly approach the stationary distribution for a given next value of the control parameter. In general, a chain length of more than 100 transitions is reasonable. In this paper, the chain length is empirically set to a value of n .

Enhanced-SA algorithm of extended cell assignment problem: The details of the simulated annealing is described as follows:

Algorithm: Enhanced-SA algorithm

Step 1. For a given initial temperature T , perform *Initial Configuration Generating Algorithm* to generate initial configuration IC. The currently best configuration (CBC) is IC, i.e. $\text{CBC} = \text{IC}$, and the current temperature value (CT) is T , i.e. $\text{CT} = T$.

Step 2. If $\text{CT} = 0$ or the stop criterion is satisfied then go to *Step 7*.

Step 3. Generate a random number p in $[0, 1)$, if $p \leq p_1$ then new configuration (NC) is generated by applying cells exchanging schema; if $p_1 < p \leq p_1 + p_2$ then NC is generated by applying multiple cells exchanging schema; otherwise NC is generated by applying local search heuristic schema.

Step 4. The difference of the costs of the two configurations, CBC and NC is computed, i.e. $\Delta C = E(\text{CBC}) - E(\text{NC})$.

Step 5. If $\Delta C \geq 0$ then the new configuration NC becomes the currently best configuration, i.e. $\text{CBC} = \text{NC}$. Otherwise, if $e^{-(\Delta C/\text{CT})} > \text{random}[0, 1)$, the new configuration NC becomes the currently best configuration, i.e. $\text{CBC} = \text{NC}$. Otherwise, go to *Step 2*.

Step 6. The cooling schedule is applied, in order to calculate the new current temperature value CT and go to Step 1.

Step 7. End.

5. EXPERIMENTAL RESULTS

From the previous sections, a heuristic algorithm and a simulated annealing algorithm are proposed to solve the extended cell assignment problem. In order to evaluate their performances, we have implemented the algorithms and applied them to a number of examples with randomly positioned cells and switches. The results of these experiments are reported below. For all experiments, the implementation language is C, and some experiments have been made on Windows NT with a Pentium II 450MHz CPU and 256MB RAM. We simulated a hexagonal system in which cells are configured as an H-mesh. The handoff frequency f_{ij} of two cells was generated by a normal random number generator with mean 100 and variance 20.

To examine effects of different number of cells, Cell Graph CG with $n = 50, 100, 150$, and 200 cells were tested. $|C^{\text{new}}| = 3n/4, |C^{\text{old}}| = n/4, m = 10, \alpha = 1$ and the Cap/n value of each problem is 0.2. The proposed heuristic algorithm consists of two phases (initial assigning phase and cell-exchanging phase), termed EAA, which is a heuristic approach to a rather complex problem. To measure the performance of each phase of the EAA algorithm, we constructed a two-phase heuristic algorithm, termed NSF, as the reference. The first phase of the NSF algorithm is assigning each cell to the nearest switch; if the nearest switch is full then find the next nearest switch. The second phase of the NSF algorithm is the cell exchanging phase same as in EEA. For all experiments, the CPU time in seconds of the heuristic algorithm and the objective cost reduction ratio are two major concerns. First, the cost reduction ratio of the cell exchanging phase is evaluated. To test the effect of cell exchanging phase, we compare the costs of the algorithm by running this phase. Let C_A , and C_B be the costs resulting from running cell initial assigning and cell exchanging phases, respectively. The cost improvement ratio $(C_A - C_B)/C_A$ is shown in Table II. As seen in Table II, after running cell exchanging phase, the total cost of EAA algorithm is reduced by 8.4 per cent on average.

To know the efficiency of the enhanced-SA algorithm (ESA) to the traditional-SA algorithm, we also implement a traditional-SA algorithm that does not guarantee to generate constraint-satisfied configurations. The experimental results shown in Figure 7 explain that the enhanced-SA algorithm have better performance than the traditional-SA algorithm. In other words, constraint-satisfied configurations and perturbation schemes of enhanced-SA algorithm are indeed more efficient than the traditional-SA algorithm and enhanced-SA algorithm has better convergent behaviours.

Table II. Cost reduction ratio of cell exchanging phase.

# of cells	NSF (%)	EEA (%)
50	4.2	6.5
100	3.2	9.3
150	3.7	9.7
200	3.1	7.9
Average	3.6	8.4

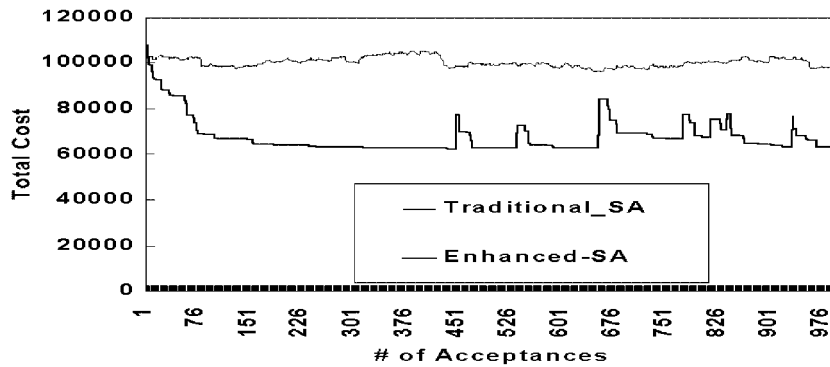


Figure 7. Comparison of the result of traditional- and enhanced-SA algorithms.

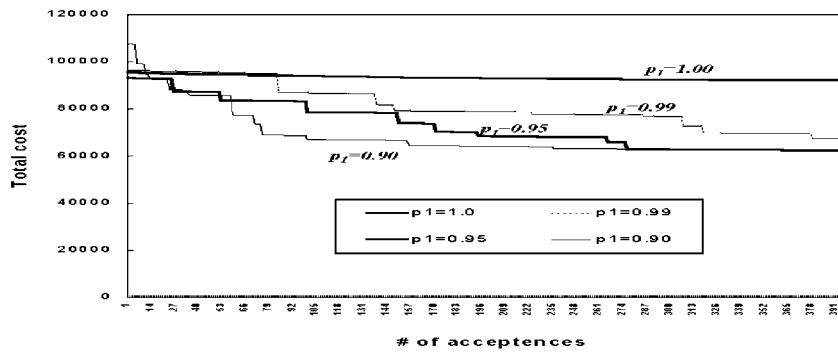


Figure 8. Comparison of the convergence of different probabilities of p_1 and p_2 .

To evaluate the effect of the probabilities of different perturbation schemes of enhanced-SA algorithm described in Section 5, we test the enhanced-SA algorithm with different values of probabilities p_1 , p_2 and p_3 . The experiments consist of three parts:

- (1) Assume p_3 is fixed, the effect of different values of p_1 and p_2 is tested,
- (2) Assume p_1 is fixed, the effect of different values of p_2 and p_3 is tested,
- (3) Assume p_2 is fixed, the effect of different values of p_1 and p_3 is tested.

In the first part of the experiments, assume $p_3 = 0$, $p_2 = 1 - p_1$, and the value of p_1 is in $\{1.00, 0.99, 0.95, 0.90, 0.01, 0.00\}$. Figures 8 and 9 show the results of the experiment, where x -axis represents the number of acceptances when the configuration is perturbed in enhanced-SA algorithm and y -axis represents the total cost of the problem instance. When $p_1 = 1.00$ ($p_2 = 0.00$), i.e. the multiple cells exchanging perturbation does not activate in the experiment. We found that the enhanced-SA algorithm converges very slowly and traps into local minima, since the cell exchanging perturbation only exchanges two cells at one time. When $p_1 = 0.99$, $p_1 = 0.95$, or $p_1 = 0.90$, that is, only a small chance that the multiple cells exchanging perturbation may activate in the experiment, the enhanced-SA algorithm converges faster than

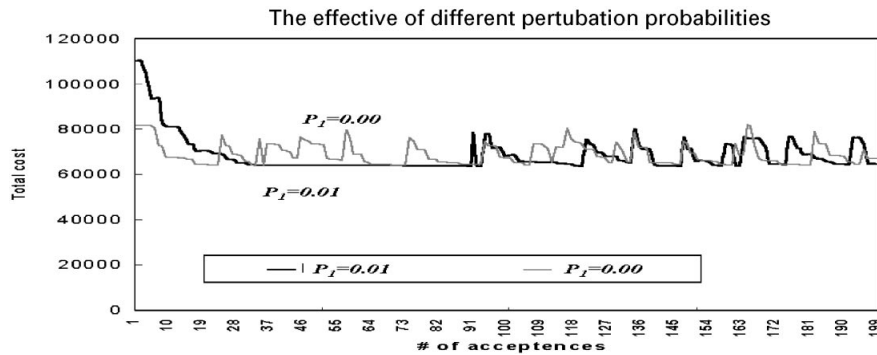


Figure 9. Comparison of the convergence of different probabilities of p_1 and p_2 .

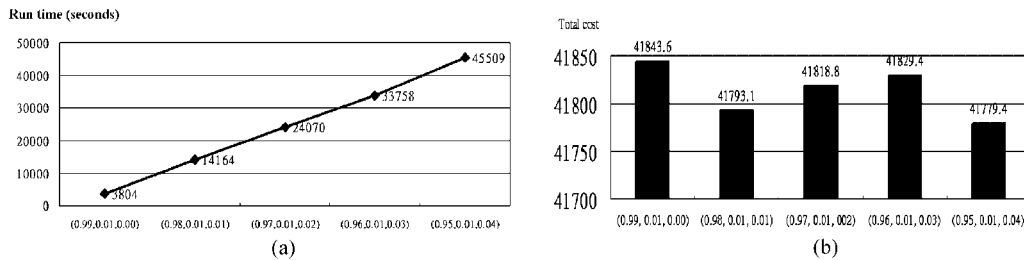


Figure 10. Comparison of the different probabilities of p_1 and p_3 . (a) Running time and (b) Total cost.

the case with $p_1 = 0$ and has very good performance as shown in Figure 8. When $p_1 = 0.00$ or $p_1 = 0.01$, the single cell exchanging perturbation does not activate or has very small chance to activate. As seen in Figure 9, the current configuration up and down rapidly and hardly converges to the global minima. Thus, we can conclude that if the probability of cells exchanging perturbation is higher ($p_1 = 0.90\text{--}0.98$), the enhanced-SA algorithm has very good performance.

In the second part of experiments, assume p_2 is fixed and is equal to 0.01, the value of p_1 is in $\{0.99, 0.98, 0.97, 0.96, 0.95\}$, and $p_3 = 0.99 - p_1$. The experimental results of the running time and the total cost of different problem instance are shown in Figures 10(a) and (b), respectively. As seen in Figure 10, we found that the execution time of ESA is increasing at a higher rate as the value of p_3 increases; but the result of total cost does not reach the best.

In the third part of the experiments, assume p_1 is fixed and is equal to 0.95, the value of p_2 is in $\{0.00, 0.01, 0.02, 0.03, 0.04, 0.05\}$, and $p_3 = 0.05 - p_1$. The experimental results of the running time and the total cost of different problem instance are shown in Figures 11(a) and (b), respectively. As seen in Figure 11, we found that the execution time of ESA is increasing at a higher rate as the value of p_3 increases; but the result of total cost does not reach the best. Therefore, we can conclude that the activation probabilities of multiple cells exchanging perturbation and local search heuristic perturbation do effect the result of ESA algorithm. If the values of p_2 and p_3 are kept smaller (0.01 or 0.02) then ESA algorithm has better performance and behaviour.

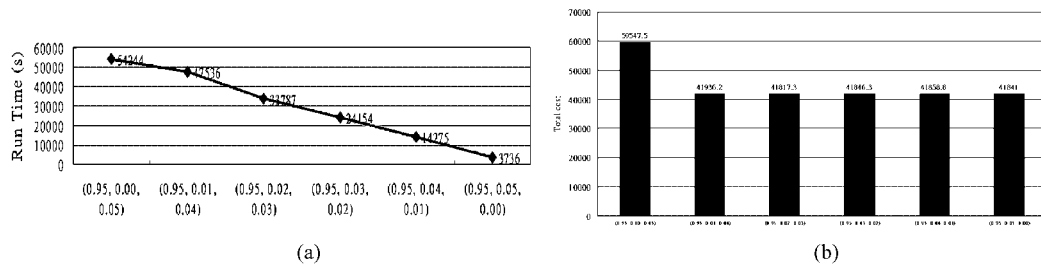
Figure 11. Comparison of the different probabilities of p_2 and p_3 . (a) Running time and (b) Total cost.

Table III. Comparison of NSF, EEA, and ESA solutions.

$ n $	$ m $	Cap	NSF		EEA		ESA				
			Min.	Ratio (%)	Min.	Ratio (%)	Min.	Mode	Avg.	Max.	Std. Dev.
50	10	10	2676.4	107	2552.5	103	2476.7	2478.2	2480.4	2485.7	352.26
100	10	20	6516.5	108	6154.7	102	6011.3	6026.9	6022.9	6035.7	1141.1
150	10	30	14806.2	104	14823.4	104	14219.9	14236.8	14237.0	14257.0	4266.5
200	10	40	24089.9	109	22740.7	103	22052.6	22075.8	22075.9	22099.8	7289.7

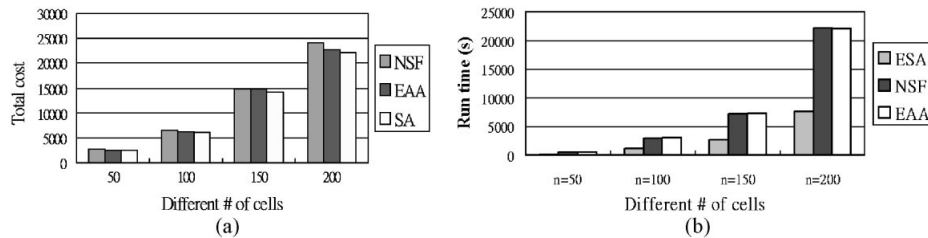


Figure 12. Comparison of the NSF, EEA and ESA solutions for 5 different example problems. (a) Total cost and (b) Running time.

The results of the executions of the NSF, the EEA and the ESA for the above networks are given in Table III. Since the NSF and the EEA are deterministic algorithms, the corresponding columns show the solutions found in the single run of the NSF and the EEA on each example problem. However, the ESA was run with 50 random seeds on each problem in order to get some statistical information about the quality of their solutions. The ESA column of Table III shows the minimum, mode, average, maximum, and standard deviation of the cost of solutions for 50 runs. The most repeated solution or mode is close to the minimum cost solution for all example problems. Take the minimum of the total cost of EEA as a reference, we compute the result of the EEA to the ESA, the ratio is 103 per cent on average.

The solutions of NSF, EEA, and ESA (with probabilities $p_1 = 0.98$, $p_2 = p_3 = 0.01$) are shown on the bar chart of Figure 12(a). As seen in Figure 12(a), ESA's advantage for the quality of solutions increases as the network gets larger. Run times of a single run of the NSF, the EEA, and the ESA for five example problems are compared in Figure 12(b). The ESA algorithm is

quicker than the NSF and EEA algorithms for all these five examples. Moreover, the execution times of the NSF and EEA increase at a higher rate than that of the ESA algorithm as the cell size increases. The ESA algorithm is superior to the NSF and EEA algorithms in both run time and quality.

6. CONCLUSIONS

In this paper, we have investigated the *extended cell assignment problem* which optimally assigned new adding and splitting cells in PCS to switches in a wireless ATM network. Given cells in PCS and switches on ATM network (whose locations are fixed and known), we would like to do the assignment in an attempt to minimize a cost criterion. 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 modelled as a complex integer programming problem and it is easy to recognize that finding an optimal solution of this problem is *NP-hard*.

A heuristic algorithm EEA (extended assignment algorithm) and a simulated annealing algorithm ESA (enhanced-SA) are proposed to solve this problem. The heuristic algorithm EEA consists of two phases, *initial assignment phase*, and *cell exchanging phase*. First, in the initial assigning phase, the initial assignments of cells to switches are found. Then, these assignments are improved by performing cell exchanging phase, which repeatedly exchanges two cells in different switches with great reduction of the total cost. Experimental results indicate that EEA algorithm has good performance and shows that cell exchanging phase can really reduce total cost near 8.4 per cent on average.

Owing to the inability of SA to generate solutions that always satisfy all the constraints, the performance of a traditional-SA approach is not so promising. The SA technique is, however, easy to implement, requires little expert knowledge and is not memory intensive. Hence, in this paper, we have developed an enhanced-SA algorithm to solve the *extended cell assignment problem*. The enhanced-SA algorithm constructs constraint-satisfied configurations and perturbation mechanism to ensure that the candidate configurations produced are constraint-satisfied. The performance of the enhanced-SA algorithm is demonstrated through simulation. The results are compared with previous reported solution method and indicate that the proposed algorithm runs efficiently.

ACKNOWLEDGEMENTS

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

REFERENCES

1. Cheng M, Rajagopalan S, Chang LF, Pollini GP, Barton M. PCS mobility support over fixed ATM networks. *IEEE Communication Magazine* 1997; **18**(2):82–91.
2. Macario RCV. *Cellular Radio*. McGraw-Hill: New York, 1993.
3. Der-Rong Din, Tseng SS. Genetic Algorithms for Optimal design of two-level wireless ATM network. *Proceeding of NSC* 2001; **25**(3):151–162.
4. Der-Rong Din, Tseng SS. Heuristic algorithm for optimal design of two-level wireless ATM network. *Journal of Information Science Engineering* 2001; **17**:674–665.

5. Merchant A, Sengupta B. Assignment of cells to switches in PCS networks. *IEEE/ACM Transaction on Networking* 1995; **3**(5):521–526.
6. Kuhn HW. The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly* 1955; **2**:83–97.
7. Kernighan BW, Lin S. An efficient heuristic procedure for partitioning graphs. *Bell System Technique Journal* 1970; **49**:291–307.
8. Kirkpatrick S, Gelatt CD, Vecchi MP. Optimization by simulated annealing. *Science* 1983; **220**:671–680.
9. Laarhoven V, Arts E. *Simulated Annealing: Theory and Application*. D. Reidel Publishing Company: Holland, 1987.
10. Arts E, Korst J. *Simulated Annealing and Boltzman Machines*. Wiley: New York, 1989.

AUTHORS' BIOGRAPHIES



Der-Rong Din received his BS degree in Computer Science from Chinese Culture University in 1991; the MS and the PhD in Computer Science from National Chiao-Tung University in 1993 and 2001, respectively. Now he is on the faculty of Department of Computer Science and Information Management and the Director of the Computer Center at Hung-Kuang Institute of Technology. His current research interests are in mobile communication, WDM networking, parallel compiler and algorithms.



Shian-Shyong Tseng received his PhD in Computer Engineering from the National Chiao Tung University in 1984. Since August 1983, he has been on the faculty of the Department of Computer and Information Science at National Chiao Tung University, and is currently a Professor there. From 1988 to 1991, he was the Director of the Computer Center at National Chiao Tung University. From 1991 to 1992 and 1996 to 1998, he acted as the Chairman of Department of Computer and Information Science. From 1992 to 1996, he was the Director of the Computer Center at Ministry of Education and the Chairman of Taiwan Academic Network (TANet) management committee. In December 1999, he founded Taiwan Network Information Center (TWNIC) and is now the Chairman of the board of directors of TWNIC. His currently research interests include parallel processing, expert systems, computer algorithm, and internet-based application.