Providing Fault Tolerance in Wireless Backhaul Network Design with Path Restoration

Chalermpol Charnsripinyo*, Pakorn Leesutthipornchai[+] and Naruemon Wattanapongsakorn[+]
*National Electronics and Computer Technology Center, Pathumthani, Thailand
[+]Department of Computer Engineering, King Mongkut's University of Technology Thonburi
126 Pracha-Utid, Tung-Kru, Bangkok 10140 Thailand. E-mail : naruemon@cpe.kmutt.ac.th

Abstract

This paper presents a survivable network design using network path restoration approach to provide fault tolerance in cellular backhaul network design, so that any breakdown of links within a network path can no longer interrupt the network services. As a consequence, the network design can greatly enhance the network reliability and quality of services. We adopt a two-phase network design approach to reduce the complexity of network design and to provide a design mechanism for enhancing reliability in existing networks. The first phase provides a minimum-cost initial network design. The problem is to find a minimum-cost network topology which includes selecting the location and type of base station controllers and mobile switching controllers as well as their link types. The second phase provides backup paths and spare capacity to the network topology from phase one to improve network reliability. Due to the complexity of network design problem, a genetic algorithm is applied as a meta-heuristic technique for obtaining good solutions. Various problem sizes of example networks are considered and discussed.

Keywords: Survivable Network Design, Network Path Restoration, Optimization Algorithm, Genetic Algorithm

1. Introduction

Wireless access networks have been increasingly important to support mobile users that need communication services any where and any time. As user dependence on wireless mobile services increases, reliable network services become important. A single component failure in wireless cellular networks (i.e., loss of a single link in a network path between a base station and mobile switching center) can disconnect communication services to a great number of mobile users in the service area. The importance of network reliability and performance of wireless mobile networks in the event of network component/facility failures have been previously studied [1]-[3]. Several techniques and solution algorithms for designing a survivable topology of wireless networks were also presented in [4]-[7].

A typical architecture of a cellular wireless network is illustrated in Figure 1. A base transceiver station (BTS) serves as a fixed access point for all mobile terminals within its coverage area. The network includes base station controllers (BSC), which manage a group of base stations and perform radio level channel management. The BTS and BSC are connected to backbone networks via mobile switching centers (MSC). The MSC is then connected to the public switched telephone network (PSTN) or other transmission networks. The network is generally organized into a tree-like structure. Each BTS is connected to a BSC and the BSC is then connected to a MSC.

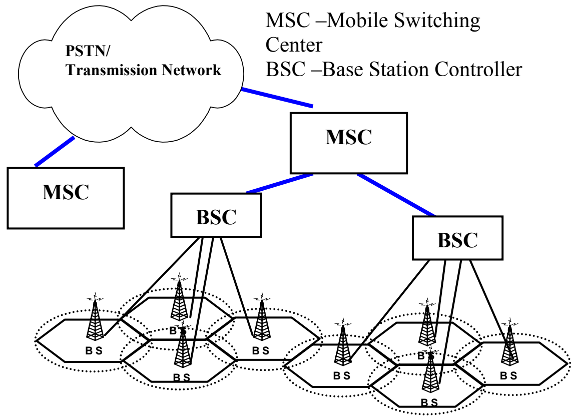

MSC –Mobile Switching Center
BSC –Base Station Controller
PSTN/ Transmission Network
MSC
MSC
BSC
BSC
BS

Fig. 1. Typical wireless network architecture.

In this paper, we present a design of survivable wireless backhaul network using network path restoration mechanism. We adopt a two-phase network design approach in [7]. The first phase design provides a minimum-cost initial network design with tree topology. We consider the network design model proposed in [8] in our first phase which is to find the network topology, the location of the BSCs and MSCs, the type of BSCs and MSCs, and the type of links. The design objective is to minimize the cost of the design, network equipments and their installation. We previously conducted a network topology design with tree-like networks in [9]. In phase two, a mesh-topology network is considered in order to improve network reliability. We use network path restoration mechanism to protect each network path against network link failure from phase one design. A number of research papers presented network path restoration mechanisms to improve the network reliability [12-15]. However, most of previous works on network path restoration techniques were applied to general backbone network design with mesh topology. In cellular network design, the characteristic of hierarchical tree structure in the backhaul network must be taken into account.

To compare the reliability of the ordinary network design with tree structure and the network design with an extension of path restoration approach, let us simply consider a small network size as shown in Figure 2 with 1 MSC, we then have a full m-ary tree network with height equal to 2 where m is 5, and the reliability of each network link equals to 0.99 assuming statistical-independent link/path failure.

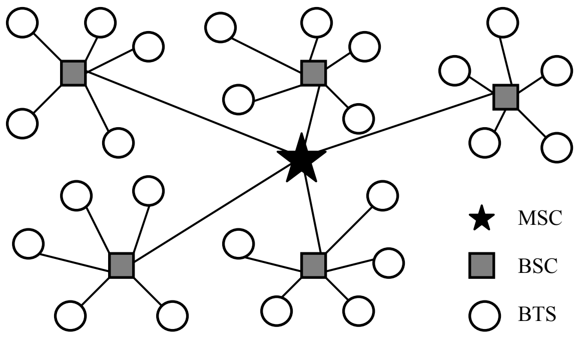


Fig. 2. Cellular network with tree topology.

As a result, the reliability of this m-ary tree network is $(0.99)^n$, where n is the number of link in the network, i.e. m^2+m. Thus reliability of ordinary m-ary tree network equal to $(0.99)^{m^2+m}$.

Consider path restoration design, the network is designed so that a backup path with a minimum of 2 links is provided for each working path. The reliability of each path with a backup path can be up to $1-(1-(0.99^2))^2 = 0.9996$. Thus the reliability of the network with path restoration is up to $(0.9996)^n$, where n is the number of path in the tree network, i.e. m^2. The reliability of network path restoration can be $(0.9996)^{m^2}$.

We see that the reliability of the ordinary tree network, $(0.99)^{m^2+m}$ is less than the reliability of network path restoration, $(0.9996)^{m^2}$.

We apply a genetic algorithm (GA) as a heuristic technique to solve the reliable wireless backhaul network design problem. Genetic Algorithm (GA) can be applied to solve many difficult engineering problems, and it is particularly effective for the combinatorial optimization problems. GA can also be used to solve optimization problems when the problem space is huge and trying all possibilities is not an option. An example of problem solving with the GA is the component redundancy allocation in series-parallel systems [10]. Previously, we also used the GA to solve the minimum-cost network design problem [9] and optimize reliability communication network design by finding network component allocation and network topology with cost constraint [11].

The remainder of this paper is organized as follows. In the next section, we present the cellular wireless network design problem and the network design model. A two-phase network design approach consisting of the minimum-cost network design and network path restoration design will be described. Section III gives the GA concept and presents our parameter settings used for obtaining good solutions in the cellular wireless network design problem. Section IV presents our numerical results and analysis. Lastly, section V concludes our research work and contribution.

2. Network Design Problem and Model

We consider the problem of wireless backhaul network design considering reliability of the network in the event of link failures. Due to the complexity of the network design problem, we adopt a two-phase network design approach proposed in [7]. In the first phase, a minimum-cost initial network design with tree topology is considered.

Phase 1: Minimum Cost Network Design with Tree Topology

The first phase of cellular wireless backhaul network design is to determine the network topology including the location of the BSCs and MSCs, the type of BSCs and MSCs, and the type of links. The network design problem can be formulated as an optimization-based model. In this paper, we adopt the constraint programming model from [8]. The network design objective is to minimize the total network cost as shown in Equation 1 consisting of the cost of the links and interface cards and the cost of the BSCs and MSCs which include the installation cost.

Objective: $$\min_{v,w,x,y,z} C_L(v,w,z) + C_N(x,y) \qquad (1)$$

where C_L is the cost of the link and interface cards, and C_N is the cost of the allocated BSCs and MSCs. Their descriptions are represented as following

$$C_L(v,w,z) = \sum_{i\in I} a_{iv_i} + \sum_{j\in J}\sum_{i\in L} b^l_{jw_j} z^l_j \qquad (2)$$

$$C_N(x,y) = \sum_{j\in J} c_j^{x_j} + \sum_{k\in K} d_k^{y_k} \qquad (3)$$

where I is the set of BTSs, J is the set of BSCs, and K is the set of MSCs. The cost parameters consist of a_{ij}, b_{jk}^{l}, c_{j}^{s} and d_{k}^{t} where a_{ij} is the link and interface card costs including the installation cost for connecting BTS $i \in I$ to a BSC at site $j \in J$, b_{jk}^{l} is the link and interface card costs including installation cost for connecting a BSC at site $j \in J$ to an MSC at site $k \in K$ through a link and interface type $l \in L$, c_{j}^{s} is the cost of a BSC of type $s \in S$ and installed at site $j \in J$ and lastly is the cost of an MSC of type $t \in T$ and installed at site $k \in K$.

The decisions variables v, w, x, y and z are defined as the following. v_i is the value of the BSC site assigned to the BTS $i \in I$. w_j is the value of the MSC site assigned to site $j \in J$. x_j is the value of the BSC type installed at site $j \in J$. y_k is the value of the MSC type installed at site $k \in K$. z_{j}^{l} is the number of links of type $l \in L$ installed from the BSC site $j \in J$ to a MSC.

This network design problem has the following constraints.

1) ***BSC Capacity Constraints***: the number of BTS interfaces that can be installed at a BSC of type x can not exceed the maximum number of interfaces that the BSC can obtain. In addition, the capacity (in circuit) of all BTSs connecting to the BSC must not exceed the capacity that the BSC can obtain.

2) ***MSC Capacity Constraints***: the number of BSC interfaces that can be installed at a MSC of type x can not exceed the maximum interface cards that the MSC can obtain. In addition, the capacity (in circuit) of all BSCs connecting to the MSC must not exceed the capacity that the MSC can obtain.

3) ***Link Capacity Constraints***: the type of each BTS-BSC interface must support the capacity requirement from the connecting BTS, and the type of each BSC-MSC interface must support the capacity requirement from the connecting BSC.

4) ***BTS Type***: the type of each BTS is chosen so that the BTS can support all the traffic (in erlang) from the BTS to a BSC. The traffic (in erlang) from BTS_i to a selected BSC is

$$t_i = \sum_{i' \in I} \left(g_{ii'} + g_{i'i} \right) + \left(g_{ip} + g_{pi} \right) \quad , i \in I \tag{4}$$

where $g_{ii'}$ is the average number of communications per hour from BTS_i to $\text{BTS}_{i'}$, g_{ip} is the average number of communications per hour from BTS_i to the public network, g_{pi} is the average number of communications per hour from the public network to the BTS_i.

5) ***BSC Type***: the type of each selected BSC is chosen so that the BSC can support all the traffic from all BTS pairs (i and i') that communicate via the BSC and another BSC, and the traffic from its own BTSs and the public network (via a MSC) defined as follows.

$$t_j = \sum_{i \in I} \left[(v_i = j) \sum_{i' \in I} (v_{i'} \neq j)(g_{ii'} + g_{i'i}) \right] + \sum_{i \in I} (v_i = j)(g_{ip} + g_{pi}) \quad ;(j \in J) \tag{5}$$

where v_i is a decision variable, which is equal to 1 if BTS_i is connecting to the BSC (i.e. $(v_i = j) = 1$). Otherwise it is zero. $v_{i'}$ is equal to 1 if the $\text{BTS}_{i'}$ is connecting to a different BSC (i.e. $(v_{i'} \neq j) = 1$).

6) ***MSC Type***: the type of each selected MSC is chosen so that the MSC can support all the capacity (in circuit) from all the connecting BSCs and can handle all the interfaces (not exceeding the limit) as discussed in constraint (2).

Based on the minimum-cost network topological design from phase one, we use a network path restoration technique in the second phase as an incremental network design to recover network failure as discussed next.

Phase 2: Incremental Network Design with Path Restoration

In this second phase, we enhance the network reliability by providing a backup path for each working path, while minimum total network cost is considered. We assume that a single link can fail at a time although in some cases, multiple links in a path can simultaneously fail. The backup path is determined between two end-nodes of each protected path to provide an alternate route in the event of the path failure. The network design cost for network path restoration is consisting of the network cost for augmenting the backup links and additional bandwidth on the existing links along the path which may be shared when any path failure occurs.

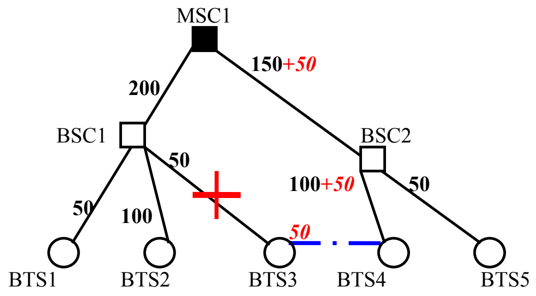


Fig. 3. A snapshot of the network design to protect BTS-BSC link failure with path restoration approach.

Figure 3 shows an example of path restoration approach. Assuming that link failure occurs on the path between BTS3 and BSC1, a path restoration algorithm is applied to find an alternate path between BTS3 and MSC1. In this example, the backup path goes from BTS3 to BTS4, to BSC2 and then MSC1. The additional cost from the backup path is the cost of new link installation between BTS3 and BTS4 and its extra bandwidth along the path BTS3-BTS4-BSC2-MSC1 to cover the failed link. As a result and shown in the figure, a backup link is applied to connect BTS3 and BTS4 with link bandwidth equal to 50, and additional links (path) bandwidth of 50 is applied for the path that connects BTS3 and MSC1. By doing this, BTS3 can connect to the MSC1 via BTS4 and BSC2.

3. Genetic Algorithm Approach

Genetic Algorithm (GA) is a stochastic optimization technique that uses the biological paradigm of evolution. It has a concept where good chromosome has a better potential of being carried to the next generation than the bad chromosome. It uses mathematical principle to indicate which chromosome is better or worse than the others.

Firstly, to use GA we must encode the solution of problem into string called "chromosome" and each string has its unique characteristic inherited by "gene". Each chromosome is evaluated by a fitness function to indicate its potentiality toward the final solutions. The desirable fitness function value is depended on the problem (maximization/minimization).

After that we must generate an initial population (a set of chromosomes), and use the three main operators to find the best solution, as described next.

Step 1 Selection operator: The process of selecting potentially good chromosomes from the current population generation to the next generation.

Step 2 Crossover operator: The process of shuffling any two randomly selected chromosomes to generate the new offspring (like breeding).

Step 3 Mutation operator: The process that randomly selects one chromosome to change one or more genes into random value for generating the new offspring.

Step 4 Repeat steps above until the goal is reached.

Phase 1: GA for Minimum-Cost Network Design

1) Encoding

The string encoding is a set of integers that indicates the relation of the BTS, BSC, and MSC. Suppose in the network design problem, we have 8 BTSs, 4 possible BSCs, and 2 possible MSCs. The corresponding string encoding is displayed in Figure 4. The first 8 integer positions, each with position i ($i \in I$) having the value a_i represents the connection from BTS$_i$ to BSC$_{ai}$. The next 4 integer positions each represents the MSC identifier number, b_j, which the BSC is connected to. The value b_j = -1 if the BSC is not selected.

Let I be the number of BTS, J be the number of BSC, K be the number of MSC. We can encode the string as shown in equation (6).

$$\text{Encoding String} = a_0 a_1 \cdots a_{I-1}\, b_0 b_1 \cdots b_{J-1} \tag{6}$$

where $I = \{0, 1, 2, \ldots, I\text{-}1\}$

$J = \{0, 1, 2, \ldots, J\text{-}1\}$, $K = \{0, 1, 2, \ldots, K\text{-}1\}$

$i \in I$, $j \in J$; $a_i \in J$; $b_j \in K$ or $\{-1\}$;

1	1	1	1	3	3	3	4	1	-1	1	1

Fig. 4. An example of string encoding.

2) Initial Population

We set the initial population by randomly generating a set of chromosomes consisting of genes with uniform random number, and calculate their fitness value according to the fitness function.

3) Selection

The chromosomes or population are sorted by their fitness values. The top 10 % of population with best fitness values (low total cost) are preserved to the next generation, 80 % of population are selected for the crossover process, and the remaining 10 % of population are new chromosomes which are randomly generated.

4) Crossover

We select a pair of chromosomes from the current population for a crossover, to produce two new offspring chromosomes. One parent chromosome comes from the top 10% and another parent chromosome can be any chromosome.

In each crossover, 10% of the genes are exchanged between the two parent chromosomes. The exchanged genes are randomly selected. The next population generation consists of the best 10 % of the current population generation with the minimum cost values, 80 % from the crossover process, and the rest 10 % comes from randomly generated population (alien).

5) Mutation

The offsprings from the crossover process are mutated with 25% rate. Only 1-2 genes in a chromosome are mutated randomly at a time. The resulted chromosomes are combined and considered as the chromosomes in the current population generation.

Phase 2: GA for Path Restoration Design

1) Encoding for Path Restoration Design

The network design result from phase one is now considered for network path restoration design using GA to find a backup path for every existing/working path so that any single path failure event can be protected by a backup path.

Considering a network with n total nodes (i.e. the number of BTS, BSC and MSC nodes) having the total network paths equal to L, the network design problem now can be string encoded for the next GA processing shown in Figure 5 as an example. The value of L is equal to the number of BTSs. The value of each gene, P_r, represents the choice of pre-calculated backup paths chosen for path r. In our experiment, the best 10 backup paths (with minimum cost) are considered for any single path failure, so that P_r can be an integer in the range of 0 to 9.

From the figure, path P_1 has a back up path number 2, path P_2 has a back up path number 7 and so on.

P_1	P_2	P_3	P_4	P_5	P_6	...	P_r
2	**7**	**5**	**4**	**1**	**9**	**...**	**4**

$r = 1, 2, \ldots, L$; $P_r = 0, 1, 2, \ldots, 9$;

Fig. 5. String encoding for network design with path restoration.

In this phase two, network path restoration design, we use similar GA setting operators (except crossover process) as those from phase 1.

2) Crossover for Path Restoration Design

In each crossover, 15% of the genes are exchanged between the two parent chromosomes. The exchanged genes are randomly selected. The next population generation consists of the best 10 % of the current population generation with the minimum cost values, 80 % from the crossover process, and the rest 10 % comes from randomly generated population (alien).

4. Computation Results

To test our two-phase network design approach, we consider the network design with 3 BTS types, 3 BSC types and 3 MSC types. Their descriptions are presented in Tables 1-3. We consider DS_1 links for the connection from the BTS layer to the BSC layer. DS_1 and DS_3 link types are considered for the connection from the BSC layer to the MSC layer. The interface costs and link costs are presented in Tables 4-6.

Various test problems are considered with a number of BTS, BSC and MSC. Their locations are randomly generated with a uniform distribution in a square region of 100 kilometers. The type of each BTS is randomly selected. The demand between each pair of BTSs and between the BTSs and the public network was generated randomly in the interval [0,0.2] erlang with a uniform distribution. The call blocking rate is 2%.

JAVA programming language is selected for our GA implementation. The code is run on a Pentium 4 PC (1.73 GHz and 512MB of RAM). For each problem size, three randomly generated networks with varying communication demands between 0-0.2 Erlang are investigated. Each result is an average of minimum cost from the three tested networks, each with 5-10 simulation runs.

Table 1. Feature of the BTS types

	Type A	Type B	Type C
Capacity (circuits)	96	288	576
Number of BTS DS_1 interfaces	1	3	6

Table 2. Costs of the BSC types (including the installation costs)

	Type A	Type B	Type C
Switch fabric capacity (circuits)	5,000	10,000	15,000
Maximum number of BTS interfaces	15	30	60
Maximum number of MSC interfaces	15	30	60
Cost ($)	50,000	90,000	120,000

Table 3. Costs of the MSC types (including the installation costs)

	Type A	Type B	Type C
Switch fabric capacity (circuits)	100,000	200,000	300,000
Maximum number of BSC interfaces	50	100	150
Cost ($)	200,000	350,000	500,000

Table 4. Costs of the interface types (including the installation costs)

Interface type	Capacity (circuits)	Cost ($)
DS_1	96	500
DS_3	2,688	2,500

Table 5. Costs of the BTS-BSC links (including the installation costs)

BTS type	Number of the DS_1	Capacity (circuits)	Cost ($/km)
A	1	96	2,000
B	3	288	3,000
C	6	576	4,000

Table 6. Costs of the BSC-MSC links (including the installation costs)

Link type	Capacity (circuits)	Cost ($/km)
DS_1	96	2,000
DS_3	2,688	4,000

Table 7. The result comparisons between Brute Force Search approach and GA

Problem Size (bts_bsc_msc)	Brute Force Search		Genetic Algorithm	
	Cost	CPU Time (sec)	Best Cost	CPU Time (sec)
5_3_2	683,206.4	1.0	683,206.4	1.0
8_4_2	989,271.5	9.0	989,271.5	3.5
10_4_2	1,066,101.6	206.2	1,066,101.6	3.0
12_4_2	1,211,395.8	5,016.2	1,211,395.8	4.2
22_9_5	1,664,949.6	84,387.0	1,664,949.6	63.0
25_20_10	1,557,714.4	702,733.2	1,643,869.9	135.8
30_15_8	2,237,392.4	714,929.2	2,343,875.9	69.7

Initially, we consider solving relatively small to moderate network problem sizes with the GA and a Brute Force Search approach where the results are compared as presented in Table 7. The results show that the GA can give the optimal or near-optimal network costs as guaranteed by the Brute Force Search approach. Moreover, the GA requires significantly less CPU time than the traditional search approach. The network costs obtained from GA differ less than 6% from the network costs obtained from Brute Force Search approach.

In Figures 6-7, we present graphical results of applying GA to minimum-cost network (tree topology) design in phase one and path restoration design in phase two. The network costs and CPU time obtained from solving different tested network sizes are presented. The results show that our GA approach can be used and easily scale to solve both minimum-cost network design and path restoration network design with large-problem sizes within reasonable computational time. Note that adding reliability into the network to recover network services in the event of any single path failure requires extra cost (about 30 percent) for backup links/paths and spare capacity.

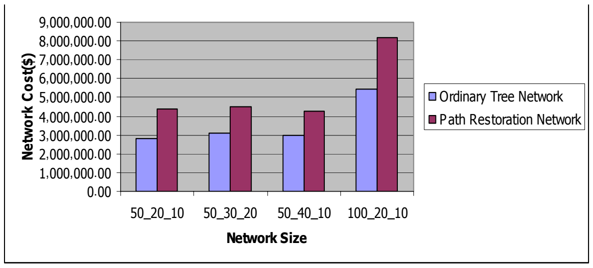


Fig. 6. Network cost comparison between phases 1 & 2.

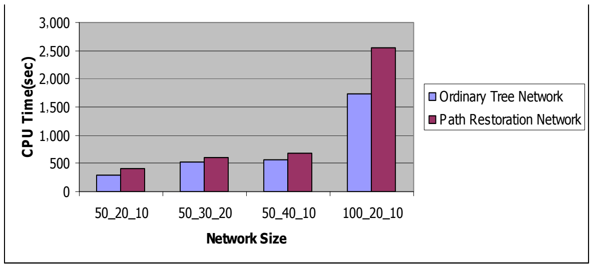
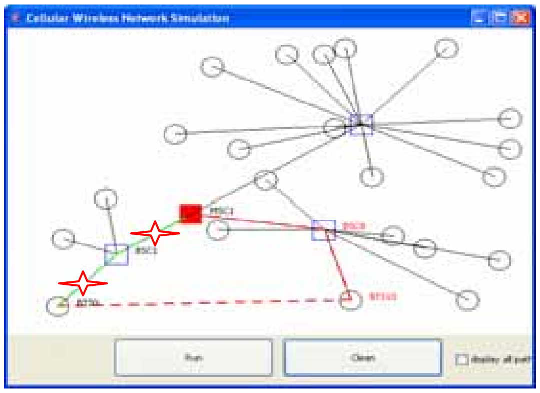


Fig. 7. CPU time comparison between phases 1 & 2.



Fi.g. 8. A snapshot of network design when a path failure occurs

Figure 8 captures a snapshot of the network design when a certain path experiences failure, and the network is still connected with the backup paths obtained from path restoration mechanism. From the figure, the dashed line (path BTS0→BTS10→ BSC8→MSC1) identifies the backup path when the marked path (path BTS0→BSC1→MSC1) encounters failure. The circles represents the BTSs, while the squares and the filled-square represent the BSCs and the MSC, respectively.

5. Conclusion

A survivable wireless backhaul network design problem is reasonably presented with a two-phase network design approach and effectively solved with Genetic Algorithm (GA). The GA heuristic technique can be applied to both the minimum-cost network design in the first phase and incremental network design with network path restoration in second phase of our adopted two-phase reliable network design approach. From our experiments, the GA heuristic method can be efficiently used in solving cellular wireless network design problem, as verified by a brute force search approach where all possible solutions are considered. The CPU time consumed by the GA is significantly less than those consumed by traditional searching algorithms such as a brute force search technique. Numerical results show that the GA can give the optimal or near-optimal solutions as guaranteed by the Brute Force Search approach for relatively small to moderate network problem sizes. For large problem sizes, the traditional searching algorithm is prohibited due to the complexity of the problem while the GA heuristic method can easily scale to find good solutions within a reasonable amount of computational time.

6. References

[1] A. Snow, et al., "Reliability and survivability of wireless and mobile networks," *IEEE Computer*, vol. 33, pp. 49-55, July, 2000.
[2] D. Tipper, T. Dahlberg, H. Shin, and C. Charnsripinyo, “Providing fault tolerance in wireless access networks,” *IEEE Communication Magazine*, vol. 40, no. 1, pp. 58-64, January 2002.
[3] D. Tipper, et al., ‘Survivability analysis for mobile cellular networks,” *CNDS2002*, Texas, Jan. 27-31, 2002.
[4] D. Alevras, et al., “Survivable mobile phone network architectures: models and solution methods,” *IEEE Comm. Magazine*, vol. 36, no. 3, pp. 88-93, March 1998.
[5] Amitava Dutta and Peter Kubat, “Design of partially survivable networks for cellular telecommunication systems,” *European Journal of Operational Research,* vol. 118, pp. 52-64, 1999.
[6] L. A. Cox, Jr., and Jennifer R. Sanchez, “Designing least-cost survivable wireless backhaul networks,” *Journal of Heuristics*, vol. 6, pp. 525-540, 2000.
[7] C. Charnsripinyo and D. Tipper, “Topological design of survivable wireless access networks,” DRCN2003, Canada, 19-22 October, 2003.
[8] Y. Pomerleau and S. Chamberland and Gilles Pesant, “A constraint programming approach for the design problem of cellular wireless networks,” *Canadian Conference on IEEE*, Vol. 2, pp. 881-884, 2003.
[9] P. Leesutthipornchai, et al., “Cellular wireless network design with genetic algorithm,” ECTI-CON 2007, Thailand.
[10] D.W. Coit and A.E. Smith, “Reliability Optimization of Series-parallel System Using Genetic Algorithm,” *IEEE Trans. Reliability,* Vol.5, pp. 4676-4681, 1996.
[11] K. Suteeca and N. Wattanapongsakorn, “Reliability optimization of communication network design using genetic algorithm,” ITC-CSCC, 2006.
[12] Veerasamy J. et al., “Effect of traffic splitting on link and path restoration planning”, IEEE Inter Conference, pp.1867-1871, 1994.
[13] Veerasamy J., et al., “Effect of traffic splitting on link and path restoration planning”, GLOBECOM, 28 Nov.-2 Dec, pp. 1867-1871, 1994.
[14] Ramamurthy S. et al., “Survivable WDM mesh networks II Restoration”, IEEE Inter Conference, 6-10 June, Vol. 3, pp. 2023-2030, 1999.
[15] Sahasrabuddhe L. et al., “Path vs. subpath vs. link restoration for fault management in IP-over-WDM networks: performance comparisons using GMPLS control signaling”, *IEEE Communications Magazine*, Nov. 2002, Vol. 40, Issue 11, pp. 80-87, 2002.