Fault-tolerant Wireless Access Network Design for Dual-homed Users

Xiaodong Huang[†], Jianping Wang[‡], Vinod M. Vokkarane[*], and Jason P. Jue[†]
[†]Department of Computer Science, The University of Texas at Dallas, Richardson, TX 75083
[‡]Computer Science Department, Georgia Southern University, Statesboro, GA 30460
[*]Computer and Information Science Department, The University of Massachusetts Dartmouth, MA 02747
Emails: {xxh020100@utdallas.edu, jpwang@georgiasouthern.edu, vvokkarane@umassd.edu, jjue@utdallas.edu}

Abstract— **In this paper, we study the survivability problem in hierarchical wireless access networks with dual-homed end users, who are connected to two base stations (BSs), a primary BS and a backup BS. The dual homing mechanism is resilient to a single failure of a BS. However, if a failure occurs at the base station controller (BSC) layer or at the mobile switching center (MSC) layer, dual-homing may not prevent connection loss. We address the problem of routing from BSs to BSCs and from BSCs to MSCs, with the objective of minimizing the maximum number of connections lost due to a single failure of BS, BSC, or MSC. We first formulate the problem using Integer Linear Programming (ILP). We then prove that this optimization problem is NP-hard by showing some of its subproblems with relaxed constraints are still NP-hard. A Tabu Search (TS) based heuristic is then proposed for the problem, which provides near optimal results in most cases.**

I. Introduction

Rapid growth in Internet traffic, a move towards enhanced mobility, and an increasing dependence on networks for supporting mission-critical applications will require the development of wireless broadband networks that are not only able to support the increasing bandwidth requirements of mobile users, but also able to provide reliable connectivity to these users.

Research and development on the survivability of networks has largely focused on fixed networks, with little attention on the survivability of wireless access networks. A wireless access network usually covers a large geographical service area partitioned into many small regions called cells. Each cell is served by a base station (BS) that serves as a fixed access point for mobile users within the cell. Generally, we assume BSs are fixed, although some work can be found to address the movement of mobile base stations [1]. The network may include base station controllers (BSCs), which manage a group of BSs. The BSs and BSCs are connected to mobile switching centers (MSCs), and MSCs in turn are connected to the core network via access routers [2]. These components are presented in Fig. 1. Generally, the wireless access network maintains a tree-like topology, with the mobile hosts at the leaves, and the MSCs at the root. A significant problem for such architectures with tree-like topologies is that they are vulnerable to failures. A failure of an intermediate node (or link) in the tree will result in the partitioning of the subtree rooted at the failed node (or link) from the rest of the network. Potential failures in the wireless access network include radio channel failures between end users and BSs, BS failures, wired link failures from BSs to BSCs, BSC failures, wired link failures from BSCs to MSCs, and MSC failures.

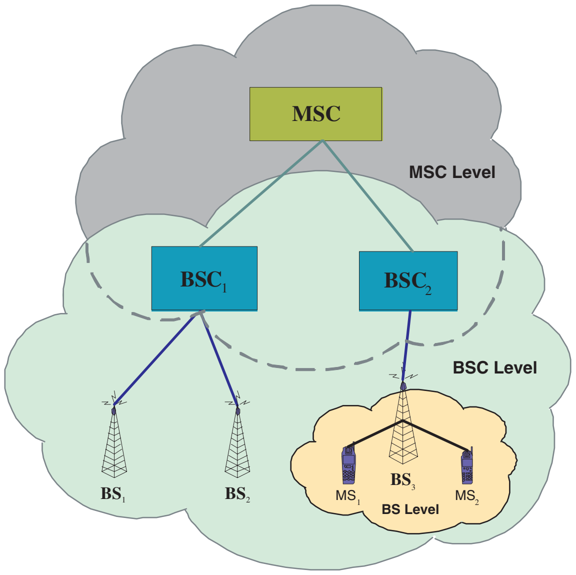

Fig. 1. A Wireless Access Network Architecture

An overview of the survivability issues in wireless access networks are provided in [2], [3], [4], [5], [6], with emphasis on the unique difficulties presented by user mobility and the wireless channel environment. Potential failures from different components or links in the wireless access network are identified, and possible solutions are also proposed. Metrics for quantifying network survivability are identified at each level. In [7], authors proposed the use of dynamic load balancing as a means to improve wireless access network survivability by providing multiple source-destination paths between mobile terminals and fixed base stations in the access network. In [8], node failures and link failures in the access network are handled differently. It is assumed that each node in the wireless network is hot swappable and can be replaced on the fly with spare units. The authors studied how to determine the access

1-4244-0222-0/06/$20.00 (c)2006 IEEE

network topology which minimizes the total cost of the links for a given cost structure, link-failure probabilities, and the capacities and locations of access routers. By considering both reliability and cost, a tree with parallel backup links appears to be the best topology for the access network, and its cost is acceptable for end users. However, since node failures are handled by dedicated redundant resources, the scheme is not efficient.

Dual-homing (or multi-homing) can be used in the wireless access network to improve the network survivability [2], [3] [6], [7], [9], [10]. In dual-homing, a node is connected to the network through two interfaces so that the traffic can be forwarded through an alternate interface if the primary interface fails or if the primary interface is no longer able to forward traffic to the destination due to a link or node failure elsewhere in the network. The advantage of dual-homing is that it can provide survivability against both link and node failures. Also, no dedicated network components are needed as redundant resources. Research on the use of dual-homing architectures to improve survivability can be also found in [11], [12], [13], [14], [15].

To handle the BS failures as well as radio channel failures, an overlapping cell site architecture [7] can be applied to provide dual-homing fault-tolerance. In such an architecture, each BS supports two groups of radio channels, namely short-haul channels and long-haul channels. Short-haul channels cover a small area and long-haul channels cover a large area. Ideally, at any time, each mobile user can access at least two BSs via the two types of channels, one serving as the primary BS and the other as the backup BS. When the communication to the primary BS fails, the mobile user can still access to the core network via the backup BS. As mentioned in [11], with the development of wireless infrastructure, more and more access points are being installed. It is becoming more likely that many overlapping areas for different BSs exist.

Given a dual-homing infrastructure in the BS level, we will study the network design problem, i.e., routing from each BS to a BSC and then to a MSC such that the maximum number of affected channels is minimized when a single failure occurs in the wireless access network.

The rest of the paper is organized as follows. Section II formally describes the problem, and an Integer Linear Programming (ILP) model is given in Section III. The computational complexity of the problem is given in Section IV. A Tabu Search (TS) based heuristic algorithm is presented in Section V, and simulation results are given in Section VI. In Section VII, we conclude the paper.

II. Problem Description

Assuming the access network maintains a tree structure, the network design problem is to determine the organization of the tree, i.e., choosing a BSC for each BS, and an MSC for each BSC, so that certain criteria related to the survivability of the network are optimized. In this paper, we study the survivability problem for mobile users that connect to the core network by dual homing.

In the access network, a failure may occur at a node such as a BS, BSC, or MSC; a failure may also occur at a link such as the link from a BS to a BSC, or the link from a BSC to an MSC. We see that a link failure has the same effect as the corresponding node failure in terms of the connection loss. For example, when a link from BS_1 to BSC_1 is disconnected, all mobile users using BS_1 as the primary BS have to switch to their backup BSs. This is equivalent to the case in which node BS_1 is down. Therefore, in the following, we only discuss node failures in the access network.

Following the convention of existing survivability studies on networking, we assume that at most one failure may occur at any time, and our concern is the worst case situation for a single failure. Specifically, a single failure may be a failure of a single BS, a single BSC, or a single MSC in the access network, and our objective is to minimize the maximum connections lost due to any single failure.

We now discuss the impact of a specific failure, and how protection is provided for dual-homed mobile users in the access network. In a tree-structure of the wireless access network, each BS connects to one BSC which then connects to one MSC. A mobile user, when connecting to a BS, has a unique route to the IP router via a BSC and an MSC. If a mobile user is dual-homed to two different BSs, then it has two routes to the IP router. These two routers may be totally disjoint, i.e., they may use different BSCs and MSCs, or they may also be partially disjoint by sharing a common BSC or MSC.

When the locations of the BSs are given, the traffic load in terms of the average number of connections from mobile users can be estimated. In particular, we assume the traffic load for each BS_i can be described by D_{ij}, i.e., the number of connections using BS_i as the primary BS and using BS_j as the backup BS. When a failure occurs along the route starting from BS_i, these D_{ij} dual-homed connections will switch to another route starting from BS_j. Thus the dual-homing architecture at the BS level provides certain protection to the mobile users.

However, such a scheme may not be able to provide 100% protection against a single node failure for two reasons, namely the partial disjointness of the two routes and the capacity constraint. First, the two routes for D_{ij} may not be totally disjoint. If a failure occurs on a component that is on the shared route, then all D_{ij} connections will be lost even if they are dual-homed to two different BSs. Second, when D_{ij} connections are switched to the backup BS_j, there may not be enough capacity on the route starting from BS_j, even if the route itself is still working. In this case, some or all D_{ij} connections will be lost. Similar events may also happen at the BSC and MSC layers.

We use the examples in Fig. 2 to demonstrate our problem. Suppose that $D_{12} = 300$, i.e., there are 300 connections using BS_1 as the primary BS and BS_2 as the backup BS. If both BS_1 and BS_2 connect to BSC_1, these two routes are not totally disjoint. This scenario can provide protection for D_{12} in the case of BS_1 failure, since these connections can switch to BS_2; however, it cannot provide protection in the

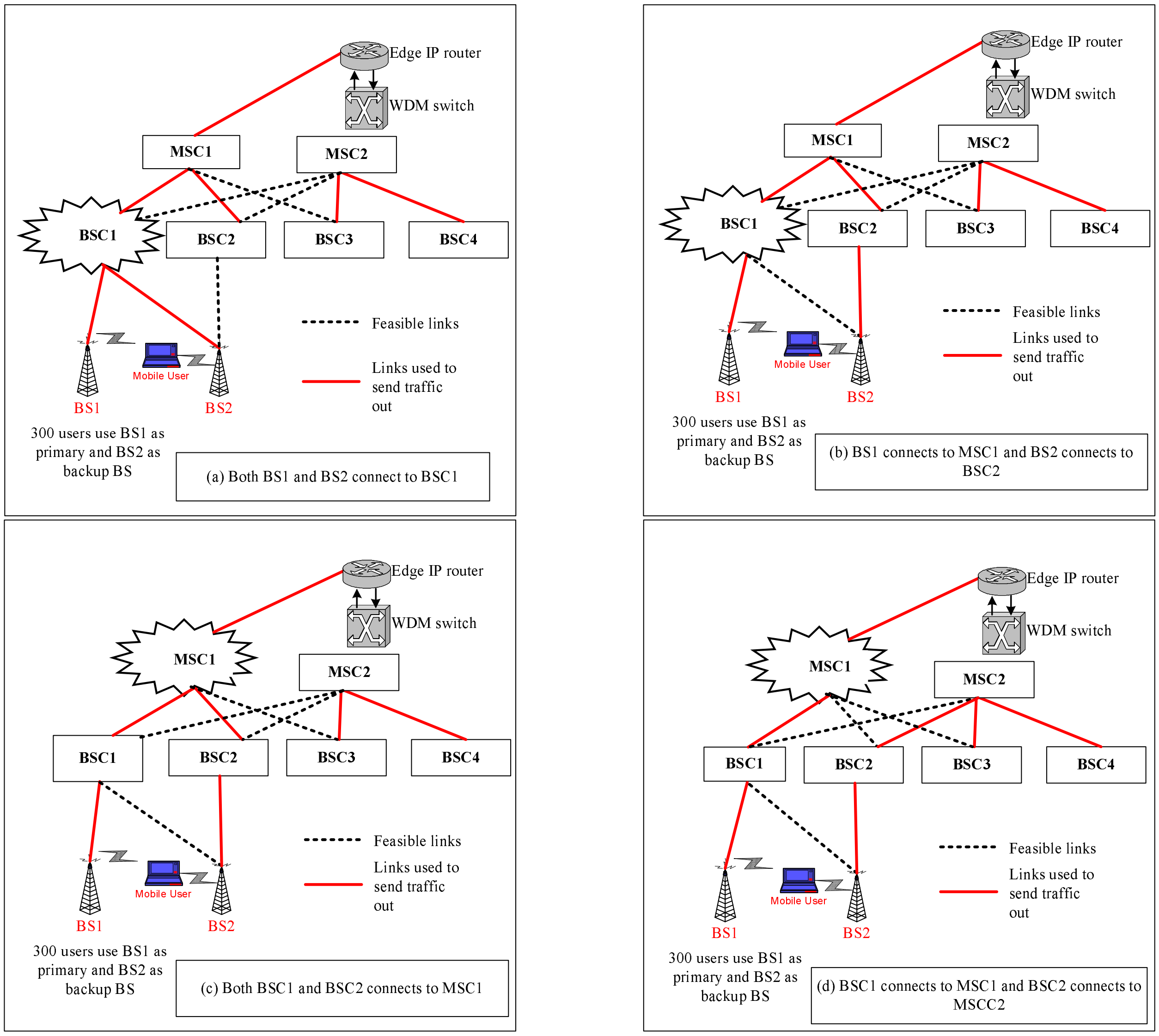
Edge IP router
WDM switch
MSC1
MSC2
BSC1
BSC2
BSC3
BSC4
Feasible links
Links used to send traffic out
Mobile User
BS1
BS2
300 users use BS1 as primary and BS2 as backup BS
(a) Both BS1 and BS2 connect to BSC1
(b) BS1 connects to MSC1 and BS2 connects to BSC2
(c) Both BSC1 and BSC2 connects to MSC1
(d) BSC1 connects to MSC1 and BSC2 connects to MSCC2

Fig. 2. Protection for Dual-homed Mobile Users

case of BSC_1 failure, although they are dual-homed to two BSs. This is illustrated in Fig. 2(a). On the other hand, if the capacity of BS_2 is tight, for example, with 100 spare channels available, then only 100 connections from D_{12} connections can be accommodated at BS_2, and 200 connections from D_{12} will still be lost, even if protection is provided.

Similarly, if BS_1 connects to BSC_1 and BS_2 connects to BSC_2, further protection can be provided for BSC_1 failure as shown in Fig. 2(b), but not for MSC_1 failure as shown in Fig. 2(c). For this case, the capacity constraint at both BS_2 and BSC_2 may cause loss of connections of D_{12}. A better solution is to connect BSC_1 and BSC_2 to two MSCs as shown in Fig. 2(d), where protection is established for either BS_1, BSC_1, or MSC_1 failure, although capacity constraint along the new route BS_2, BSC_2, and MSC_2 may still cause some connection loss.

In order to minimize the worst case loss for every single failure in the access network, it is sufficient to consider a single MSC failure. If the failed node is a BSC, then it has the same effect as the simultaneous failure of all BSs which are connected to the failed BSC. Therefore, a failure of a single BSC will cause at least as many lost connections as a failure of any single BS which is connected to the BSC. For the same reason, the connection loss caused by a single BSC failure cannot be more than the connection loss caused by the failure of the MSC that connects the BSC. Consequently, with respect to the worst-case analysis, we only need to consider a single failure of a MSC which results in the maximum connection loss.

III. Integer Programming Formulation

The problem can be formally defined by an integer linear programming model. In the formulation, we regard the problem as a network flow problem with the connection requests as flows. For each scenario of an MSC failure, we can calculate the total connection loss at all nodes. We then define the loss for the worst scenario case as the objective value.

An alternative intuitive formulation could be easily obtained as an integer quadratic programming formulation with a linear objective function and quadratic constraints; however, such a formulation is not solvable by most well-known optimization softwares, such as CPLEX.

We have the following notations.

Indices:

- i, j: index for BS
- k: index for BSC
- h: index for MSC

Parameters:

- n: number of BSs
- m: number of BSCs
- l: number of MSCs
- K_i: set of BSCs to which BS_i can be connected
- H_k: set of MSCs to which BSC_k can be connected
- D_{ij}: number of connections that use BS_i as the primary BS, and BS_j as the backup BS. In particular, D_{ii} is the number of connections that are not dual-homed to BSs.
- A_i: capacity of BS_i
- B_k: capacity of BSC_k
- C_h: capacity of MSC_h

Variables:

- δ_{ik}: binary variable, equal to 1 if BS_i connects to BSC_k
- θ_{kh}: binary variable, equal to 1 if BSC_k connects to MSC_h
- σ_{ih}: binary variable, equal to 1 if the requests D_{ij} go through MSC_h under normal case
- $x_{ik}^{(h')}$: traffic flow going from BS_i to BSC_k when $MSC_{h'}$ is down
- $y_{kh}^{(h')}$: traffic flow going from BSC_k to MSC_h when $MSC_{h'}$ is down
- $u_i^{(h')}$: connection loss at BS_i when $MSC_{h'}$ is down
- $v_k^{(h')}$: connection loss at BSC_k when $MSC_{h'}$ is down
- $w_h^{(h')}$: connection loss at MSC_h when $MSC_{h'}$ is down

Constraints:

In the access network, each BS connects to a single BSC, and each BSC connects to a single MSC. Thus we have

$$\sum_{k\in K_i} \delta_{ik} = 1, \quad i = 1, \ldots, n. \tag{1}$$

$$\sum_{h\in H_k} \theta_{kh} = 1, \quad k = 1, \ldots, m. \tag{2}$$

$$\sum_{h=1}^{l} \sigma_{ih} = 1, \quad i = 1, \ldots, n. \tag{3}$$

$$\begin{aligned} 1 + \sigma_{ih} &\geq \delta_{ik} + \theta_{kh}, \\ & i = 1, \ldots, n; k = 1, \ldots, m; h = 1, \ldots, l. \end{aligned} \tag{4}$$

Without failure, the capacity constraints on BSs, BSCs and MSCs must not be violated. We obtain

$$\sum_{j=1}^{n} D_{ij} \leq A_i, \quad i = 1, \ldots, n. \tag{5}$$

$$\sum_{i=1}^{n} \delta_{ik} \sum_{j=1}^{n} D_{ij} \leq B_k, \quad k = 1, \ldots, m. \tag{6}$$

$$\sum_{i=1}^{n} \sigma_{ih} \sum_{j=1}^{n} D_{ij} \leq C_h, \quad h = 1, \ldots, l. \tag{7}$$

The satisfiability of formula (5) is determined by the input parameters of an instance.

Consider the case that $MSC_{h'}$ is down. For BS_i, the injected requests consist of the original incoming requests to BS_i and the transferred requests due to the failure of $MSC_{h'}$. Some of the injected requests are passed through to the BSC layer and the remaining requests, which exceed the capacity of BS_i, are dropped. Due to flow conservation at BS_i, we have

$$\begin{aligned} u_i^{(h')} + \sum_{k\in K_i} x_{ik}^{(h')} &\geq \sum_{j=1}^{n} D_{ij} + \sum_{j\neq i} D_{ji}\sigma_{jh'}, \\ & i = 1, \ldots, n; h' = 1, \ldots, l, \end{aligned} \tag{8}$$

where the last term represents the total connection requests that would be transferred to BS_i when $MSC_{h'}$ is down. Because each BS connects to a single BSC and the outgoing flow of a BS can not exceed the capacity of the BS, we need

$$\begin{aligned} x_{ik}^{(h')} \leq A_i \delta_{ik}, \quad & i = 1, \ldots, n; \\ & k = 1, \ldots, m; h' = 1, \ldots, l. \end{aligned} \tag{9}$$

It is worth noting that, even if BS_i is connected (indirectly) to $MSC_{h'}$ (i.e, the case $\sigma_{ih'} = 1$), formula (8) and formula (9) are still valid.

Similarly, due to flow conservation at BSC_k, we have

$$\begin{aligned} v_k^{(h')} + \sum_{h\in H_k} y_{kh}^{(h')} &\geq \sum_{i=1}^{n} x_{ik}^{(h')}, \\ k = 1, \ldots, m; & h' = 1, \ldots, l, \end{aligned} \tag{10}$$

$$y_{kh}^{(h')} \leq B_k \theta_{kh}, \quad k = 1, \ldots, m; h, h' = 1, \ldots, l. \tag{11}$$

For the same reason, at MSC_h for $h \neq h'$, we have

$$w_h^{(h')} \geq \sum_{k=1}^{m} y_{kh}^{(h')} - C_h, \quad h = 1, \ldots, l; h' = 1, \ldots, l, \tag{12}$$

At $MSC_{h'}$(the failed MSC), we have

$$w_{h'}^{(h')} = \sum_{k=1}^{m} y_{kh'}^{(h')} - \sum_{i=1}^{n} (\sigma_{ih'} \sum_{j\neq i} D_{ij}), \tag{13}$$

whose correctness is proved in the appendix.

Thus the total connection loss when $MSC_{h'}$ is down is given by

$$F^{(h')} = \sum_{i=1}^{n} u_i^{(h')} + \sum_{k=1}^{m} v_k^{(h')} + \sum_{h=1}^{l} w_h^{(h')}, \quad h' = 1, \ldots, l. \tag{14}$$

The problem of minimizing the maximum loss for a single MSC failure can be defined by the following linear integer program:

$$\min F$$

subject to

$$F \geq F^{(h')}, \;\; h' = 1, \ldots, h \tag{15}$$

$$\delta_{ik}, \theta_{kh}, \sigma_{ih} \in \{0,1\}, \quad i = 1, \ldots, n; k = 1, \ldots, m; h = 1, \ldots, l. \tag{16}$$

$$x_{ik}^{(h')}, y_{kh}^{(h')}, u_i^{(h')}, v_k^{(h')}, w_h^{(h')} \in Z_0^+, \quad i = 1, \ldots, n; k = 1, \ldots, m; h', h = 1, \ldots, l. \tag{17}$$

and (1) to (14).

IV. Complexity

We will show the intractability of the problem by showing that some of its sub-problems are NP-hard. Specifically, in evaluating the complexity of the problem, we consider a wireless access network with only one MSC and we assume the MSC is reliable. We also assume no reachability constraints from BSs to BSCs. In this context, the problem now is how to set up connections from BSs to BSCs such that the maximum number of connections lost due to a single failure of a BSC is minimized.

First, we study the complexity of the sub-problem in which there is no capacity constraint on BSCs. In other words, we assume the capacities of BSCs are unlimited. The routing problem from BSs to BSCs is equal to the partition problem, which can be formally defined as follows.

Problem $\mathbb{R}$: Given a finite set $A = \{1, 2, \ldots, n\}$ of BSs, with connection demand $D_{ij} \in Z^+, i, j \in A$, a number $m \in Z^+$ of BSCs and a connection loss value $K \in Z^+$, is there a partition $A = A_1 \cup A_2 \cup \ldots \cup A_m$ of A into m disjoint sets (all BSs in A_i are connected to BSC_i) such that the maximum connection loss due to a single failure of BSCs is no more than K, i.e.,

$$\max \left\{ \sum_{i,j \in A_k} D_{ij} : k = 1, 2, \ldots, m \right\} \leq K \; ? \tag{18}$$

We discuss the complexity of Problem $\mathbb{R}$ in two cases.

In the first case, it is assumed that $D_{ij} = 0$ and $D_{ii} > 0$, where $i \neq j$ and $i, j \in A$, i.e., there is no dual-homing protection for all users. Thus, Eq. (18) is simplified to

$$\max \left\{ \sum_{i \in A_k} D_{ii} : k = 1, 2, \ldots, m \right\} \leq K \; ? \tag{19}$$

Then, the problem can be thought of as the *multiprocessor scheduling problem*: each BS as a task, each BSC as a processor, D_{ii} as the "length" of task i, and K as the deadline. The multiprocessor scheduling problem is known to be NP-complete [16]. Therefore, Problem $\mathbb{R}$ is NP-complete in the first case.

In the second case, it is assumed that $D_{ij} > 0$ and $D_{ii} = 0$, where $i \neq j$ and $i, j \in A$, i.e., there is a dual-homing protection for each user. Thus, Eq. (18) is simplified to

$$\max \left\{ \sum_{\substack{i,j \in A_k \\ i \neq j}} D_{ij} : k = 1, 2, \ldots, m \right\} \leq K \; ? \tag{20}$$

Then, the problem can be thought of as one of the classic *k-clustering problem*: each BS as a node, each BSC as a cluster, D_{ij} as the distance between node i and node j, and K as the distance parameter. The corresponding clustering problem is known to be NP-complete [16] (details in [17]). Therefore, Problem $\mathbb{R}$ is also NP-complete in the second case.

Next, we study the complexity of the sub-problem in which BSCs have limited capacity. Intuitively, with capacity constraint, the routing design problem will become harder, which is verified by our proof that even finding a feasible solution to the capacitied routing problem is NP-complete. This capacitied version of the problem can be formally defined as follows.

Problem c-$\mathbb{R}$: Given a finite set $A = \{1, 2, \ldots, n\}$ of BSs, with connection demand $D_{ij} \in Z^+, i, j \in A$, a number $m \in Z^+$ of BSCs with capacity B_k for BSC_k, is there a partition $A = A_1 \cup A_2 \cup \ldots \cup A_m$ of A into m disjoint sets (all BSs in A_i are connected to BSC_i) such that the capacity constraint of each BSC is not violated, i.e.,

$$\sum_{i \in A_k} \sum_{j=1}^{n} D_{ij} \leq B_k, \;\; k = 1, 2, \ldots, m \; ? \tag{21}$$

We prove the NP-completeness of the problem by a reduction from the 3-PARTITION problem which is known to be NP-complete [16].

Proof:

3-PARTITION: Given $3m$ elements with integer sizes $a_1, a_2, \ldots, a_{3m}$, where $\sum_{i=1}^{3m} a_i = mA$ and $A/4 < a_i < A/2$ for $i = 1, \ldots, 3m$, does there exist a partition $S_1, \ldots, S_m$ of the index set $\{1, \ldots, 3m\}$, such that each S_k has 3 elements and that $\sum_{i \in S_k} a_i = A$, for $k = 1, \ldots, m$?

Given an instance of 3-PARTITION, we can construct an instance of Problem c-$\mathbb{R}$ as follows. Let there be $3m$ BSs, m BSCs, and a single MSC. Each BSC has the same capacity A, and the MSC has a capacity of mA. For BS_i, $D_{ii} = a_i$ and $D_{ij} = 0$ where $j \neq i$, i.e., the number of incoming connections to BS_i is a_i, and there is no dual-homing. The problem is to find a feasible network design from the BS level to the BSC level subject to the BSC capacity constraint.

(1) Suppose 3-PARTITION has a feasible solution. Without loss of generality, assume the elements in S_k are $a_{3k-2}, a_{3k-1}, a_{3k}$ for $k = 1, \ldots, m$. Then we can have a feasible network design where $BS_{3k-2}, BS_{3k-1}, BS_{3k}$ are connected to BSC_k for $k = 1, \ldots, m$. It is easy to verify that this is a feasible network design.

(2) Suppose we have a feasible network design. Then we can see that each BSC_k can only be connected by 3 BSs, and the sum of connections from these 3 BSs must be A. Therefore, such a network design corresponds to a feasible solution to 3-PARTITION.

$\square$

In a summary, we have proven that both sub-problems, Problem $\mathbb{R}$ and Problem c-$\mathbb{R}$, are NP-complete:

1) without capacity constraint on BSCs, finding the optimal solution (minimizing maximum connection loss of a single failure), either with or without dual-homing;

2) with capacity constraint on BSCs, finding a feasible solution, without dual-homing.

The above results suggest that the sub-problem with capacity constraint is even harder than the one without capacity constraint (a feasible solution vs. an optimal solution). The original problem involves 3 layers (MSC, BSC, and BS), while the above NP-complete sub-problems only involve on 2 layers (BSC and BS). In the original problem, there are capacity constraints on MSCs, BSCs, and BSs, and there are dual-homed end users from mobile stations to BSs. Hence, it is obvious that, not only is the original problem NP-hard, but finding a feasible solution to the original problem is also NP-hard.

V. Algorithms

In the previous section, we have proved that the discussed problem is NP-hard by showing that its sub-problems, which have weaker constraints, are still NP-hard. Hence, it is reasonable to work on polynomial heuristics instead of polynomial exact algorithms. For completeness, we first briefly discuss a brute force explicit enumeration approach which, obviously, has an exponential complexity. In the rest of the section, we then present the details of a more practical algorithm which is based on the Tabu Search (TS) technique.

Of the problems, there is a special instance in which each BS can reach only one BSC and each BSC can reach only one MSC. It is obvious that in this case there is only one feasible solution, if there exists any feasible solution. If a BS can reach only one BSC, then the route from the BS to BSCs is determined (to the only reachable BSC) so that we can eliminate the BS from the problem. Similarly, if a BSC can reach only one MSC, the BSC can also be eliminated from the problem. Hence, without loss of generality, we assume that any BS can reach at least two BSCs and any BSC can reach at least two MSCs.

A. Brute force approach

One approach is to enumerate all possible route configurations. For an instance of the problem, let r_i be the number of BSCs to which BS_i can reach; let s_i be the number of MSCs to which BSC_k can reach. Then, the number of all possible route configurations can be denoted as $N_c = \Pi_{i=1}^{n} r_i \cdot \Pi_{k=1}^{m} s_k$, where n and m are the number of BSs and BSCs, respectively. Since any BS can reach at least two BSCs and any BSC can reach at least two MSCs, we obtain $N_c \geq 2^{(n+m)}$. If a route configuration does not violate any of the capacity constraints, such a route configuration is a feasible solution. Among all feasible solutions, the one with the minimum connection loss is the optimal solution.

Obviously, the explicit enumeration approach has exponential complexity. Such a brute force approach is not practical for reasonable size problems. Next, after a brief introduction of the Tabu Search (TS) technique, we propose an efficient algorithm based on TS.

B. Introduction of Tabu Search

Tabu Search (TS) is a meta-heuristic which guides a local search heuristic to cross boundaries of local optimality or feasibility [18]. TS starts from an initial solution, which is usually obtained by a (simple) heuristic. Then the algorithm continues an iterative procedure until certain preset terminating criteria are satisfied. In each iteration, TS walks from a current solution to a neighbor solution by applying a local modification (*move*). Usually, it is desired to move to a neighbor solution with a better objective value. However, in order to escape from a local optimum, TS also accepts non-improving moves, which may lead to a visiting cycle.

To (partially) prevent cycling, latest moves are put into a list, referred to as *tabu list*, such that moves in the list are not allowed to be applied to the current solution before the *tabu tenure* of a move expires. A risk of the tabu list is that it may prevent an unvisited solution from being visited. *Aspiration criteria* are employed to handle this drawback. Aspiration criteria are conditions under which, if a move is satisfied, even if the move is in the tabu list, the move is permitted. To speed up the exploration of different areas of the solution space, TS takes a *diversification strategy*, which are types of (global and significant) perturbations to the current solution. In contrast, moves can be thought of as local and less significant perturbations.

Using all these strategies and some other advanced techniques, Tabu Search has proven to be a very successful heuristic algorithm paradigm for many combinatorial problems. Details on Tabu Search can be found in books such as [18], [19], and [20].

In the rest of this section, we will discuss the essential design aspects of the proposed algorithm based on the Tabu Search technique. We first present the framework of the proposed algorithm and then discuss details.

C. Framework of the algorithm

According to the TS paradigm, we design the tabu search algorithm with the following framework:

- Step 1: Obtain an initial solution s. Set the best solution $s^* = s$ and $iter = 0$. Initialize the BS tabu list and BSC tabu list as empty.
- Step 2: Apply *BS moves* one by one to solution s to obtain neighbor solutions. If a BS move is not in the BS tabu list or the aspiration criteria is satisfied, the resulting neighbor solution is put into the candidate neighbor solution list $L_{bs}(s, iter)$.
- Sep 3: Choose from $L_{bs}(s, iter)$ the best solution s'. If s' is better than s, set $s = s'$, add the corresponding BS move into the BS tabu list, and then goto Step 6; otherwise, continue to Step 4.
- Step 4: Apply *BSC moves* one by one to solution s to obtain neighbor solutions. If a BSC move is not in the BSC tabu list or the aspiration criteria is satisfied, the resulting neighbor solution is put into the candidate neighbor solution list $L_{bsc}(s, iter)$.

- Step 5: Choose from $L_{bsc}(s, iter)$ the best solution s'. Set $s = s'$. Add the corresponding BSC move into the BSC tabu list.
- Step 6: If s' is better than s^*, set $s^* = s'$.
- Step 7: Set $iter = iter + 1$. If $iter$ does not reach the pre-set maximum iteration times, goto Step 2; otherwise, continue to Step 8.
- Step 8: Do an intensification search on s^*. Return the best solution as the final solution.

In the rest of this section, we discuss the detailed design issues of the proposed algorithm framework.

D. Search space and neighborhood structure

The *search space* of Tabu Search is the space of all possible solutions which can be visited during the search. The *neighborhood structure* of a solution is a set of solutions which can be reached from the designated solution through a single *local move*.

Solution definition

We have proved in Section IV that finding a feasible solution to the proposed problem is NP-hard. If we confine the search space to include only feasible solutions, it will be very hard to obtain an initial solution or to further find the neighbors of a solution, which in turn will impede the Tabu Search procedure. We relax the constraint that all BSs and BSCs must be connected and allow some BSs and/or some BSCs to be unconnected to the network. Using the same notation as in the ILP (see Section III), we define a *generalized solution*, s, as a collection of assignments to variables

$$\delta_{ik},\ \theta_{kh},\ \sigma_{ih}$$
$$i = 1, \ldots, n; k = 1, \ldots, m; h = 1, \ldots, l,$$

such that

$$\sum_{k \in K_i} \delta_{ik} \leq 1, \quad i = 1, \ldots, n. \tag{22}$$

$$\sum_{h \in H_k} \theta_{kh} \leq 1, \quad k = 1, \ldots, m. \tag{23}$$

$$\sum_{h=1}^{l} \sigma_{ih} \leq 1, \quad i = 1, \ldots, n. \tag{24}$$

with capacity constraints of Eq. (4) - (7). By this definition, a generalized solution could be a feasible solution (in which all BSs and BSCs are connected) or an infeasible solution (in which some BSs and/or BSCs are unconnected). Hereafter, we will use the term *generalized solution* and *solution* interchangeably, if there is no confusion by the context.

Objective function definition

With the generalized solution definition, the objective function should also be revised accordingly, since with some BSs and/or BSCs unconnected, some connections are lost permanently even if there is no failure in the network. We will take this effect as a penalty function to the original objective function (Eq. (14)-(15)) by defining a *generalized objective function*. First, we introduce some notation:

- $u(s)$: connection loss due to unconnected BSs and BSCs in solution s. $u(s)$ is a penalty factor resulting from the relaxation in the definition of the generalized solution.
- $f(s, h)$: connection loss in solution s due to the failure of MSC_h, where $h = 1, 2, \ldots, m$.
- $f'(s, h)$: total connection loss in solution s when MSC_h fails, where $h = 1, 2, \ldots, m$. Obviously, $f'(s, h) = u(s) + f(s, h)$.

To make our Tabu Search based algorithm more flexible, we define a weighted total connection loss function $g(s, h)$ as:

$$g(s, h) = \alpha \cdot u(s) + f(s, h),\ \alpha \in R^+. \tag{25}$$

Then, the *generalized objective function* $g(s)$ is defined as follows:

$$g(s) = \max_{h=1}^{m}\ g(s, h). \tag{26}$$

In Eq. (25), α is a positive real number used to adjust the weight of the penalty function in the generalized objective function. In the Tabu Search procedure, a larger α means that unconnected BSs and unconnected BSCs have a high chance of being re-connected, while a smaller α usually means that the moves evolved in connected nodes are preferred. If solution s is a feasible solution, i.e., all BSs and BSCs are connected, $u(s)$ will be 0, in which case the generalized objective function $g(s)$ takes the same value as the original objective function.

The objective function $g(s)$ is a maximum function. For such an objective function, local perturbations to a current solution may very likely generate many neighbor solutions that have the same objective value as the current solution, which is not good for guiding the local search procedure. In order to distinguish neighbor solutions, we further define an *extended objective function* $\overrightarrow{g(s)}$ as follows:

$$\overrightarrow{g(s)} =< g(s, \pi(1)), g(s, \pi(2)), \cdots, g(s, \pi(l)) >, \tag{27}$$

where π is a permutation of $(1, 2, \cdots, l)$, i.e., indices of all MSCs, such that $g(s, \pi(h+1)) \leq g(s, \pi(h)$ for $h = 1, \ldots, l-1$. We define a lexicographic order $\prec$ on the extended objective function as:

$$\overrightarrow{g(s_2)} \prec \overrightarrow{g(s_1)} \tag{28}$$

if and only if

$$\begin{aligned} &\exists h,\ 1 \leq h \leq l, \text{ such that} \\ &\quad g(s_2, \pi(i)) = g(s_1, \pi(i)),\ 1 \leq i \leq h-1, \text{ and} \\ &\quad g(s_2, \pi(h)) < g(s_1, \pi(h)). \end{aligned} \tag{29}$$

It can be seen clearly that

$$\overrightarrow{g(s_2)} \prec \overrightarrow{g(s_1)} \Rightarrow g(s_2) \leq g(s_1), \tag{30}$$

which means that the optimal solution s^* with objective function $\overrightarrow{g(s)}$ is also an optimal solution with the objective function $g(s)$. If s^* is also a feasible solution, then s^* is an optimal solution to the problem with the original objective function.

The principle behind the extended objective function is that, by trying to make connection loss distributed as evenly as possible to all MSCs, the maximum connection loss is very likely minimized. We may wish to extend the objective function further to include the connection loss due to failures of BSCs. However, even distribution of connection loss on BSCs will usually not minimize the maximum connection loss due to a single failure of MSCs.

BS Move

A move is a local perturbation to a solution which leads to a neighbor solution, either a feasible or an infeasible solution. We define a BS move as a triple vector (BS_i, BSC_s, BSC_d), which means that BS_i is first disconnected from BSC_s and then is connected to BSC_d if possible. By the definition, the number of neighbors of a solution is bound by $O(nm)$, where n and m is the number of BSs and BSCs, respectively. The upper bound is reached in the case that all BSs are connected and every BS can reach all BSCs. For a BS move (BS_i, BSC_s, BSC_d), we denote MSC_s and MSC_d as the MSC to which BSC_s and BSC_d is connected. A BS move operates as follows.

- *Step* 1: Disconnect BS_i from BSC_s, update the residual capacities of BSC_s and MSC_s.
- *Step* 2: If BSC_d is reachable by BS_i and there are enough residual capacities in both BSC_d and MSC_d to carry the connections from BS_i, then BS_i is connected to BSC_d, the residual capacities of BSC_d and MSC_d are updated, and the resulting solution is a feasible solution; otherwise, BS_i becomes unconnected and the resulting neighbor solution is an infeasible solution.
- *Step* 3: For each BS_j which is unconnected and can reach to BSC_s, if both BSC_s and MSC_s have enough residual capacities to carry connections from BS_j, connect BS_j to BSC_s and update the residual capacities of BSC_s and MSC_s.

It is not surprising that some neighbor solutions are infeasible solutions according to the definition of the BS move. However, even if a neighbor solution is infeasible, it may be a better solution than the current solution under the extended objective function Eq. (27).

E. Initial solution

In general, the performance of a tabu search procedure does not depend on the initial solution. A good initial solution, however, may speed up the search process. As finding a feasible solution is NP-hard, the initial solution may be an infeasible solution. We propose the following simple algorithm to obtain an initial solution.

- *Step* 1: Sort BSs in a decreasing order in terms of the primary traffic of BSs;
- *Step* 2: Sequentially handle each BS as follows. Among all BSCs which are reachable by the BS, connect the BS to the BSC which has the maximum residual capacity and its residual capacity is greater than or equal to the primary traffic of the BS. If no such BSC exists, the BS is left unconnected.
- *Step* 3: Sort BSCs in a decreasing order in terms of carried traffic from BSs.
- *Step* 4: Sequentially handle each BSC as follows. Among all MSCs which are reachable by the BSC, connect the BSC to the MSC which has the maximum residual capacity and its residual capacity is greater than or equal to the carried traffic of the BSC. If no such MSC exists, the BSC is left unconnected temporarily.
- *Step* 5: For each of the unconnected BSCs in Step 4, reduce connections to the BSC as follows. Disconnect, one by one, BSs which were connected to the BSC in Step 2 until the BSC can connect to some MSC which has enough residual capacity or until all BSs are disconnected from the BSC. In the later case, the BSC is left unconnected.

The idea behind this algorithm is to maximize the number of connected nodes and to balance the load of nodes as much as possible. Hence, in the algorithm, we sequentially choose the most heavily loaded node to be assigned to the most lightly loaded upper layer node. The complexity of the initial algorithm is $O(n \lg n + nm + m \lg m + ml)$, where n, m, l is the number of BSs, BSCs, and MSCs, respectively. The complexity is approximately equal to $O(n \lg n)$ for a typical network with $m < \lg n$ and $l < m$.

F. Aspiration criteria

In this paper, we adopt the most commonly used aspiration criteria which allows a move, even if it is tabu, if it results in a solution with an objective value better than the current best solution.

G. Diversification strategy

At some points in the tabu search procedure, only applying BS moves may not be able to further improve the extended objective function. We implement a diversification strategy by making higher level moves, *BSC moves*. A procedure similar to the BS move is applied to BSC moves. To save space, we omit the details here.

It is worth noting that, since each BSC usually has a larger number of end users than a BS, it may be likely that a BSC move will increase, instead of decrease, the extended objective function, especially when load is virtually balanced among nodes. This is the difference between a BSC move and a BS move. However, the diversification strategy in Tabu Search is not intended to find a better solution than the current solution, but intended to systematically exploit the solution space. Our numeric results show that the above diversification strategy is very effective.

H. Tabu list

A tabu list is used to (partially) prevent visit-cycling. As mentioned above, we have two types of moves in the algorithm, BS moves and BSC moves. We use two separated tabu lists, one for BS moves and the other for BSC moves.

After a BS move (BS_i, BSC_s, BSC_d) is executed, we prevent a move (BS_i, BSC_d, BSC_s) from being executed in a few iterations τ_{bs} by adding the move into the BS move tabu list with a *tabu tenure* τ_{bs}. Tabu tenure τ_{bs} is usually a small number. After τ_{bs} iterations, the tabu status of the BS move will expire and the move will then be allowed in the future iterations.

Similar handling is applied to BSC moves and the BSC tabu list. The only difference is that the tabu tenure for a BSC move is longer than that of a BS move, since the diversification operation is executed less frequently than the BS moves.

I. Intensification strategy

The proposed algorithm goes through an additional intensification search phase after the iterative phase finishes. The purpose is to further fine tune and to improve the best solution obtained in the iterative phase. When the load of nodes are almost balanced (after the iterative phase), a single move of a BS from one BSC to another BSC will usually result in a worse solution. The reason is that under a load balanced state, one additional BS to a BSC will greatly increase the connection loss. However, if we exchange the connections of a pair BSs, the net impact to the two BSCs may be minimized. We implement a *BS swap* (BS_i, BS_j) as our intensification strategy.

We denote the BSCs to which BS_i and BS_j is connected as BSC_i and BSC_j, the MSCs to which BSC_i and BSC_j is connected as MSC_i and MSC_j, respectively. A BS swap (BS_i, BS_j) operates as follows.

- *Step* 1: Disconnect BS_i from BSC_i, update the residual capacities of BSC_i and MSC_i.
- *Step* 2: Disconnect BS_j from BSC_j, update the residual capacities of BSC_j and MSC_j.
- *Step* 3: If BSC_j is reachable by BS_i and there are enough residual capacities in both BSC_j and MSC_j to carry the connections from BS_i, then BS_i is connected to BSC_j and the residual capacities of BSC_j and MSC_j are updated; otherwise, BS_i becomes unconnected.
- *Step* 4: If BSC_i is reachable by BS_j and there are enough residual capacities in both BSC_i and MSC_i to carry the connections from BS_j, then BS_j is connected to BSC_i and the residual capacities of BSC_i and MSC_i are updated; otherwise, BS_j becomes unconnected.

A BS swap has a constant complexity $O(1)$.

For the best solution obtained in the iterative phase, do all possible pair-wise BS swaps to obtain new solutions, among which the solution with the minimum connection loss is the final best solution.

VI. Numerical Results

In this section, we first evaluate the performance of the proposed algorithm on small networks by comparing the simulation results with the results obtained from the ILP formulation. For non-trivial networks, the ILP formulation is not solvable in reasonable time and the non-integer relaxation gives out a very loose lower bound. Hence, for non-trivial networks, we will only compare the performance of the Tabu Search based algorithm with the initialization algorithm. Numeric results show that, on the test cases we are running, the proposed algorithm usually produces results within 5% of the optimal solutions on small networks. For moderate networks, solutions by the proposed algorithm are much better than the initialization algorithm.

A. Test settings

The configuration for a test point can be described by a configuration vector $< u, n, m, l, r_1, r_2, c >$, in which

- u: number of total users;
- n: number of BSs;
- m: number of BSCs;
- l: number of MSCs;
- r_1: number of BSCs to which each BS can reach;
- r_2: number of MSCs to which each BSC can reach;
- c: scale factor to the capacity of BS, BSC and MSC, $c \geq 1.0$.

Ideally, if each BS has capacity of u/n, then all u users can be connected to BSs; if the capacity of BS, BSC, and MSC is set respectively by

$$C_{bs} = c \cdot u/n, \;\; C_{bsc} = c \cdot u/m, \;\; C_{msc} = c \cdot u/l,$$

we can statistically adjust the spare capacity in the network by adjusting the value of c.

For each test point, we randomly generate 10 test instances based on its configuration vector and according to the following rules:

1) For each BS, randomly choose r_1 BSCs among all m BSCs as the set of BSCs to which the BS can reach;
2) For each BSC, randomly choose r_2 MSCs among all l MSCs as the set of MSCs to which the BSC can reach;
3) Generate traffic matrix $D_{n \times n}$.
   - Initialize all elements in the matrix $D_{n \times n}$ to be 0;
   - Put BSs, one by one, randomly into a 1.0×1.0 area and set the residual capacity of each BS to be the capacity of BSs;
   - Put end users, one by one, randomly into the 1.0×1.0 area. For each end user, among all BSs which still have enough residual capacity, find the nearest BS_i as its primary BS and the second nearest BS_j as its backup BS. Then reduce the residual capacity of BS_i and BS_j by 1 unit and increase the traffic element D_{ij} by 1 unit.

B. Test results

We first study the performance of the proposed algorithm on a small network, which is specified by a configuration vector $< 1000, 10, 4, 2, 4, 2, c >$, i.e., there are 1000 users, 10 BSs, 4 BSCs, and 2 MSCs; each BS can reach all 4 BSCs and each BSC can reach all 2 MSCs; the capacity scale factor c is specified as in Table I. For each capacity scale factor c, 10 test instances are generated randomly according to the rules in Section VI-A. The results are shown in Table I, each line for an instance. There are four columns for each capacity configuration, which are the objective value from the initialization procedure, the objective value from the Tabu Search based algorithm, the objective value from the ILP, and the performance difference between the Tabu Search based

TABLE I

Objective values of randomly generated instances with configuration $< 1000, 10, 4, 2, 4, 2, c >$.

c=2.0				c=1.8				c=1.6				c=1.4			
Init	Tabu	ILP	Perf Diff	Init	Tabu	ILP	Perf Diff	Init	Tabu	ILP	Perf Diff	Init	Tabu	ILP	Perf Diff
411	226	226	0.00%	331	279	269	3.72%	452	329	314	4.78%	392	389	372	4.57%
309	211	211	0.00%	329	245	245	0.00%	358	294	292	0.68%	459	370	370	0.00%
289	228	184	23.91%	327	223	223	0.00%	365	280	280	0.00%	422	370	370	0.00%
359	229	229	0.00%	352	249	248	0.40%	360	292	292	0.00%	464	370	370	0.00%
268	136	136	0.00%	288	190	190	0.00%	322	280	280	0.00%	400	370	370	0.00%
274	172	172	0.00%	294	221	221	0.00%	327	300	280	7.14%	393	370	370	0.00%
357	227	227	0.00%	407	239	239	0.00%	440	280	280	0.00%	471	370	370	0.00%
208	119	119	0.00%	265	193	190	1.58%	335	280	280	0.00%	404	370	370	0.00%
339	192	183	4.92%	388	241	241	0.00%	424	299	299	0.00%	440	370	370	0.00%
297	210	210	0.00%	357	251	251	0.00%	433	338	301	12.29%	441	370	370	0.00%

TABLE II

Objective values of randomly generated instances with configuration $< 5000, 40, 12, 6, 6, 3, c >$.

c=2.0		c=1.8		c=1.6		c=1.4	
Init	Tabu	Init	Tabu	Init	Tabu	Init	Tabu
398	130	460	231	733	345	661	403
514	185	582	197	455	246	495	323
576	234	667	283	678	372	563	415
528	231	652	230	587	363	525	443
513	114	444	216	691	310	918	415
390	198	447	200	660	312	691	394
459	213	604	208	511	303	557	335
461	72	732	190	432	250	711	351
453	132	446	210	642	325	658	432
475	98	749	161	505	275	743	320

algorithm and the ILP. According to the results in Table I, we have the following observations:

- The proposed Tabu Search based algorithm produces, in most cases, optimal solutions or have performance difference less than 5% compared to the optimal solutions, although in our tests there are 3 out of 40 cases whose performance difference is greater than 5%.
- The Tabu Search based algorithm works well both in networks with abundant capacity (such as the case with $c = 2$) and in networks with sparse capacity (such as the case with $c = 1.4$).
- The solutions from the initialization algorithm are usually very far away from the optimal solutions. The Tabu Search based algorithm can significantly improve the initial solutions, even though only 32 iterations are executed in our tests.

Next, we evaluate the performance of the Tabu Search based algorithm in a middle scale network, which is specified by a configuration vector $< 5000, 40, 12, 6, 6, 3, c >$. That is, there are 5000 users, 40 BSs, 12 BSCs, and 6 MSCs; each BS can reach 6 BSCs and each BSC can reach 3 MSCs; the capacity scale factor c is specified as in Table II. For each capacity scale factor c, 10 test instances are generated randomly according to the rules in Section VI-A. The results are shown in Table II, each line for an instance. There are two columns for each capacity configuration, which are the objective value from the initialization procedure and the objective value from the Tabu Search based algorithm. It can be seen clearly that the Tabu Search based algorithm can significantly improve the initial solutions.

VII. Conclusion

In this paper, we identify and define an important wireless access network design problem, whose objective is to minimize the maximum connection loss due to a single failure of a node, under the assumption that end users are dual-homed to two base stations. We first formulate the problem using Integer Linear Programming. The hardness of the problem is revealed by showing that some of its subproblems, which are obtained by relaxing some constraints, are NP-hard. An iterative Tabu Search based algorithm is proposed for the problem. Simulation results show that the proposed algorithm has good performance in most cases, while it may produce far from optimal solution in some cases. This performance variation is not uncommon among heuristics which have no performance guarantee. An interesting area of future work is to develop performance guaranteed approximation algorithms, especially for the k-clustering subproblem discussed in Section IV. The k-clustering problem is well-studied with variable objective functions. However, except for the proof of NP-hardness, there is no further study on the k-clustering problem with the objective function of Eq. (20), which is of significant importance in designing fault-tolerant wireless access networks.

Acknowledgments

This work was supported in part by the National Science Foundation (NSF) under Grant number CNS-0435105 and CNS-0435095.

Appendix
Proof of Eq. (13)

We use the same notations as in Section III. When $MSC_{h'}$ fails, according to Eq. (8) - Eq. (12), we obtain the incoming traffic to the failed $MSC_{h'}$ as

$$\sum_{k=1}^{m} y_{kh'}^{(h')} = \sum_{i=1}^{n} \left[\sigma_{ih'} \cdot (\sum_{j\neq i}^{n} D_{ij} + D_{ii})\right] \qquad (31)$$

$$+ \sum_{j\neq i}^{n} \sigma_{ih'} \cdot \sigma_{jh'} \cdot D_{ij} \qquad (32)$$

$$- \sum_{i=1}^{n} u_i^{(h')} \cdot \sigma_{ih'} - \sum_{k=1}^{m} v_k^{(h')} \cdot \theta_{kh'}, \qquad (33)$$

in which the item (31) is the original traffic to $MSC_{h'}$ before the $MSC_{h'}$ fails, the item (32) is the transferred traffic between BSs which are both connected (indirectly) to $MSC_{h'}$, and the item (33) is the overflow traffic at BSs and BSCs which are connected to $MSC_{h'}$.

Then, Eq. (13) can be simplified as follows:

$$\begin{aligned} w_{h'}^{(h')} &= \sum_{k=1}^{m} y_{kh'}^{(h')} - \sum_{i=1}^{n} (\sigma_{ih'} \sum_{j\neq i}^{n} D_{ij}) \\ &= \sum_{i=1}^{n} \sigma_{ih'} \cdot D_{ii} + \sum_{j\neq i}^{n} \sigma_{ih'} \cdot \sigma_{jh'} \cdot D_{ij} \\ &\quad - \sum_{i=1}^{n} u_i^{(h')} \cdot \sigma_{ih'} - \sum_{k=1}^{m} v_k^{(h')} \cdot \theta_{kh'}. \qquad (34) \end{aligned}$$

The explanation of the above equation is intuitive. The lost connections at the failed $MSC_{h'}$ are connections which are satisfied the conditions (a)+(c) or (b)+(c): (a) connections which are routed through $MSC_{h'}$ but not dual-homed; (b) connections whose primary BS and backup BS are both connected to $MSC_{h'}$; and (c) connections which are not lost at BSs or BSCs that are connected to $MSC_{h'}$.

References

[1] E. Gelenbe, P. Kammerman, and T. Lam, "Performance considerations in totally mobile wireless," *Performance Evaluation*, vol. 36, no. 7, pp. 387-399, Aug. 1999.

[2] D. Tipper, T. Dahlberg, H. Shin, and C. Charnsripinyo, "Providing fault tolerance in wireless access networks," *IEEE Communication Maganize*, vol. 40, no. 1, pp. 58-64, Jan. 2002.

[3] D. Tipper, S. Ramaswamy, and T. Dahlberg, "PCS network survivability," *Proceedings of Wireless Communications and Networking Conference*, vol. 2, pp. 1028-1032, Sep. 1999.

[4] U. Varshney, A. Snow, and A. Malloy, "Designing survivable wireless and mobile networks," *Proceedings of Wireless Communications and Networking Conference*, vol. 3, pp. 21-24, Sep. 1999.

[5] A. Snow, U. Varshney, and A. Malloy, "Reliability and survivability of wireless and mobile networks," *Computer*, vol. 33, no. 7, pp. 49-44, July 2000.

[6] C. Charnsripinyo, and D. Tipper, "Designing fault tolerant wireless access networks," *Proceedings of MILCOM*, vol. 1, pp. 525-529, Oct. 2002.

[7] T. A. Dahlberg and J. Jung, "Survivable load sharing protocols: a simulation study," *Wireless Networks*, Kluwer Academic Publishers, Vol. 7, no. 3, pp. 283-296, May 2001.

[8] F. Houeto, S. Pierre, R. Beaubrun, and Y. Lemieux, "Reliability and cost evaluation of third-generation wireless access network topologies: a case study," *IEEE Transactions on Reliability*, vol. 51, no. 2, pp. 229-239, June 2002.

[9] D. Din and S. Tseng, "A genetic algorithm for solving dual-homing cell assignment problem of the two-level wireless ATM networks," *Computer Communications*, vol. 25, no. 17, pp. 1536-1547, Nov. 2002.

[10] A. Dutta and P. Kubat, "Design of partially survivable networks for cellular telecommunication systems," *European Journal of Operational Research*, vol. 118, pp. 52-64, 1999.

[11] C. Ahlund and A. Zaslavsky, "Multihoming with mobile IP," *Lecture Notes in Computer Science*, vol. 2720, pp. 235-243, 2003.

[12] F. Farahmand, A. F. Fumagalli, and M. Tacca, "Near-optimal design of WDM dual-ring with dual-crossconnect architecture," *Proceedings, SPIE Optical Networking and Communication Conference (OptiComm) 2002*, Boston, MA, vol. 4874, pp. 286-297, July 2002.

[13] C. Lee and S. Koh, "A design of the minimum cost ring-chain network with dual-homing survivability: a Tabu Search approach," *Computers Operations Research*, vol. 24, no. 9, pp. 883-897, 1997.

[14] A. Proestaki and M. Sinclair, "Design and dimensioning of dual-homing hierarchical multi-ring networks," *IEE Proceedings of Communications*, vol. 147, no. 2, pp. 96-104, April 2000.

[15] M. Willebeek-LeMair and P. Shahabuddin, "Approximating dependability measures of computer networks: an FDDI case study," *IEEE-ACM Transactions on Networking*, vol. 5, no. 2, pp. 311-327, April 1997.

[16] M. R. Garey and D. S. Johnson, "*Computers and intractability: a guide to the theory of NP-completeness*," W. H. Freeman and Company, New York, 1979.

[17] P. Brucker, "On the complexity of clustering problems," in R. Henn, B. Korte, and W. Oletti (eds.), *Optimization and Operations Research, Lecture Notes in Economics and Mathematical Systems*, Springer-Verlag, Berlin, 1978.

[18] F. Glover and M. Laguna, "*Tabu Search*," Kluwer Academic Publishers, 1997.

[19] C. R. Reeves, "*Modern heuristic techniques for combinatorial problems*," John Wiley & Sons, INC, 1993.

[20] F. Glover and G. A. Kochenberger (eds), "*Handbook of metaheuristics*," Kluwer Academic Publishers, 2003.