Heuristic and Simulated Annealing Algorithms for Wireless ATM Backbone Network Design Problem*

DER-RONG DIN
Department of Computer Science and Information Engineering
National Changhua University of Education
Changhua City, 500 Taiwan

Personal Communication Network (PCN) is an emerging wireless network that promises many new services for the telecommunication industry. The high speed backbone network (*asynchronous transfer mode*, ATM or *wavelength division multiplexing*, WDM) is one possible approach to provide broadband wireless transmission with PCN's using the ATM switches for interconnection of PCN cells. The *wireless ATM backbone network design* (*WABND*) problem is to allocate backbone links among ATM switches such that the effects of terminal mobility on the performance of ATM-based PCN's can be reduced. In this paper, the WABND problem is formulated and studied. The goal of the WABND is to minimize the location update cost under constraints. Since WABND is an NP-hard problem, two heuristic algorithms and a simulated annealing algorithm were proposed and used to find the close-to-optimal solutions. The simulated annealing algorithm was able to achieve good performance as indicated from the simulated results.

Keywords: wireless ATM, heuristic algorithm, simulated annealing, NP-hard, backbone network design

1. INTRODUCTION

Personal Communication Network (*PCN*) [1, 2] is an emerging wireless network that promises many new services. Users may move from one place to another and can maintain transparent network access through wireless links. Information exchanging between users, may be bidirectional, which includes voice, data, and image. In a PCN, the covered geographical area is typically partitioned into a set of *cells*. Each cell has a *base station* (*BS*) used for exchanging radio signals with mobile terminals. Due to the limited power of wireless transceivers, mobile users can communicate only with base stations that reside within the same cell. Moreover, several coverage areas of cells are grouped and formed a *location area* (*LA*). That is, an LA consists of an aggregation of coverage areas of cells forming a contiguous geographical region.

A typical PCN architecture based on ATM (*asynchronous transfer mode*) switches is illustrated in Fig. 1 (a). The covered geographical area is partitioned into a set $P = \{P_1, P_2, \ldots, P_m\}$ of m disjoint *clusters* (or LAs). An ATM switch is allocated within each cluster and each BS in this cluster is connected to the ATM switch. The ATM switch offers the services of establishing/releasing channels for the mobile terminals in the cluster. Two neighboring clusters can be interconnected via the associated ATM switches. The

Received February 14, 2006; revised September 14, 2006; accepted January 3, 2007.
Communicated by Yu-Chee Tseng.
* This work was supported in part by the National Science Council of Taiwan, R.O.C. under grants No. NSC 95-2221-E-018-012 and NSC 96-2221-E-018-007.

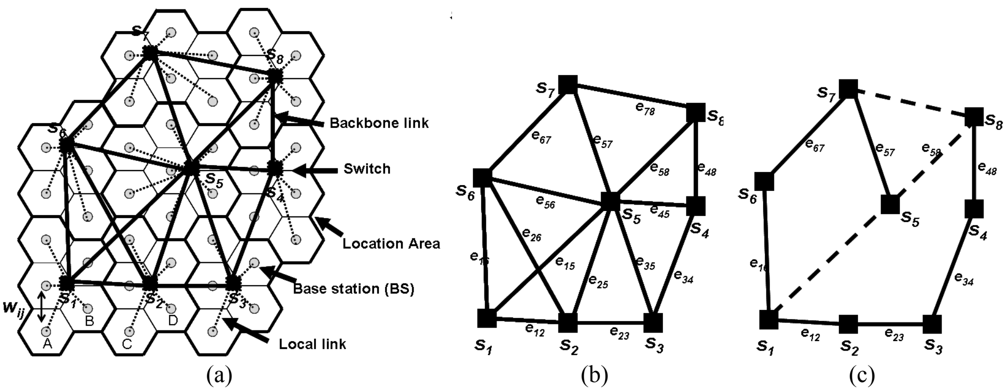


Fig. 1. (a) A typical PCN architecture based on ATM switches; (b) $H(S, F, z)$l (c) $G(S, E, z)$ with $p = 10$, $deg = 3$.

links between the ATM switches are called *backbone links*, and the links between ATM switches and BSs are called *local links* [2].

When a subscriber enters a cell that belongs to a different LA, a *location update* (LU) or *handoff* procedure that informs the network about the subscriber's new location is performed. This will generate network traffic overhead in PCN and consume scarce radio resources. Moreover, LU also increases the load on distributed location databases and, thus, increases the complexity of implementing the databases [3].

During the wireless environment, two types of handoffs should be considered in the designing of the network, they are *inter-switch handoff* and *intra-switch handoff*. The *intra-switch handoff* involves only one switch and the inter-switch handoff involves two switches. The inter-switch handoffs that occur between two cells, which connected to different switches, consume much more network resources (therefore, are much more costly) than the intra-switch handoff that occur between cells, which connected to the same switch [4-6]. Thus, the cost of intra-switch handoffs involving only one switch is negligible in designing the two-level wireless ATM network.

Consider the example shown in Fig. 1, where 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. Now imagine that the subscriber moves from cell B to cell C. Then the inter-switch handoff involves the execution of a fairly complicated protocol between switches s_1 and s_2 [4-6].

In this paper, the *wireless ATM network design* (*WABND*) problem is studied. Given the PCN network, the handoff frequencies between cells, the local links between cells and switches, the degree constraint, and the number of backbone links, the WABND problem is to find a set of backbone links which forms a connected network such that the total cost is minimized under the degree constraint. In [7], the WABND problem is studied and formulated, a heuristic algorithm and a genetic algorithm have proposed to find the sub- optimal solution. Due to the complexity of the WABND problem in a 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. *Simulated annealing* (*SA*) 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 problem of combinatorial optimization. In this paper, two heuristic algorithms and a simulated annealing algorithm are proposed to solve it.

2. NOTATIONS, PROBLEM FORMULATION AND RELATED WORKS

This section first provides an overview of various terms and notations used in explaining the concepts outlined in subsequent sections. There after, the formulation of the problem and the related works are presented.

2.1 Notations and Assumptions

In this paper, the location of cells and switches are fixed and known. Let there be n cells in PCN network $CG(C, L, w)$ where $C = \{c_1, c_2, \ldots, c_n\}$. Define Cell Graph $CG(C, L, w)$, where C is a finite set of cells with $|C| = n$ and L is the set of edges such that $L \subset C \times C$, all the edges are undirected and with weight function w. Let f_{ij} be the cost per unit time of the handoffs that occurs between cell c_i and c_j, $(i, j = 1, 2, \ldots, n)$. Thus, f_{ij} is proportional to the frequency of handoffs that occur between cell c_i and c_j. Assume cells c_i and c_j are connected by an edge (c_i, c_j) in CG with weight f_{ij}. Since the backbone link between two nodes are bidirectional and the traffic between two nodes in different directions is assumed passing through the same shortest routing path. Thus, to simplify the discussion, the handoff frequency between two cells is summed up and represented by $w_{ij} = f_{ij} + f_{ji}$, then $w_{ij} = w_{ji}$, and $w_{ii} = 0$.

An ATM-based PCN topology can be represented by an undirected graph $H(S, F, z)$; where S is a finite set of switches and $|S| = m$. Each node s_k in S stands for a cluster P_k (or an ATM switch) and an edge e_{kl} is in F if clusters P_k and P_l are adjacent in the given network with communication cost z_{kl}. For example, the corresponding graph of the ATM-based PCN topology depicted in Fig. 1 (a) is shown in Fig. 1 (b) and a possible backbone network with 10 backbone links for the PCN topology is given in Fig. 1 (c). Consider Las (or clusters) P_1 and P_5 whose cells are connected to the switch s_1 and s_5, respectively. Since P_1 is a neighbor cluster of P_5, there is a backbone link connects switches s_5 and s_1 in H. Define that a sequence of switches is a *path*. For each pair of neighboring switches in the path, there is a *backbone link* between the two corresponding clusters. Therefore, any established call among clusters could be represented by a path in the corresponding graph [2].

For the switch s_k, $k = 1, 2, \ldots, m$ in S and (s_k, s_l) in F, let d_{kl} be the minimal communication cost on the path between the switch s_k, and s_l. If cells c_i and c_j are assigned to different switches, then a (intra-switch) handoff cost is incurred. Assume for each cell c_i in C, cell c_i has been connected to a unique switch $sid(c_i)$, that is, $sid(c_i) = k$ if $c_i \in C$ is assigned to the switch s_k, k is called the *sid* of cell c_i.

With the given graph H, a backbone network $G(S, E, z)$ could then be built. The backbone network is built by utilizing the available edges in graph H. Typically, some limitation is placed on the number of links that could to be laid in the backbone network as the cost of the network is proportional to the number of links that are to be set up in the network. The objective is to determine the link between switches so as to minimize the handoff costs per unit time under the degree constraint. Let *deg*(*s*) denote the degree of switch *S* and *deg*(*G*) denote the degree of graph *G* which is the maximum degree of the switches in *G*. Thus $deg(G) = \max\{deg(s) \mid s \in S\}$. For example, given the graph shown in Fig. 1 (b), the corresponding graph of a possible backbone network with 10 links and $deg(G) = 3$ is shown in Fig. 1 (c).

2.2 Problem Formulation

To formulate the location update (or handoff) cost, variables h_{ij} $i, j = 1, 2, \ldots, n$ takes a value of 1, if both cells c_i and c_j are connected to a common switch; 0, otherwise. That is, $h_{ij} = 1$ if and only if $sid(c_i) = sid(c_j)$. With this definition, it is easy to see that the cost of handoffs per unit time is given by

$$\sum_{i=1}^{n}\sum_{j=1}^{n}(1-h_{ij})w_{ij} \times d_{sid(c_i),sid(c_j)}.$$

Given *m* nonempty sets of cells $P = \{P_1, P_2, \ldots, P_m\}$, *P* is called an *m*-way *cell partition* of *CG*, if $P_1 \cup P_2 \cup \ldots \cup P_m = CG$ and $P_k \cap P_l = \emptyset$, where $k \neq l$, $k, l = 1, 2, \ldots, m$. Without loss of generality, assume the cells in set P_k are assigned to switch s_k, $k = 1, 2, \ldots, m$. To calculate the overall handoff cost of a cell c_i to the set of cells assigned to switch s_l, let $LUCS(i, l) = \sum_{c_j \in P_l} w_{ij}$, if c_i not in P_l; $LUCS(i, l) = 0$, otherwise. Then for a given *m*-way cell partition *P*, the location update cost of the partition can be represented as

$$\sum_{c_i \in CG}\sum_{s_l \in S} LUCS(i, l) \times d_{sid(c_i),l}.$$

Because the assignments of cells to switches are fixed and known, the location update cost between switches is also fixed. To calculate the overall handoff cost between the set of cells c_i assigned to switch s_k and the set of cells assigned to switch s_l, let $LUSS(k, l) = \sum_{c_j \in P_k} LUCS(i, l)$, if $k \neq l$; $LUSS(k, l) = 0$, otherwise. Then the location update cost can be represented as $\sum_{k=1}^{m}\sum_{l=1}^{m} LUSS(k, l) \times d_{kl}$. In this paper, the designing problem of allocating the backbone links among the ATM switches with the objective that minimizes the cost of location update under the degree constraint is studied. With the above definition, the *wireless ATM backbone network design* (*WABND*) problem defined in [7] is showed as follows:

Wireless ATM backbone network design (WABND) problem [7] Given a graph $H = (S, F, z)$ with $|S| = m$, the $m \times m$ matrix *LUSS*, and positive integers *deg* and *p*, $deg \leq |S|$ and $|S| - 1 \leq p \leq |F|$, the WABND problem is to find a connected subgraph $G(S, E, z)$ of *H* with $|E| = p$, such that the location update cost $\sum_{k=1}^{m}\sum_{l=1}^{m} LUSS(k, l) \times d_{kl}$ is minimized and satisfies the constraint $deg(G) \leq deg$, where d_{kl} is the minimal communication cost between s_k and s_l on $G(S, E, z)$.

2.3 ILP Formulation

The WABND problem can be formulated as integer program. Define $F_{ij} = 1$, if the link (i, j) between switches s_i and s_j is in graph H; $F_{ij} = 0$, otherwise. Note that, $\sum_{i=1}^{m}\sum_{j=1}^{m} F_{ij} = |F|,$ and the value F_{ij} is known and fixed. Let $x_{ij} = 1$, if the link (i, j) between switches s_i and s_j is selected as a backbone link; $x_{ij} = 0$, otherwise. Let z_{ij} be the cost of the link (i, j) in F. Let $y_{ij}^{kl} = 1,$ if the shortest path between switches s_k and s_l pass the link (i, j); $y_{ij}^{kl} = 0$, otherwise.

The following is an integer programming formulation of the problem:

$$Minimize: \sum_{k=1}^{m}\sum_{l=1}^{m} LUSS(k,l) \times d_{kl}. \tag{1}$$

Thus the minimal communication cost d_{kl} between switches s_k and s_l is defined as

$$d_{kl} = \min \sum_{i=1}^{m}\sum_{j=1}^{m} z_{ij} y_{ij}^{kl}, \forall k, l = 1, 2, ..., m, \tag{2}$$

$$\sum_{\forall i} y_{ij}^{kl} - \sum_{\forall q} y_{jq}^{kl} = \begin{cases} -1, & \text{if } j = k \\ 1, & \text{if } j = l \\ 0, & \text{otherwise,} \end{cases} \tag{3}$$

$$y_{ij}^{kl} \le x_{ij}, \forall k, l = 1, 2, ..., m, \tag{4}$$

$$y_{ij}^{kl} \in \{0, 1\}, \forall k, l, i, j = 1, 2, ..., m, \tag{5}$$

$$\sum_{i=1}^{m}\sum_{j=1}^{m} x_{ij} = p, \tag{6}$$

$$\sum_{i=1}^{m} x_{ij} \le deg, \forall j = 1, 2, ..., m, \tag{7}$$

$$x_{ij} \le F_{ij}, \forall i, j = 1, 2, \ldots, m, \tag{8}$$

$$x_{ij} \in \{0, 1\}, \forall i, j = 1, 2, \ldots, m. \tag{9}$$

The backbone network must be a connected graph, the reachability from one switch to any switch is ensured this way. The graph G is partitioned into two connected parts, X and Y. Obviously there must be at least an edge going from X to Y for each feasible solution. Otherwise, the switches in Y cannot be reached. The reachability of graph G is stated as $\sum_{i \in X, j \in Y} x_{ij} \ge 1, \forall \text{partition } X \text{ and } Y \text{ of } G.$ There are 2^m ways to partition a graph, so the number of constraints is in the order of 2^m. This constraint is satisfied by all feasible solutions, and vice versa, any solution in which there exist unreachable switch must violate at least one of them.

2.4 Related Works

Given the PCN, handoff frequencies of cells in PCN, the connected ATM backbone network, and the capacities of switches, Merchant and Sengupta [4] formulated the *cell assignment problem* and considered the problem of assigning cells to switches (determined local links) in wireless ATM network. Recently, several extended studies of the cell assignment problem have been explored in [5, 6] to find the optimal solution of the cell assignment problem in the extended two-level wireless-ATM environment. In [5], a solution model which consists of three phases (*Cell Pre-partitioning Phase*, *Cell Exchanging Phase*, *and Cell Migrating Phase*) is proposed. In [6], the simulated annealing technique was applied.

In [2], Huang and Wang investigated the problem of allocating the backbone links among the ATM switches with the objective of reducing the effect of terminal mobility on the performance of wireless PCN's. The problem, which is called the *Link Allocation Problem* (*LAP*) on ATM-based PCN, is known NP-complete [2]. Formally, given a graph $H(S, F, z = 1)$ and positive integers *deg* and p, $deg \le |S|$ and $|S| - 1 \le p \le |F|$, the LAP is to find a connected subgraph $G = (S, E, z = 1)$ of H with $|E| = p$, such that the *diameter* (= $\max_{\forall s_k, s_l \in S}\{d_{kl}\}$) of the graph G is minimized and $deg(G) \le deg$. Two efficient heuristic algorithms were proposed in [2, 9] to solve the LAP.

The *Optimal Communication spanning Tree* (*OCT*) problem [10] is similar to LAP problem. The difference to the LAP is that the constructed graph is a tree ($p = |S| - 1$) without degree constraint ($deg = \infty$) and the goal is to minimize the summation of the weighted distance. The OCT problem is defined as follows. Let $H(S, F, z)$ be an undirected graph with nonnegative edge length function z. Given the requirements $\lambda(s_k, s_l)$ for each pair of nodes s_k and s_l. For any spanning tree T of H, the communication cost of T is defined as $\sum_{\forall s_k, s_l} \lambda(s_k, s_l) \times d_{kl}$. The goal of OCT is to construct a spanning tree T with the minimal communication cost. Like other constrained spanning tree problem, OCT problem is NP-hard [11].

For a given network, to find the best backbone topology of a network, the well-known problem is the *Degree Constrained Minimum Spanning Tree* (*DCMST*) problem. For a given graph, DCMST is the problem of generating a minimum cost spanning tree with degree constraints. The difference to the LAP is that the objective cost is the summation of the edge cost of the tree. By reducing it to an equivalent symmetric TSP, Garey and Johnson [11] showed the DCMST problem is NP-hard. Several studies about DCMST on complexity, performance of approximation algorithms and worst case performance guarantees were described in [12-14]. In [7], a genetic algorithm was proposed to solve the WABND problem. The Prüfer number was used in [7] to construct the degree- constraint tree of the network and then augmented it to the backbone consists of p links. Since the OCT problem is a special case of the WABND problem, therefore the WABND is NP-hard [7]. That is, finding an optimal solution for it is impractical due to exponential growth in execution time.

2.5 Contributions

In this paper, following the problem formulation defined in the previous result [7], two heuristic algorithms named *remove-based heuristic* (*RBH*) and *weight-median-based* *heuristic* (*WMBH*) are proposed to solve the WABND problem. Moreover, a hybrid method which combines the proposed heuristic algorithms and the well-known simulated annealing technique is proposed to find the near-optimal solution. Experiments are also given to show the efficiency of the proposed methods.

3. HEURISTIC ALGORITHM FOR WABND

In this section, two heuristic algorithms are proposed to solve the WABND problem, they are *remove-based heuristic* (RBH) and *weight-median-based heuristic* (WMBH) algorithms.

3.1 Remove-Based Heuristic (RBH)

Given network $H(S, F, z)$, since $p \le |F|$, $|F| - p$ edges should be removed from $H(S, F, z)$ to construct the backbone network $G(S, E, z)$. Define *bridge edge* be an edge whose remove disconnected $H(S, F, z)$. After removing edges from $H(S, F, z)$, the graph $G(S, E, z)$ should be a connected and constraint-satisfied (degree-constraint and edge-constraint) graph. Obviously, bridge edges in $H(S, F, z)$ can not be removed. Moreover, if the connected components of graph $H(S, F, z)$ are given, bridge edge is the edge which connects different components. Obviously, the connected-component finding algorithm (in $O(|F| + |S|)$ time) and a sequential testing procedure (in $O(|F|)$ time) can be used to find the set BE_H of bridge edges in $H(S, F, z)$. Let $CC_H = \{CC_1, CC_2, \ldots, CC_q\}$ be the set of the connected components of the $H(S, F, z)$.

Let $degree_H(s_k)$ be the degree of switch s_k in $H(S, F, z)$. For each switch s_k, define $CON_H(s_k) = 1$, if $degree_H(s_k) > deg$; $CON_H(s_k) = 0$, otherwise. For each edge $e = (s_k, s_l)$ in CC_H, define $CON_H(e) = CON_H(s_k) + CON_H(s_l)$. Moreover, edges in $F - BE_H$ are partitioned into three groups $FREE_H$, $CON1_H$, $CON2_H$ according to the CON_H value of edges. That is, $FREE_H$, $CON1_H$, and $CON2_H$ are the set of edges in CC_H, whose CON_H value is equal to 0, 1, and 2, respectively.

Define $opt(H')$ be the object cost of the network H', that is, $opt(H') = \sum_{k=1}^{m} \sum_{l=1}^{m} LUSS(k, l) \times d_{kl}$, where d_{kl} is the shortest distance between switches s_k and s_l in H'. Initially, let $H'(S, F', z)$ be the same as $H(S, F, z)$. A non-bridge edge $e = (s_k, s_l)$ is found in ($CON2_{H'}$, $CON1_{H'}$ or $FREE_{H'}$) which minimize $\Delta = opt(H' - e) - opt(H')$, $\forall e \in CON2_{H'}$. That is, if $CON2_{H'}$ is nonempty, find an edge $e = (s_k, s_l)$ in $CON2_{H'}$; otherwise if $CON1_{H'}$ is nonempty find an edge $e = (s_k, s_l)$ in $CON1_{H'}$; otherwise find an edge $e = (s_k, s_l)$ in $FREE_{H'}$.

Then, remove e from H'. Since edge e should be in $CC_i \subseteq CC_{H'}$ for some i, remove e from H' may make the CC_i change. The component CC_i may be divided into several smaller connected components and some new bridge edges may be generated. These may change $BE_{H'}$ and $CC_{H'}$; moreover, the sets $FREE_{H'}$, $CON1_{H'}$ and $CON2_{H'}$ should be updated. The edge-removing process is performed repeatedly until $H'(S, F', z)$ contains exact p edges. Clearly, if there exists a feasible solution, then after performing the edge-removing processes, all edges (include bridge edges) should be in $FREE_{H'}$. The details of the RBH algorithm are stated as follows:

Remove-Based Heuristic (RBH)

Let $H'(S, F', z) = H(S, F, z)$.

Perform Connected-component finding algorithm to find the $CC_{H'} = \{CC_1, CC_2, \ldots, CC_q\}$ of $H'(S, F', z)$. Group those edges which connected two different connected components into $BE_{H'}$.

Determine the associated group ($FREE_{H'}$, $CON1_{H'}$, or $CON2_{H'}$) of edge e in $F - BE_{H'}$ according to the value of $CON_{H'}(e)$.

```
while (|F'| > p) do
    begin
    if (CON2_H' ≠ ∅)
        then
            Select and edge e = (s_k, s_l) in CON2_H' which minimizes opt(H' − e) − opt(H'), ∀e ∈ CON2_H';
        else if (CON1 ≠ ∅)
            then
                Select an edge e in CON1_H' which minimizes opt(H' − e) − opt(H'), ∀e ∈ CON1_H';
            else
                Select an edge e in FREE_H' which minimizes opt(H' − e) − opt(H'), ∀e ∈ FREE_H';

        Remove e from H' (S, F', z).
        Update degree_H'(s_k), degree_H'(s_l), CC_H', BE_H', FREE_H', CON1_H', CON2_H'.
    end    //end of while

if ((CON1_H' ≠ ∅) or (CON2_H' ≠ ∅))
    then
        return notfound.
    else
        return H' (S, F', z).
```

3.2 Weight-Median-Based Heuristic (WMBH)

The backbone network of the wireless-ATM can also be constructed by first finding a spanning tree of the network which satisfies the degree constraint and then augmenting the tree to consist p links. To find the spanning tree of a given graph, the shortest path algorithm is used. But the cost of the spanning tree may vary if the source node is changed. Since the objective cost of the WABND problem is weighted-distance based, the *median* node with the minimal summation of weighted distance to all nodes may be the best choice. Thus, in the proposed weight-median-based heuristic, first, the weighted median is found; then the degree-constrained spanning tree rooted at median is constructed; finally, the spanning tree is augmented to form the backbone of the ATM network. For an undirected network $H(S, F, z)$ with m nodes, choose a vertex s_x on the graph H such that $\sum_{k=1}^{m} LUSS(k, x) \times d_{kx}$ is minimized. The location of x is called the 1-*weight-median* of the network $H(S, F, z)$. To find the 1-weight-median of the network, first, Floyd's algorithm is used to find the shortest distance matrix $[d_{kl}]_{m\times m}$ for all pairs of vertices, s_k and s_l, of the network $H(S, F, z)$. Next the terms $LUSS(k, x) \times d_{kx}$ is computed by multiplying each element of the distance matrix $[d_{kl}]_{m\times m}$ by the weight of corresponding element in matrix LUSS. Third, for each row (or column) the sum is computed. The node that corresponds to the row with the minimum sum of terms is the location for the l-weight-median.

1-Weight-Median Algorithm
Perform Floyd's algorithm to find the shortest distance matrix $[d_{kl}]_{m\times m}$ for the nodes of H.
Let $sum_k = 0$, for $k = 1, 2, \ldots, m$ represents the sum of row k.
for $k = 1$ **to** m
  for $l = 1$ **to** m
    $sum_k = sum_k + LUSS(k, l) \times d(k, l)$
Find the node s_x which minimizes sum_x.

Then, the famous Dijkstra's algorithm is modified to find shortest-path based degree-constrained spanning tree. The details of algorithm are shown as follows:

Degree-constrained-tree (source *s*)
$V = \{s\}$, $N = S - V$, $T = \emptyset$;
while $(S - V \neq \emptyset)$ **begin**
  for all links (u, v), $u \in V$ and $v \in S - V$ **begin**
    find the link (k, l) such that $deg(T \cup (k, l)) \leq deg$ and with smallest cost
    $z_{kl} \leq \min_{\forall u \in V, v \in S-V}\{z_{uv}\}$
    if (found)
     then $\{T = T \cup (k, l);\ V = V \cup \{l\};\ N = S - V\}$;
    else return *notfound*;
  end
end
return T;

Finally, the shortest-path based degree-constrained spanning trees $T_x = (S, E_1, z)$ of $H(S, F, z)$ with 1-weight-median s_x as root is found by performing the *degree-constrained algorithm* [7] (as shown above). Define $opt(T)$ be the object cost of the spanning tree T, that is, $opt(T) = \sum_{k=1}^{m}\sum_{l=1}^{m} LUSS(k,l)\times d_{kl}$, where d_{kl} be the shortest distance between switches s_k and s_l in T. Finally, a set of $p - |E_1|$ edges are added in phase two to form the graph G. The order of edges to be added to G, is determined by a greedy scheme. For each link e in $F - E_1$, compute $\Delta = opt(T_x) - opt(T_r \cup e)$. Sort links in $F - E_1$ in non-descending order according to the value of Δ. The detail of the weight-median-based heuristic algorithm is shown as follows.

Weight-Median-Based Heuristic (WMBH)
Call 1-*Weight-Median Algorithm* to find the 1-weight-median s_x of $H(S, F, z)$.
Call *degree-constrained-tree*(s_x) to find the spanning tree T_x of H with root s_x.
Let $T_x = (S, E_1)$, $G = T_x$.
// augment-edge phase
for each link e in $F - E_1$ do
  {
  Compute $\Delta = opt(T_x) - opt(T_x \cup e)$;
  Sort links in $F - E_1$ in non-descending order according to value Δ.

```
    }
Assume links are re-indexed as {e_1, e_2, e_3, …, e_{|F|-m+1}};
for i = 1 to |F − E_1| do
    if ((deg(G ∪ e_i) ≤ deg) and (|G ∪ e_i| ≤ p))
        then G = G ∪ e_i;

if (|G| ≠ p)
then
    return not found;
else
    return G;
```

4. SIMULATED ANNEALING ALGORITHM FOR WABND

Due to the complexity of the WABND problem, the provision of an optimal solution in reasonable time is not guaranteed. Moreover, heuristic algorithms may trap into local optimal. 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 WABND problem in this section. In this section, the details of simulated annealing algorithm developed to solve the WABND problem are present.

4.1 Configuration Space

The objective of WABND problem is to find an optimal backbone network topology so that the object function value is minimized under the connected and degree-constraint. To do this, the configuration space is designed to be the set of possible solutions. Each configuration is defined as a topology network which is represented by *adjacency matrix*.

4.2 Initial Configuration Generation

The initial configuration of the simulated annealing algorithm is generated by performing one of the heuristic algorithms described in section 3.

4.3 Cost Function

Generally, simulated annealing algorithms use the cost function to achieve the goal of finding optimally assignments. The goal is to minimize the total cost of the wireless ATM network. For the given network topology, the all-pair shortest path algorithm is used to find the distance d_{kl} between switches s_k and s_l. Thus, the cost function is $\sum_{k=1}^{m}\sum_{l=1}^{m} LUSS(k, l)\times d_{kl}$. The generated topology is constraint-satisfied, that is, there is no need to attach the penalty cost.

4.4 Perturbation Mechanism

Simulated Annealing algorithm uses perturbation to change the configuration from one state to another state. In this subsection, the perturbation methods used for designing the topology of backbone network are developed. The resulted configuration is still a constraint-satisfied one after performing perturbation. The basic idea of the perturbation is *edge–exchange*, that is, an edge in current topology is removed and the other edge is added into the topology to form a new configuration. The order and the method of determining edge-addition or edge-deletion do affect the result of configuration (as showed in Table 1). In the following, several types of perturbations are introduced into the SA algorithm for solving the WABND problem.

Table 1. Perturbation schemes.

ADD-First		
Method	Random ADD	Best ADD
Random DELETE	RARD	BARD
Best DELETE	RABD	BABD
Delete-First		
Method	Random ADD	Best ADD
Random DELETE	RDRA	RDBA
Best DELETE	BDRA	BDBA

Let $G(S, E, z)$ be the network topology in current configuration. Recall that $degree_G(s_k)$ be the degree of switch s_k in $G(S, E, z)$. For each switch s_k, define $CDG_G(s_k) = 1$, if $degree_G(s_k) = deg$; $CDG_G(s_k) = 0$, otherwise. For each edge $e = (s_k, s_l)$ in $F - E$, define $CDG_G(e) = CDG_G(s_k) + CDG_G(s_l)$. For a given graph G, define $FREE_G$, $CDG1_G$, and $CDG2_G$ be the set of edges, whose CDG_G value is equal to 0, 1, and 2, respectively. Let $UNFREE_G$ be the set of edges whose CDG_G value is greater than 1.

Edges in $F - E$ are partitioned into three groups $FREE_G$, $CDG1_G$, $CDG2_G$ according the to the CDG_G value of edges. If an edge e in $CDG2_G$ is selected and added into $G(S, E, z)$. Then after removing an arbitrary edge other than e, the resulted configuration will not be a constraint-satisfied one. Thus, only those edges in $FREE_G$ or $CDG1_G$ can be selected and added to $G(S, E, z)$. Let $G'(S, E', z) = G(S, E, z) \cup e$.

To remove an edges from $G'(S, E', z)$, the resulted graph should be connected and constraint-satisfied. Thus bridge edges in $G'(S, E', z)$ can not be removed. Let $BE_{G'}$ be the set of bridge edges in $G'(S, E', z)$. Edges in $G' - BE_{G'}$ are partitioned into two groups $FREE_{G'}$ and $CDG1_{G'}$ according to the $CDG_{G'}$ value of edges.

Eight types of perturbations are introduced into the simulated annealing algorithm and shown as follows. The following perturbation schemes are: first added an edge into $G(S, E, z)$, then deleted an edge to construct the new configuration.

- **P1:** *random add random delete* (*RARD*): Randomly select an edge e_{in} in $FREE_G \cup CDG1_G$ and add to $G(S, E, z)$. If $e_{in} \in FREE_G$ then random select an edge e_{out} in $FREE_{G'}$ and delete. If $e_{in} \in CDG1_G$ and s_k be the switch with $degree(s_k)_{G'} > deg$ then random select an edge e_{out}, one of the endpoint of e_{out} is s_k and e_{out} in $CDG1_{G'}$. Delete the edge e_{out} form $G'(S, E', z)$ to generate the new configuration.
- **P2:** *random add best delete* (*RABD*): Randomly select an edge e_{in} in $FREE_G \cup CDG1_G$ and add to $G(S, E, z)$. If $e_{in} \in FREE_G$ then find the edge e_{out} which minimizes $opt(G'(S, E', z) - e_{out}) - opt(G'(S, E', z))$; if tie, random select one of them. If $e_{in} \in CDG1_G$ and s_k

be the switch with $degree_{G'}(s_k) > deg$ then find the edge e_{out} in $CDG1_{G'}$ which minimizes $opt(G'(S, E', z) - e_{out}) - opt(G'(S, E', z))$. Delete the edge e_{out} form $G'(S, E', z)$ to generate the new configuration.

- **P3:** *best add random delete* (*BARD*): Select an edge e_{in} in $FREE_G \cup CDG1_G$ which maximizes $opt(G(S, E, z)) - opt(G(S, E, z) \cup e_{in})$; if tie, random select one of them and add to $G(S, E, z)$. If $e_{in} \in FREE_G$ then random select an edge e_{out} in $FREE_{G'}$. If $e_{in} \in CDG1_G$ and s_k be the switch with $degree_{G'}(s_k) > deg$ then random select an edge e_{out}, one of the endpoint of e_{out} is s_k and e_{out} in $CDG1_{G'}$. Delete the edge e_{out} form $G'(S, E', z)$ to generate the new configuration.
- **P4:** *best add best delete* (*BABD*): Select an edge e_{in} in $FREE_G \cup CDG1_G$ which maximizes $opt(G(S, E, z)) - opt(G(S, E, z) \cup e_{in})$; if tie, random select one of them and add to $G(S, E, z)$. If $e_{in} \in FREE_G$ then random select an edge e_{out} in $FREE_{G'}$. If $e_{in} \in CDG1_G$ and s_k be the switch with $degree_{G'}(s_k) > deg$ then find the edge e_{out} in $CDG1_{G'}$ which minimizes $opt(G'(S, E', z) - e_{out}) - opt(G'(S, E', z))$. Delete the edge e_{out} form $G'(S, E', z)$ to generate the new configuration.

The following perturbation schemes are: first deleted an edge from $G(S, E, z)$, then add an edge to construct the new configuration. The resulted graph $G''(S, E'', z)$ which is obtained be removing an edge in $G(S, E, z)$ may be disconnected while the bridge edge is selected and removed. Let BE_G be the set of bridge edge in $G(S, E, z)$. Edges in G are divided into two groups BE_G and $E - BE_G$. After selecting and deleting e_{out}, if $G''(S, E'', z)$ is disconnected then $G''(S, E'', z)$ is divided into two connected components, the set of edges which connected two components and make $G''(S, E'', z) \cup e$ be a feasible solution of the WABND problem is denoted as $CONN_{G''}$.

If $G''(S, E'', z)$ is connected then edges in $F - E''$ are partitioned into two groups $FREE_{G''}$ and $UNFREE_{G''}$ according to the $CDG_{G''}$ value of edge e.

- **P5:** *random delete random add* (*RDRA*): Randomly select an edge e_{out} in $G(S, E, z)$ and delete. If $e_{out} \in BE_G$ then random select an edge e_{in} in $CONN_{G''}$ and add. Otherwise, random select an edge e_{in} in $FREE_{G''}$ and add.
- **P6:** *random delete best add* (*RDBA*): Randomly select an edge e_{out} in $G(S, E, z)$ and delete. If $e_{out} \in BE_G$ then find the edge $e_{in} \in CONN_{G''}$ which minimizes $opt(G''(S, E', z) \cup e_{in})$; if tie, random select one of them. Otherwise, find the edge e_{in} in $FREE_{G''}$ which minimizes $opt(G''(S, E', z)) - opt(G''(S, E', z) \cup e_{in})$; if tie, random select one of them and add.
- **P7:** *best delete random add* (*BDRA*): Select an edge e_{out} in $E - BE_G$ which minimizes $opt(G(S, E, z) - e_{out}) - opt(G(S, E, z))$; if tie, random select one of them and delete. Random select an edge e_{in} in $FREE_{G''}$ and add.
- **P8:** *best delete best add* (*BDBA*): Select an edge e_{out} in $e - BE_G$ which minimizes $opt(G(S, E, z) - e_{out}) - opt(G(S, E, z))$; if tie, random select one of them and delete. Find the edge e_{in} in $FREE_{G''}$ which minimizes $opt(G''(S, E', z)) - opt(G''(S, E', z) \cup e_{in})$; if tie, random select one of them and add.

Let P_i be the probability of transforming current configuration to a new one by applying the perturbation **Pi**, $i = 1, 2, \ldots, 8$, respectively. Assume that $\sum_{i=1}^{8} p_i = 1$. Let $AP_0 = 0$ and $AP_i = \sum_{j=1}^{i} p_j$ be the accumulated probability of p_i, $i = 1, 2, \ldots, 8$.

4.5 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. These parameters are described as follows.

(1) *Initial value of the control parameter*: The rule used in SA that the starting temperature c_0 is determined by calculating the average increasing in cost, $\overline{\Delta C}^{+}$, for 50 random transitions and solve c_0 from $c_0 = \overline{\Delta C}^{+} / ln(\mathcal{X}_0^{-1})$, where *accepted ratio* $\mathcal{X}_0$ defined as the number of accepted transitions divided by the number of proposed transitions. In this paper, the accepted ratio $\mathcal{X}_0$ is empirically set to 0.55.
(2) *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. The decrement rule in SA is defined as follows: $T_{k+1} = \gamma T_k$, where γ is empirically determined and discussed in section 5.1.
(3) *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.
(4) *The length of Markov Chains*: In this paper, the chain length is empirically determined and discussed in section 5.1.

4.6 Simulated Annealing Algorithm of WABND Problem

The details of the simulated annealing are described as follows:

Algorithm Simulated Annealing
Step 1: For a given initial temperature T, perform **RBH** or **WMBH** algorithm to generate initial configuration *IC*. The currently best configuration (*CBC*) is *IC*, *i.e.* *CBC* = *IC*, and the current temperature value (*CT*) is T, *i.e.*, *CT* = T. Determine p_i, $i = 1, 2, \ldots, 8$, $AP_0 = 0$ and $AP_i = \sum_{j=1}^{i} p_j$, $i = 1, 2, \ldots, 8$.
Step 2: If *CT* = 0 or the stop criterion is satisfied then go to step 7.
Step 3: Generate a random number p in [0, 1), if $AP_{i-1} \le p \le AP_i$, ($i = 1, 2, \ldots, 8$) then new configuration (*NC*) is generated by applying the *Pi* perturbation schema.
Step 4: The difference of the costs of the two configurations, *CBC* and *NC* is computed, *i.e.*, $\Delta C = E(CBC) - E(NC)$.
Step 5: If $\Delta C \ge 0$ then the new configuration *NC* becomes the currently best configuration, *i.e.*, *CBC* = *NC*. Otherwise, if $e^{-(\Delta C/CT)} >$ *random* [0, 1), the new configuration *NC* becomes the currently best configuration, *i.e.*, *CBC* = *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. SIMULATION RESULTS

In order to evaluate the performance of the proposed algorithms, these algorithms have been implemented and applied to solve several randomly generated examples. The results of these experiments are reported below. All the algorithms were implemented by *C*, and all experiments conducted on a personal computer (PC) with Pentium IV 2.8 GHz CPU and 512MB RAM. A hexagonal system in which the cells were configured as an H-mesh was used for simulation. Assume the antenna for each cell was at the center of the cells. Switches are located at the same position of cells which are randomly selected from the cells. The cabling cost between a switch and a cell was taken to be proportional to the geometric distance between the two. 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.

The experiment shown here is divided into two parts, the goal of the first part is to determine the control parameters of SA, and the goal of the second part is to examine the performance of the proposed simulated annealing algorithm and other solving methods.

5.1 Determining the Control Parameters

In the following, for each experiment, 10 runs are tested and the average result is computed. First, the effect of the various values of chain length is determined. For the experiments, chain length is in {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 20, 30, 40, 50} and the number of links (p) in backbone is in {40, 50, 60, 70, 80, 90}. The perturbation method is randomly selected from the eight perturbations with equal probabilities. The effect of the different chain lengths to the cost is shown in Fig. 2. In Fig. 2 only those cases whose chain length is less than or equal to 10 are shown since for the other cases (chain length greater than 10) the objective cost can not be reduced. Observe form Fig. 2, it is easily to find that SA can find best result if the chain length greater than 7.

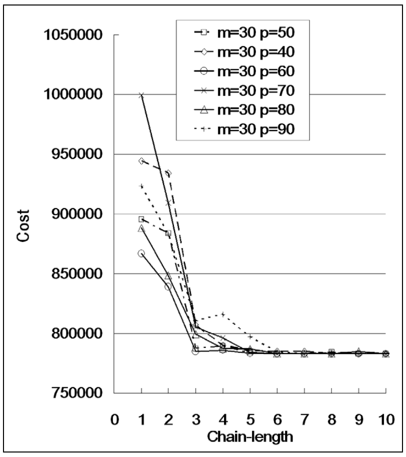


Fig. 2. The objective cost of the set of examples with m = 30 for different values of p and chain lengths.

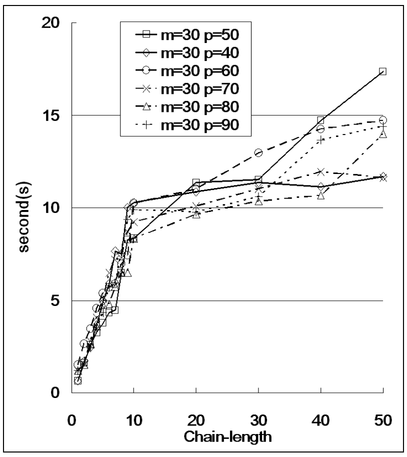
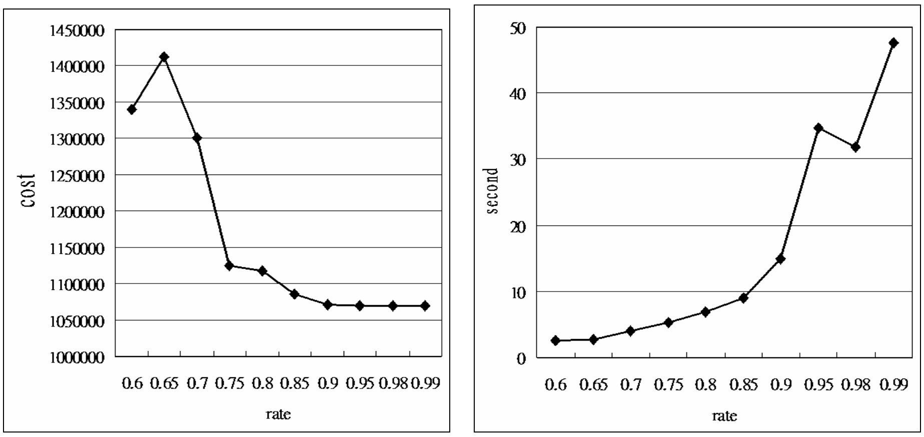


Fig. 3. The CPU time in seconds of the set of examples with m = 30, for different values of p and chain lengths.

The effect of the different chain lengths to the CPU time in seconds is shown in Fig. 3. The time shown in Fig. 3 is the time that the objective cost can not be reduced for the same value of γ (temperature decreasing rate). Observe form Fig. 3, it is easily to find that as the chain length increases the CPU time in seconds increases proportionally. As the chain length greater than 10, the curve is slow down.

The effect of different values of temperature decreasing rate γ to the cost and the CPU time in seconds is shown in Figs. 4 (a) and (b), respectively. The result in Fig. 4 (a) shows that SA with $\gamma \geq 0.9$ can get the best performance. The result in the Fig. 4 (b) shows that γ set to 0.9 is cost-effective.



(a) (b)

Fig. 4. Experiments on the set of examples with m = 40, p = 50, deg = 5, (a) objective cost for different values of γ, (b) CPU time is seconds for different values of γ.

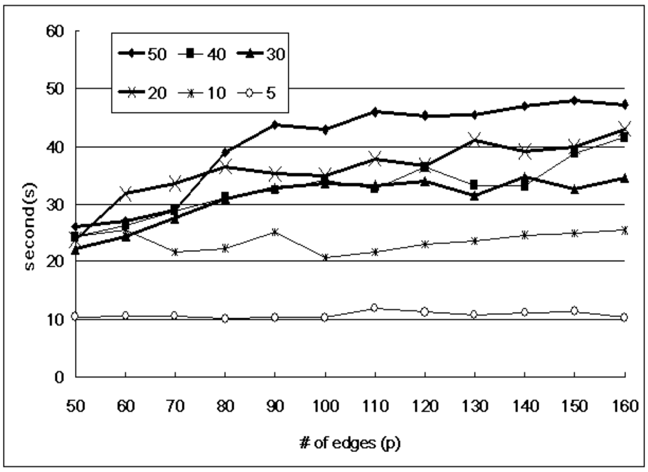


Fig. 5. Experiments on the set of examples with m = 40 for different values of p and chain lengths.

The effect of the number of edges (p) in backbone network to the chain length and CPU time in seconds is shown in Fig. 5. In this experiment, p is in {50, 60, 70, 80, 90, 100, 110, 120, 130, 140, 150, 160}. Observe the result shown in Fig. 5, it is easily to find that the CPU time in seconds increases proportionally as p increases. In most cases, it is easily to find that the CPU time in seconds increases proportionally as the chain length increases.

5.2 Performance of SA

Several examples were used to test the efficiency of the proposed algorithms. The mean result obtained by running ten times of simulated annealing algorithm (SA), genetic algorithm (GA) [7], and the result obtained by performing the heuristic algorithms RBH, WMBH, and HA (proposed in [7]) were examined. The parameters of the GA [7] were: crossover probability is 1.0, mutation probability is 0.3, and the population size is $5 \times (m + p)$, and the number of generations of the GA is 500. In the experiment of GA, since the population size is large enough, for most cases, the result of GA with generation greater than 500 can not be reduced further.

For the set of examples: $m = 20$, $deg = 5$, p is in {25, 30, 35, 40, 45}, the result is showed in Fig. 6 (a). For the set of examples: $m = 20$, deg is in {5, 6, 7, 8, 9, 10, 11, 12}, $p = 25$, the result is showed in Fig. 6 (b). For the set of examples: $m = 30$, $deg = 10$, p is in {40, 50, 60, 70, 80, 90}, the result is showed in Fig. 7 (a). If $p = |F|$, then result backbone network $G(S, E, z)$ should be $H(S, F, z)$. Let "LB" represent the cost obtained on the topology H, which is the lower bound of the WABND problem. For the set of examples: $m = 30$, deg is in {5, 10, 15, 20, 25, 30}, $p = 40$, the result is showed in Fig. 7 (b). Observe the results shown in Figs. 6 and 7, the proposed simulated annealing algorithm can get the best result as the same obtained by the genetic algorithm (GA) as indicated from the experimental results. Moreover, the proposed heuristic algorithms RBH and WMBH get better results than HA.

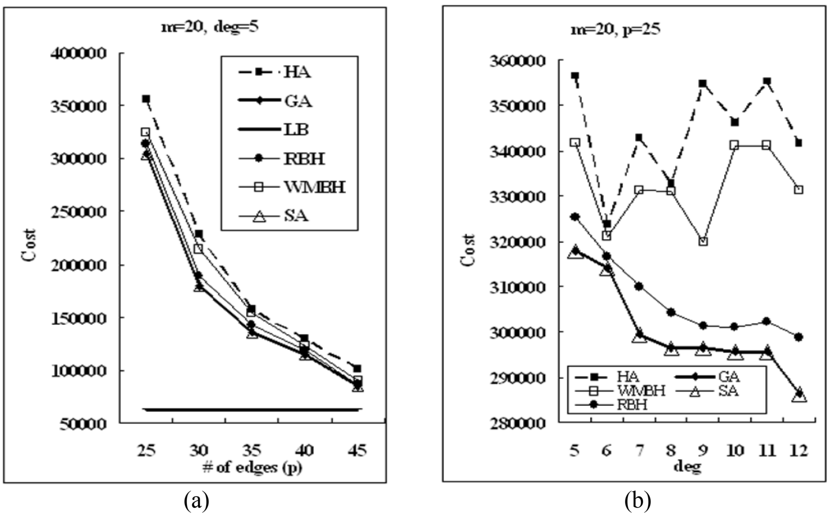
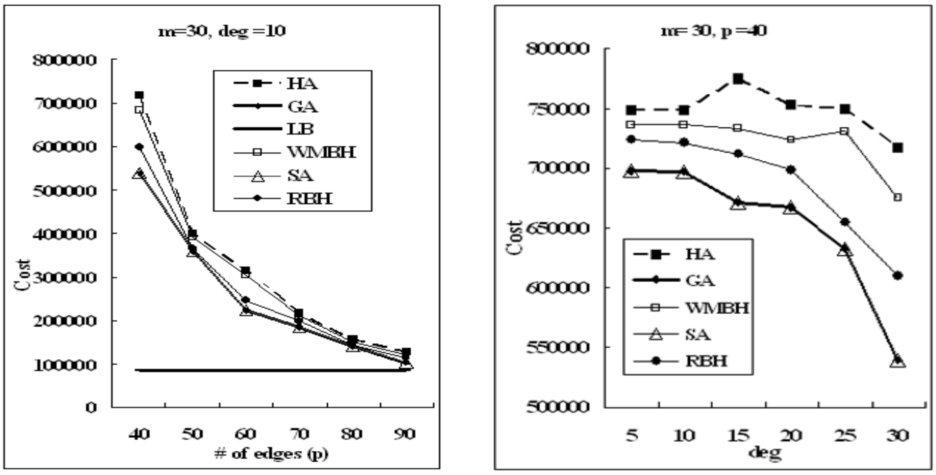


Fig. 6. Experiments on set of examples with $m = 20$, (a) different values of p, (b) different values of deg.



(a) (b)

Fig. 7. Experiments on set of examples with m = 30, (a) different values of p, (b) different values of *deg*.

Table 2. Comparison of SA, GA, and LB.

m=20	GA					SA					GA/SA
p	min	average	max	std	time	min	average	max	std	time	
25	304985	306075	309216	2197	12	304985	305586	311591	3653	5	100.00%
30	179483	182250	186192	3371	12	179483	180628	180759	702	5	100.00%
35	135602	135969	140594	2782	14	135602	136254	144463	4938	6	100.00%
40	114363	114775	114969	309	14	114363	115146	115522	591	6	100.00%
45	85106	87624	88251	1664	16	85106	85267	90665	3164	7	100.00%
m=30	GA					SA					GA/SA
p	min	average	max	std	time	min	average	max	std	time	
40	538963	540345	545346	3358	15	538963	540797	543051	2047	8	100.00%
50	359563	359648	361441	1060	15	359563	361020	365167	2907	8	100.00%
60	222179	223470	224239	1041	18	222179	222680	226904	2595	9	100.00%
70	182511	182828	188415	3321	18	182511	184257	186921	2221	9	100.00%
80	139605	141188	142015	1224	21	139605	141423	143593	1997	10	100.00%
90	103744	105642	104082	1012	23	103744	103756	106732	1721	10	100.00%
m=50	GA					SA					GA/SA
p	min	average	max	std	time	min	average	max	std	time	
100	1085053	1086720	1094830	5230	213	914023	915689	918148	2075	123	118.71%
150	381691	382115	384157	1318	254	336654	337614	341961	2828	156	113.38%
200	217325	217435	224833	4303	312	185169	186014	188475	1717	213	117.37%
m=100	GA					SA					GA/SA
p	min	average	max	std	time	min	average	max	std	time	
150	17643700	17645311	17653089	5021	521	15643700	15644674	15652395	4763	324	112.78%
200	7853825	7855918	7859129	2671	545	7129875	7131519	7134910	2567	354	110.15%
250	4553825	4554008	4556586	1544	576	4413595	4413833	4418169	2575	367	103.18%
300	3353825	3355944	3358945	2572	589	3241032	3241578	3246105	2784	386	103.48%
350	2453825	2454091	2460217	3616	643	2333048	2333539	2333994	473	401	105.18%
400	2353825	2354803	2358067	2221	754	2233048	2234120	2240100	3800	413	105.41%
average											106.70%

To know the effect of GA and SA in depth, the SA and GA were run with 10 random seeds on each problem in order to get some statistical information about the quality of their solutions. The result is summarized in Table 2. In this table, the heading of "GA" refers to the objective value obtained by performing genetic algorithm in [7]. The heading of "SA" refers to the objective value obtained by performing simulated annealing algorithm. The heading of "time" refers to the CPU time in seconds spent by performing algorithm. The SA and GA columns of Table 2 show the minimum, average, maximum, and standard deviation of the cost of solutions for 10 runs.

The result shows that simulated annealing and genetic algorithms obtained same results for the cases $m = 20$ and $m = 30$. For the other cases ($m = 50$, $m = 100$), the proposed simulated annealing algorithm can get better result than GA. Take the value of SA as a reference, it is easy to compute the result of the GA to that of the SA, the ratio is 106.70% on average. The result of SA and GA in Table 2 shows that SA has better performance than GA in objective cost and CPU time in seconds.

6. CONCLUSIONS

In this paper, the problem of optimum design of the two-level wireless ATM network is investigated. Given PCN network, handoff frequencies between cells, locations of switches on an ATM network, the number of backbone links, the communication cost between switches, and the degree constraint of switch, the problem is to determine the topology of the backbone network in an optimum manner and named as the *wireless ATM backbone network design problem* (*WABND*).

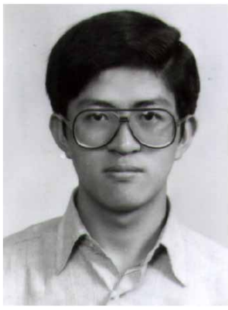
First, the formulation of the WABND problem is given; since the optimal communication spanning tree (OCT) problem is a special case of WABND which is NP-hard, thus the WABND is NP-hard. Thus finding an optimal solution of this problem in reasonable time is impractical. In this paper, two heuristic algorithms (RBH and WMBH) and a simulated annealing algorithm (SA) are proposed to solve this problem. Simulation results showed that simulated annealing algorithm is robust for this problem.

In the SA method, adjacency matrix is used to represent the topology of the backbone network. In the design encoding method, the configuration represents a connected graph witch is a constraint-satisfied one. Thus there is no need for penalty function and thus the performance of SA can be improved. Experimental results indicate that the proposed SA runs efficiently. The total cost of SA gets better performance than heuristic algorithms.

REFERENCES

1. R. Steele, "Deploying personal communication networks," *IEEE Communications Magazine*, Vol. 4, 1990, pp. 12-15.
2. N. F. Huang and R. C. Wang, "The link allocation problem on ATM based personal communication networks," *Journal of Wireless Personal Communications*, Vol. 4, 1997, pp. 257-275.
3. D. Raychaudhuri and N. D. Wilson, "ATM-based transport architecture for multiservices wireless personal communication networks," *IEEE Journal on Selected Areas in Communications*, Vol. 12, 1994, pp. 1041-1414.
4. A. Merchant and B. Sengupta, "Assignment of cells to switches in PCS networks," *IEEE/ACM Transactions on Networking*, Vol. 3, 1995, pp. 521-526.
5. D. R. Din and S. S. Tseng, "A solution model for optimal design of two-level wireless ATM network," *IEICE Transactions on Communications*, Vol. E85-B, 2002, pp. 1533-1541.

6. D. R. Din and S. S. Tseng, "Heuristic and simulated annealing algorithms for solving extended cell assignment problem in wireless ATM network," *International Journal of Communication Systems*, Vol. 15, 2002, pp. 47-65.
7. D. R. Din, "Wireless ATM backbone network design problem," *IEICE Transactions on Fundamentals of Electronics*, *Communications and Computer Sciences*, Vol. E88-A, 2005, pp. 1777-1785.
8. S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, Vol. 220, 1983, pp. 671-680.
9. C. P. Low, "An efficient algorithm for the link allocation problem on ATM-based personal communication networks," *IEEE Journal on Selected Areas in Communications*, Vol. 18, 2000, pp. 1279-1288.
10. T. C. Hu, "Optimum communication spanning tree," *SIAM Journal on Computing*, Vol. 3, 1974, pp. 188-195.
11. M. R. Garey and D. S. Johnson, *Computers and Intractability – A Guild to the Theory of NP-Completeness*, W. H. Freeman, San Francisco, 1979.
12. B. Boldon, N. Deo, and N. Kumar, "Minimum-weight degree-constrained spanning tree problem: heuristics and implementation on an SIMD parallel machine," *Parallel Computing*, Vol. 22, 1996, pp. 369-382.
13. M. Krishnamoorthy, A. T. Ernst, and M. S. Yazid, "Comparison of algorithms for the degree constrained minimum spanning tree," *Journal of Heuristics*, Vol. 7, 2001, pp. 587-611.
14. N. Deo and N. Kumar, "Computation of constrained spanning trees: a unified approach," *Network Optimization: Lecture Notes in Economics and Mathematical Systems*, 1997, pp. 194-220.



Der-Rong Din (丁德榮) received M.S. and Ph.D. degrees in Computer and Information Science from National Chiao Tung University in 1993 and 2001, respectively. From 2001 to 2003, he was the Assistant Professor of the Department of Computer Science and Information management at Hung Kuang University. Now, he is on the faculty of Department of Computer Science and Information Engineering at National Changhua University of Education. His current research interests include in WDM networking, mobile communication, and algorithm.