We are IntechOpen,
the world's leading publisher of
Open Access books
Built by scientists, for scientists

6,900
Open access books available

186,000
International authors and editors

200M
Downloads

154
Countries delivered to

Our authors are among the

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities

CLARIVATE ANALYTICS
BOOK CITATION INDEX
INDEXED

WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index
in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com

Multiple Tabu Search Algorithm for Solving the Topology Network Design

Kanyapat Watcharasitthiwat, Saravuth Pothiya and Paramote Wardkein
Department of Telecommunication Engineering,
King Mongkut's Institute of Technology Ladkrabang
Thailand

1. Introduction

A communication network can be illustrated by set of nodes (or switches) and links (or arcs) where all nodes are connected by links. The typical communication network structure is composed of two levels. The first one is backbone network and the second one is local access network (LAN). The backbone network dedicate for delivery of information from source to destination (end to end) using its switching nodes. The LAN is typically centralized system that allows users to access hosts or local servers. This chapter focuses only on the backbone network design considered as distributed network.

The advent of low cost devices has led to explosive growth in communication networks. The topology network design is a part of network planning which finds a topology in order to satisfy any constraints. One of the major advantages of distributed network over the centralized system is their flexibility to improve system reliability. The reliability of a system depends not only on the reliability of its nodes and links, but also on the topology of network. A completely connected network has the highest network reliability while the simple loop (ring) network has the lowest network reliability.

Many researchers have studied the designing of the network when considering system reliability as a constraint or an objective. The reliable network design problems state as source-sink and all-terminal network reliability (also known as overall reliability). The problem of optimal topology network design that selects the links connections that either maximizes reliability or minimizes cost can formulate as a combinatorial problem.

This problem is a well known NP-hard problem and very difficult to solve. For such problem, many researchers have studied with enumerative-based and heuristic method. Jan (1993) developed an algorithm using decomposition based on brand and bound to minimize link cost of communication network subject to reliability constraint. Aggarwal et al. (1982) employed greedy heuristic approach to maximize reliability given a cost constraint for networks with different reliability of links and nodes. Pierre et al. (1995) also used simulated annealing to find the optimal design for packet switching networks where considering the delay and capacity, but reliability was not. For the network design, Kumar et al. (1995) developed a genetic algorithm (GA) considering diameter, average distance, and communication network reliability and applied it to four test problems of up to nine nodes. Darren & Smith (1998) presented a GA approach for minimum cost network design problem

with alternative link reliabilities and all-terminal network reliability constraint. Furthermore, Glover et al. (1991) used tabu search (TS) algorithm to choose the topologies of network when considering cost and capacity, but not reliability. Another work of TS algorithm, Beltran & Skorin-Kapov (1994) used TS algorithm to design reliable networks by searching for the least cost spanning two-tree, where the two-tree objective was a coarse surrogate for reliability.

Recently, modern heuristic optimization techniques such as simulated annealing (SA), GA, evolutionary programming (EP), TS algorithm, artificial neural network (ANN), particle swarm optimization (PSO) and ant colony optimization (ACO) have been paid much attentions by many researchers because of their abilities to find an almost global optimal solution. Among of them, the TS algorithm expected as one of the advanced search technique. The TS algorithm is able to escape from local optimal and fast converge to global optimum. However, a conventional TS algorithm might have a problem of reaching the global optimum solution in a reasonable computational time when the initial solution is far a way from the region where optimum solution exists.

Nowadays, the personal computer has more high-speed computation. To solve the large-scale problem, several computers may use for computation at the same time. This method called "parallel searches". On the other hand, the "multiple search" implemented in this chapter is executed by only one personal computer. The multiple searches help to find the promising region where the global optimum solution exists. This idea applies for improvement the performance of a conventional TS algorithm.

Pothiya et al. (2006) developed the multiple tabu search (MTS) algorithm and applied this algorithm to design the optimal fuzzy logic PI controller. After that, MTS applied to solve the economic dispatch problem with generator constraint (Pothiya et al., 2008). The MTS is the execution of individual TS algorithm simultaneously by only single personal computer. The MTS introduces the additional salient mechanisms for improvement of search process, i.e. initialization, adaptive searches, multiple searches, replacing and restarting process. The feasibility study of the MTS for this chapter demonstrated for solving the topology network design when considering both reliability and cost. The optimized results by the MTS compared to those obtained by the conventional approaches such as GA, TS and ACO in terms of solution quality and computational efficiency.

The remainders of this chapter organize as follows. Section 2 describes the problem formulation, assumptions of system, and reliability calculation. Section 3 presents the principle of MTS algorithm for solving the topology network design. Section 4 illustrates the examples and simulation results. Finally, Section 5 contains the discussion and conclusion.

2. Statement of the problem

Notations

L	Set of possible link
l_{ij}	Option of each link
d_{ij}	Distance between node i and node j
N	Set of given node
n	Number of node
$p(l_k)$	Reliability of link option

$c(l_k)$ Unit cost of link option
x Architecture of network design
$C(x)$ Total cost of network design
$R(x)$ Reliability of network design
R_o Minimum network reliability constraint

2.1 Problem formulation

In the source-sink and all-terminal reliability network design problem, there are a set of N nodes with specified topology, which can be found from real world networks or will be interpreted as Euclidean distance between coordinates on a plane. It is noted that this distance is not an existing distance. It only represents some cost of connection between two nodes regardless of type of connection. The network nodes are assumed to be fully reliable or assumed not to fail under any circumstances. There are a set of L links, which connect all nodes in N. In this problem, it is assumed that every possible links are included in L, i.e., a fully connected network. Hence, for any (n_i, n_j) pair of elements of N there exists the possibility of l_{ij} element of L such that l_{ij} connects n_i and n_j. In addition, it is assumed that a link is bi-directional if l_{ij} is turned on it lets n_i communicate with n_j and vice versa. It is also assumed that there is only one link per location. Therefore, all links fail independently and repairing link is required if any link fails. The total number of possible links in single design is:

$$|L| = \frac{|N| \cdot |N-1|}{2} \tag{1}$$

Generally, one link can possibly have more than two states. Thus a candidate solution, x, to this problem then consists of a selected number of links $l_{ij} = k$ where k is the level at which those links connect nodes n_i and n_j. The mathematical formulation for this problem when minimizing cost is subject to a minimum network reliability constraint is:

$$\text{Minimize } C(x) = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} c_{i,j} \cdot l_{i,j} \cdot d_{ij} \tag{2}$$

$$\text{Subject to } \quad R(x) \geq R_0 \tag{3}$$

The cost of a specific architecture x is given by $C(x)$ and the reliability of x is given by $R(x)$. The problem find an x so that $R(x) \geq R_0$ base on the following assumptions:

1. The locations of all the network nodes are given.
2. The cost c_{ij} and the operation probability p_{ij} of each link (i, j) are fixed.
3. Each link is bi-directional.
4. No redundant link is allowed in the network.

2.2 Reliability calculation

The problem of calculating or estimating the reliability of a network is another active area of research related to the economic network design problem. There are two main approaches

the exact calculation through analytic methods (Ball & Van-Slyke 1977, Jan 1993) and the estimation calculation through Monte Carlo simulation (Fishman 1986, Ramirez-Marquez & Coit 2005). For the all-terminal network reliability problem, efficient simulation is difficult because these methods generally lose efficiency as a network approaches a fully connected state. There are also upper and lower bound expressions for network reliability (Shinmori & Ishii 1995), however these are too loose to be effective surrogates in the all-terminal design process. Furthermore, many bounding procedures and improved efficiency simulations depend on the assumption that all arcs have the same reliability, which is relaxed in this research. Therefore in this chapter, either the network reliability was calculated exactly using a backtracking procedure or a classic Monte Carlo procedure. Due to the computationally tractable size, a backtracking algorithm is used to correctly calculate the system reliability, $R(x)$, for the problems in this chapter. The outline of the backtracking algorithm is given as follow:

Step 0: (Initialization). Mark all links as free; create a stack which is initially empty.

Step 1: (Generate modified cut-set)

(a) Find a set of free links that together with all inoperative links will form a network-cut.

(b) Mark all the links found in 1(a) inoperative and add them to the stack.

(c) The stack now represents a modified cut-set; add its probability into a cumulative sum.

Step 2: (Backtracking)

(a) If the stack is empty, end.

(b) Take a link off the top of the stack.

(c) If the link is inoperative and it is operative a spanning tree of operative links exists, then mark it free and go to 2(a)

(d) If the link is inoperative and the condition tested in 2(c) does not hold, then mark it operative, put it back on the stack and go to Step 1.

(e) If the link is operative, then mark it free and go to 2(a).

It should be mentioned that the algorithm above is for all-terminal reliability and needs to be modified for use in a source-sink design problem as given below:

Step 0: (Initialization). Mark all links as free; create a stack that is initially empty.

Step 1: (Generate modified cut-set)

(a) Find a set of free links that together with all inoperative links will form a source-sink cut.

(b) Mark all the links found in 1(a) inoperative and add them to the stack.

(c) The stack now represents a modified cut-set; add its probability to a cumulative sum.

Step 2: (Backtracking)

(a) If the stack is empty, end.

(b) Take a link off the top of the stack.

(c) If the link is inoperative and if when made operative, a path from the source to the sink exists, then mark it free and go to 2(a).

(d) If the link is inoperative and the condition tested in 2(c) does not hold, then mark it operative, put it back on the stack and go to Step 1.

(e) If the link is operative, then mark it free and go to 2(a).

For larger networks, Monte Carlo simulation is used to accurately estimate network reliability. The network is simulated t times given the design and the link reliabilities:

Initialize $i = 0$, $c = 0$

Step 0: While $i < t$ Repeat.

Step 1: Randomly generate network.

    (a) $i = i + 1$

Step 2: Check to see if the network forms a spanning tree.

    (a) If the network forms a spanning tree then $c = c + 1$ go to Step 0.

    (b) If the network does not form a spanning tree go to Step 0.

Step 3: $R(x) = c / t$

When the need to simulate a network's reliability arises, other issues become important. One of these issues is whether or not the estimator is biased. The other issue is the variance of the estimate. Every referenced Monte Carlo technique is an unbiased estimator where the variance of the Monte Carlo method described above is:

$$\text{Var}\,(\,R(x)\,) = \frac{R(x)(1 - R(x))}{t} \tag{4}$$

To get a more accurate reliability estimate, t must be high value.

Note: Var is the variance of variable.

3. Multiple tabu search algorithm

The MTS algorithm is the execution of individual TS algorithm at the same time by a single personal computer. Here, the individual TS algorithm and the MTS algorithm are explained.

3.1 Principle of tabu search

1) Overview

In general terms, a conventional TS algorithm is an iterative search that starts from an initial feasible solution and attempts to determine a better solution in the manner of a hill-climbing algorithm. The TS has a flexible memory to keep the information about the past steps of the search. The TS uses the past search to create and exploit the better solutions (Glover 1989, Glover 1990). The main two components of TS algorithm are the tabu list (TL) restrictions and the aspiration criterion (AC).

2) Tabu list restrictions

In order to prevent cycling, repeated search at the same solution, a TL is introduced. The TL stores a set of the tabu (prohibition) moves that can not be applied to the current solution. The moves stored in TL called *tabu restrictions* and used for decreasing the possibility of cycling because it prevents returning in a certain number of iterations to a solution visited recently.

In this chapter, the size of TL is $n \times 3$ (row × column), n is a number of neighborhoods around current solution. In the TL, the first column is used for storing the moves, the second column is the frequency of a move direction, and the last column is the recency (time to keep solutions) of a move (Bland & Dawson 1991).

3) Aspiration criterion

Another key issue of TS algorithm arises when all moves under consideration have been found to be tabued. The tabu status of a move is not absolute, but it can be overruled if

certain conditions are met and expressed in the form of AC. If appropriate aspiration criterion is satisfied, the move will be accepted in spite of the tabu classification. Roughly speaking, AC is designed to override tabu status if a move is 'good enough' (Bland & Dawson 1991).

4) Stopping criterion

There are several possible conditions for stop searching. Here, the stopping search is used if any of the following two conditions are satisfied: first, the accuracy of the best solution is lower than the expected value, and the second, the maximum allowable number of iterations is reached.

5) General tabu search algorithm

To solve a combinatorial optimization problem by tabu search, the basic idea is to choose randomly a feasible solution and attempt to find a best neighbor to current solution. A move to this neighbor is performed if either it does not belong to the TL or, it passes the AC test. During these procedures, the best solution is always updated and stored aside until the stopping criterion is satisfied. The following notations are used through the description of the TS algorithm for a general combinatorial optimization problem:

X : the set of feasible solutions

x : the current solution, $x \in X$

x_b : the best solution reached

x_{nb} : the best solution among of trial solutions

$E(x)$: the objective function of solution x

$N(x)$: the set of neighborhood of $x \in X$

TL : tabu list

AC : aspiration criterion

The procedure of TS algorithm is described as follows:

Step 0: Set TL as empty and AC to be zero.

Step 1: Set iteration counter $k = 0$. Select an initial solution $x \in X$, and set $x_b = x$.

Step 2: Generate a set of trial solutions neighborhood of x. Let x_{nb} as the best trial solution.

Step 3: If $E(x_{nb}) > E(x_b)$, go to Step 4, else set the best solution $x_b = x_{nb}$ and go to Step 4.

Step 4: Perform the tabu test. If x_{nb} is NOT in the TL, then accept it as a current solution, set $x = x_{nb}$, and update the TL and AC and go to Step 6, else go to Step 5.

Step 5: Perform the AC test. If satisfied, then override the tabu state, set $x = x_{nb}$, update the AC.

Step 6: Perform the termination test. If the stopping criterion is satisfied then stop, else set $k = k + 1$ and go to Step 2.

3.2 Multiple tabu search for solving topology network design

Although the TS algorithm is able to escape from local optimal and fast converge to global optimum. It might have a problem with reaching the global optimum solution in a reasonable computation time when an initial solution is far away from the region where the optimum solution exists. The convergence speed of TS algorithm depends on an initial solution. The convergence speed can be improved by introducing a multiple structure into the algorithm.

The MTS algorithm uses several initial solutions to increase the probability of reaching the region where the optimum solution exists. The procedure of the MTS algorithm is depicted in Fig. 1, which consists of several independent conventional TS algorithms (TS#1, TS#2,
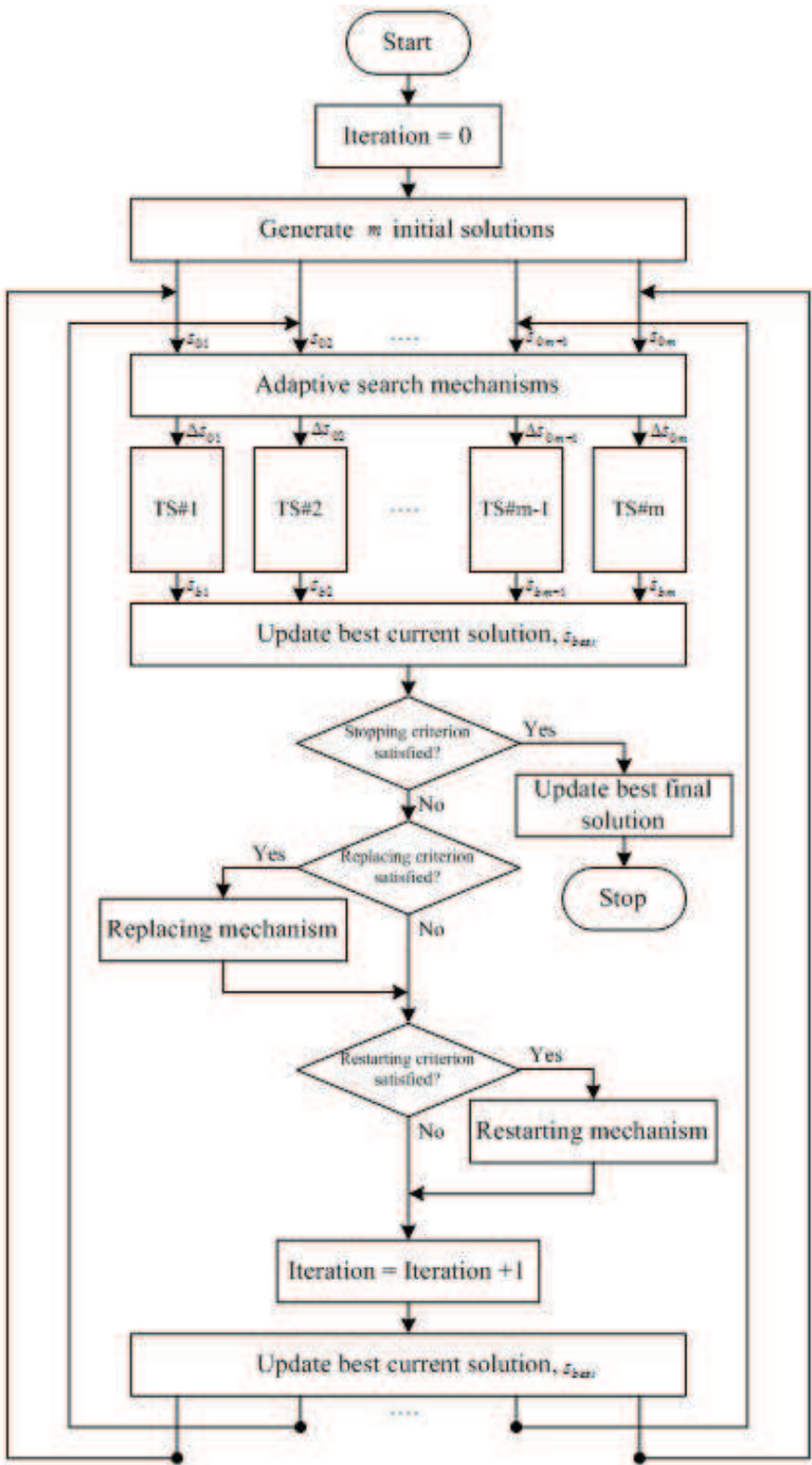

Fig. 1 Procedure of Multiple Tabu Search algorithm

..,TS#m). Furthermore, the additional mechanisms namely, *initialization*, *adaptive searches*, *multiple searches*, *replacing* and *restarting process* help to improve the search process in terms of both solution quality and computational time. The additional mechanisms are explained as follows:

1) Initialization

To improve searching, the MTS algorithm starts to search from several initial solutions which are different from the TS algorithm. In fact, the starting with several initial solutions has the probability of reaching the optimum solution higher than the single initial solution. This mechanism helps MTS algorithm to converge quickly to the global optimum solution.

In the initialization process, several initial solutions are created randomly for individual TS algorithm. In this chapter, the structure of a solution for topology network problem is composed of a set of links. Therefore, the initial solutions of individual TS algorithm at iteration #0 can be represented as the matrix of X_i^0, $i=1,\dots,m$ where m is the number of multiple TS algorithms. Note that it is very important to create a set of solution satisfying the equality and inequality constraints.

2) Adaptive searches

The step size is the range of variance at current solution which is the important factor for search process. Accordingly, the step size should be chosen appropriately. In general, this value is fixed. Low value of the step size can increase the accuracy of solution. But it takes a long computational time. On the other hand, high value of the step size is used for decreasing the computational time. But the searched result may not reach the global optimum. Consequently, the adaptive search mechanism has been developed to adjust suitably the step size during the search process. This mechanism helps to increase the computational speed and the accuracy of solution.

3) Multiple searches

Nowadays, the personal microcomputer has high speed computation. To solve the large-scale problem, several computers may be used at the same time. This method is called *parallel searches*. For *multiple searches*, they are executed by only one personal microcomputer. The multiple searches help to find the promising region where the global optimum solution exists.

The MTS algorithm uses the multiple searches mechanism to enhance its capacity. The multiple searches mechanism is executed by using only a personal microcomputer. Each step of searching for finding the better solution is used in the procedure of TS algorithm which is given in section 3.1 (5). The sequence of execution starts from TS#1 to TS#m. Fig. 2 illustrates the sequence of execution for finding better solutions.

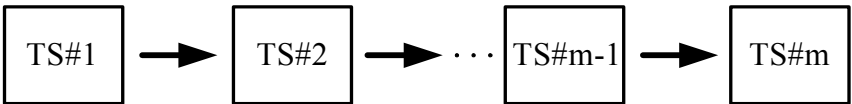TS#1 → TS#2 → ⋯ TS#m-1 → TS#m

Fig. 2 Sequence of execution

4) Replacing

After the search process satisfies with the condition for replacing, all independent TS algorithms are stopped. The condition for replacing depends on the size of problem. The experiment can be found the appropriate value. The replacing mechanism is used for

comparison and exchanging the solutions which are found by these TS. Then, the replacing mechanism generates the best initial solutions for the next search. Like the crossover of GA, the MTS algorithm performs with replacing mechanism for improvement the solution. Here, the replacing is set to apply to each individual TS every 30 iterations. As a result, new solutions are generated for the next iteration.

5) Restarting process

When the search is stroke on the local solution for a long time and the procedure of TS algorithm can not escape from the local solution. The restarting process is applied to keep searching and finding a new solution. The restarting mechanism is applied when the search is stroke on the local solution for 20 iterations and the procedure of TS algorithm can not escape from local solution. The procedure of restarting mechanism is almost the same as the initialization mechanism.

4. Test problems and results

Three test problems were studied and each exhibits a different aspect of the MTS approach. These four link types, shown in Table 1 were used for all test problems. To examine the performance of the proposed MTS method, it is then compared the results with those of GA, TS and ACO. The simulation results shown in this chapter are simulated using MatLab® program on a Pentium 4, 2.66 GHz 512 DRAM personal computer.

Connection Type	Reliability	Cost ($)/Unit Distance
0 (not connected)	0.00	0
1	0.85	8
2	0.90	10
3	0.95	14

Table 1 Link Unit Costs and Corresponding Reliabilities

4.1 Application test problems

1) Test Problem 1: Five nodes network

A communication network has five nodes with multiple levels as show in Fig. 3. The problem is based on Jan (1993) but expanded by changing the links from a simple on/off state to one of four possible states using link costs as distances and using the unit costs and reliabilities shown in Table 1. Since this problem has $k = 4$ levels its search space size is $4^{(5 \times 4)/2} = 1,048,576$. This problem was considered at six different system reliability constraints and the backtracking algorithm is employed to calculate $R(x)$ exactly. The cost matrix of links is given as follow:

$$[l_{ij}] = \begin{bmatrix} - & 32 & 54 & 62 & 25 \\ & - & 34 & 58 & 45 \\ & & - & 36 & 52 \\ & & & - & 29 \\ & & & & - \end{bmatrix} \tag{5}$$
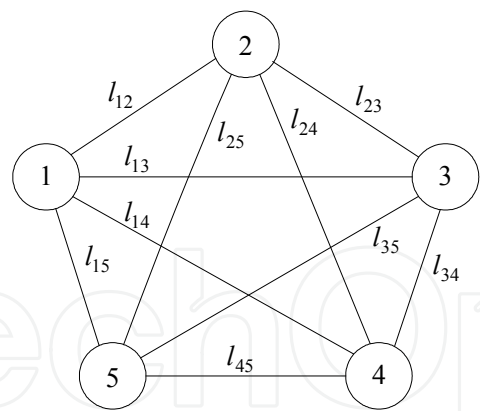
Fig. 3 Five nodes communication network system

2) Test Problem 2: Source-Sink network

This problem shows that the flexibility of the approach by considering economic design of 18 links source-sink network as shown in Fig. 4. This problem is taken from the literature (Jan 1993) and has 6.9×10^{10} possible architectures, thus precluding enumeration to identify the optimal design. The distance matrix for this problem appears in Table 2. This problem was considered at six different system reliability constraints and uses the backtracking algorithm to calculate the exact value of $R(x)$.
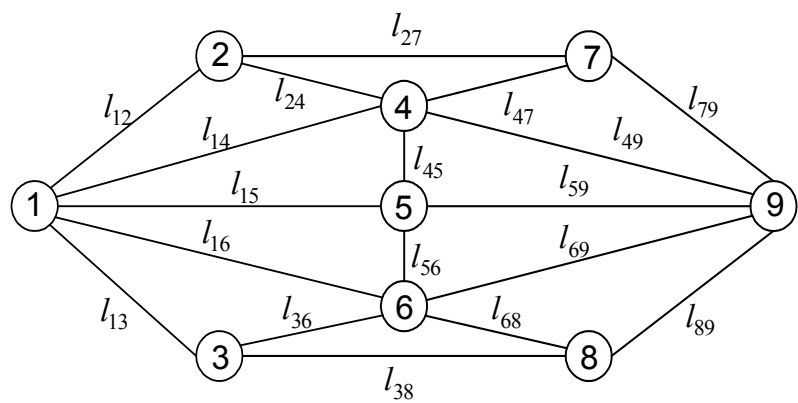
Fig. 4 Source-Sink network (Source node is 1 and Sink node is 9)

	2	3	4	5	6	7	8	9
1	58	63	60	63	58	-	-	-
2		42	-	-	-	60	-	-
3			20	-	-	42	-	63
4				20	-	-	-	60
5					42	-	42	63
6						-	60	-
7							-	58
8								58

Table 2 Distance matrix for Source-Sink network

3) Test Problem 3: 19 Districts in Bangkok, Thailand

To demonstrate the applicability of this work on a realistic application, the network of 19 districts in Bangkok is therefore applied for an example of scaled-up problem. This 19 node problem was constructed by selecting districts in Bangkok as seen in Fig. 5 and computing the Euclidean distances between them using their coordinates as shown in Table 3 and using the four links choices from Table 1. The search space of this problem is 8.959×10^{102}. Besides the scale-up issue, the difference between this problem and the other design problems is that this problem illustrates the flexibility of the MTS by reversing the constraint and the objective function. This problem was considered at six different system reliability constraints and uses the Monte Carlo simulation is used to accurately estimate network reliability.
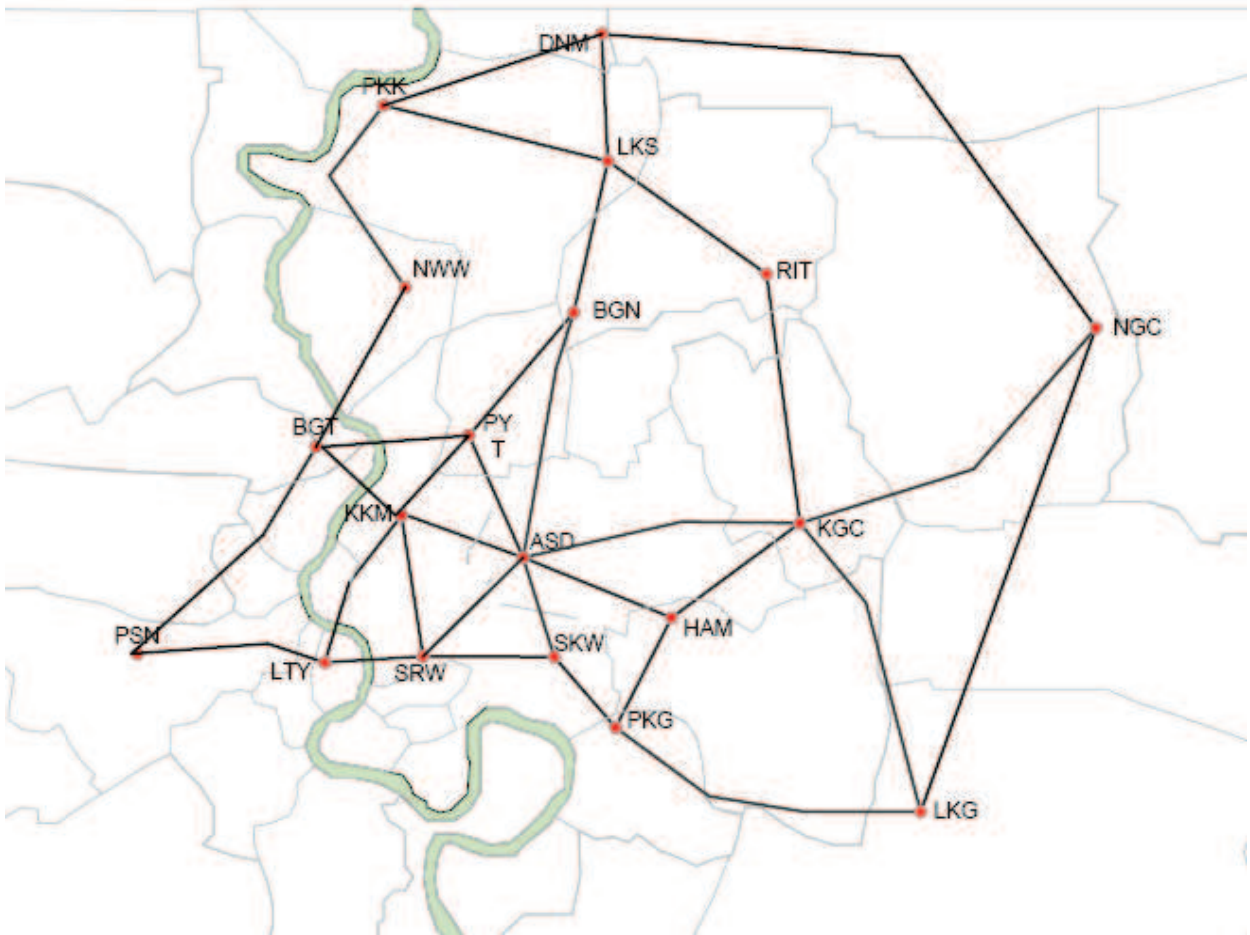

Fig. 5 Network of 19 districts in Bangkok, Thailand

4.2 Simulation results

Four methods, which are MTS, GA, TS and ACO, are employed to examine three test problems with six different system reliability constraints. To perform 30 trials, the simulation results in terms of maximum, average and minimum cost, the standard deviation, the average computational time, and percentage of get near optimum solution for test problem 1, 2 and 3 are presented in Table 4, 5 and 6, respectively. Obviously, the

proposed MTS approach always provided better solution than other methods, thus resulting in the higher quality solution.

In order to demonstrate the efficiency of the proposed MTS method, the distribution outlines of the best solution of each trial are considered. Fig. 6 and 7 demonstrate the distribution outlines of best solution of each trail for test problem 1 and 2, respectively. Most of the costs obtained by the proposed MTS method were lower than those of the compared methods, that the efficiency of the proposed MTS method is superior to other methods.

	LKS	PKK	NWW	RIT	BGN	NGC	BGT	PYT	KKM	KGC	ASD	PSN	LTY	SRW	SKW	HAM	PKG	LKG
DNM	8.10	19.40	27.88	17.50	12.80	45.41	29.92	20.30	30.20	25.90	36.59	48.62	36.61	38.90	40.47	31.90	38.00	66.06
LKS		11.30	19.78	9.40	4.70	45.28	21.82	12.20	22.10	17.80	28.49	40.52	28.51	30.80	32.37	23.80	29.90	41.75
PKK		-	8.48	20.70	16.00	62.01	25.80	25.58	26.08	29.10	39.79	44.50	32.49	32.18	43.67	45.49	49.37	49.75
NWW			-	29.18	24.48	70.49	17.32	17.10	17.60	19.35	25.10	36.02	24.01	23.70	28.98	30.80	34.68	54.27
RIT				-	14.10	35.88	31.22	21.60	31.50	8.40	20.10	49.92	37.91	30.00	26.00	14.20	20.30	32.35
BGN					-	52.96	17.12	7.50	17.40	25.48	23.79	35.82	23.81	34.69	27.67	29.49	33.37	52.96
NGC						-	56.80	47.18	57.08	27.48	39.18	65.74	55.88	50.08	43.06	33.48	39.58	20.65
BGT							-	9.62	5.32	28.97	17.62	18.70	11.96	17.76	12.92	23.32	18.62	38.21
PYT								-	9.90	19.35	8.00	28.32	16.31	16.00	11.88	13.70	17.58	37.17
KKM									-	20.90	9.20	16.27	6.41	6.10	7.60	14.90	13.30	32.89
KGC										-	11.70	38.26	28.40	22.60	12.10	6.00	12.10	23.95
ASD											-	26.56	16.70	10.90	3.88	5.70	9.58	29.17
PSN												-	9.86	15.66	24.96	36.76	30.66	50.25
LTY													-	5.80	15.10	26.90	20.80	40.39
SRW														-	9.30	21.10	15.00	34.59
SKW															-	11.80	5.70	25.29
HAM																-	6.10	25.69
PKG																	-	19.59

DNM	Donmuang	LKS	Laksi	PKK	Pakkret
NWW	Ngamwongwan	RIT	Ram-intra	BGN	Bangken
NGC	Nongchok	BGT	Bangplad	PYT	Paholyothin
KKM	Krungkasem	KGC	Klongchan	ASD	Asok Dindeang
PSN	Pasicharoen	LTY	Latya	SRW	Surawong
SKM	Sukhumwit	HAM	Huamark	PKG	Phrakanong
LKG	Ladkrabang				

Table 3 Distance matrix of 19 districts in Bangkok, Thailand

Reliability	Optimal Cost	Configuration	Algorithm	Max. Cost	Average Cost	Min. Cost	Standard Deviation	% Get Optimal	CPU Time
0.999	5498	3323323333	GA	5498	5498	5498	0.00000	100	12.49
			TSA	5498	5498	5498	0.00000	100	9.93
			ACO	5498	5498	5498	0.00000	100	7.11
			MTS	**5498**	**5498**	**5498**	**0.00000**	**100**	**3.15**
0.995	4250	3303222313	GA	4250	4250	4250	0.00000	100	47.21
			TSA	4250	4250	4250	0.00000	100	43.65
			ACO	4250	4250	4250	0.00000	100	35.50
			MTS	**4250**	**4250**	**4250**	**0.00000**	**100**	**12.11**
0.990	3638	3203312303	GA	3900	3760	3638	88.05423	13	57.12
			TSA	3886	3700	3638	68.02699	17	55.53
			ACO	3686	3654	3638	21.00489	50	44.89
			MTS	**3638**	**3638**	**3638**	**0.00000**	**100**	**10.24**
0.950	2184	3003300303	GA	2692	2481	2184	121.65421	56	57.35
			TSA	2622	2305	2184	98.014345	80	31.52
			ACO	2444	2193	2184	47.469288	97	22.11
			MTS	**2184**	**2184**	**2184**	**0.00000**	**100**	**6.53**
0.900	1904	3003200203	GA	2402	1982	1904	126.28258	43	47.55
			TSA	2124	1937	1904	87.234497	93	17.51
			ACO	1904	1904	1904	0.00000	100	14.27
			MTS	**1904**	**1904**	**1904**	**0.00000**	**100**	**2.36**
0.850	1724	2003300102	GA	1764	1734	1724	10.061878	90	21.12
			TSA	1736	1726	1724	4.290594	100	1536.
			ACO	1732	1727	1724	3.835706	100	8.11
			MTS	**1724**	**1724**	**1724**	**0.00000**	**100**	**1.10**

Table 4 Summary results of five nodes problem (performed 30 trails)

Reliability	Optimal Cost	Configuration	Algorithm	Max. Cost	Average Cost	Min. Cost	Standard Deviation	% Get Optimal	CPU Time
0.999	6008	033300000001013330	GA	6008	6008	6008	0.00000	100	63.41
			TSA	6008	6008	6008	0.00000	100	55.48
			ACO	6008	6008	6008	0.00000	100	26.42
			MTS	**6008**	**6008**	**6008**	**0.00000**	**100**	**15.23**
0.995	4464	022200000000003330	GA	5382	4795	4464	232.843107	13	112.42
			TSA	4970	4681	4464	200.417828	36	105.20
			ACO	4596	4506	4464	56.766673	63	50.87
			MTS	**4464**	**4464**	**4464**	**0.00000**	**100**	**10.24**
0.990	4074	013100000000001330	GA	5212	4190	4074	265.212310	76	72.66
			TSA	4952	4189	4074	251.683618	80	69.75
			ACO	4086	4078	4074	5.881590	100	42.27
			MTS	**4074**	**4074**	**4074**	**0.00000**	**100**	**18.42**
0.950	2826	002100000000000330	GA	3446	2980	2826	164.221423	30	106.43
			TSA	3112	2922	2826	147.749415	53	93.21
			ACO	3248	2900	2826	130.371018	70	72.68
			MTS	**2826**	**2826**	**2826**	**0.00000**	**100**	**22.56**
0.900	2328	011000000000001300	GA	2648	2386	2328	75.218902	57	90.23
			TSA	2530	2374	2328	47.981001	60	79.92
			ACO	2394	2337	2328	22.816509	87	49.98
			MTS	**2328**	**2328**	**2328**	**0.00000**	**100**	**21.23**
0.850	1990	002000001000000220	GA	2312	2074	1990	97.273386	50	65.98
			TSA	2335	2104	1990	105.07909	48	63.02
			ACO	2180	2008	1990	70.477535	93	61.39
			MTS	**1990**	**1990**	**1990**	**0.00000**	**100**	**25.14**

Table 5 Summary results of source-sink nodes problem (performed 30 trails)

Reliability	Algorithm	Max. Cost (฿)	Average Cost (฿)	Min.Cost (฿)	Standard Deviation
0.999	GA	10,364,282	10,361,136	10,359,058	1522.41
	TSA	10,361,728	10,359,303	10,356,430	1305.12
	ACO	10,358,095	10,357,185	10,356,121	567.75
	MTS	**10,343,501**	**10,331,632**	**10,324,514**	**315.54**
0.995	GA	8,274,131	8,271,620	8,269,961	1215.39
	TSA	8,272,093	8,270,157	8,267,863	1041.90
	ACO	8,269,192	8,268,466	8,267,616	453.25
	MTS	**8,257,863**	**8,251,491**	**8,245,361**	**251.62**
0.990	GA	7,022,953	7,020,822	7,019,414	1031.60
	TSA	7,021,223	7,019,579	7,017,633	884.35
	ACO	7,018,761	7,018,144	7,017,423	384.71
	MTS	**6,899,623**	**6,898,631**	**6,894,144**	**160.55**
0.950	GA	4,975,827	4,974,316	4,973,319	730.90
	TSA	4,974,601	4,973,436	4,972,057	626.56
	ACO	4,972,856	4,972,419	4,971,908	272.57
	MTS	**4,954,615**	**4,950,241**	**4,942,416**	**121.19**
0.900	GA	4,443,038	4,441,690	4,440,799	652.54
	TSA	4,441,944	4,440,904	4,439,672	559.47
	ACO	4,440,586	4,439,996	4,439,540	253.39
	MTS	**4,425,619**	**4,418,629**	**4,412,612**	**110.82**
0.850	GA	4,021,989	4,020,768	4,019,962	622.54
	TSA	4,020,998	4,020,056	4,018,942	590.79
	ACO	4,019,988	4,019,234	4,018,822	280.32
	MTS	**3,892,512**	**3,889,415**	**3,867,642**	**135.46**

Note: 1 $US = 35 ฿

Table 6. Summary results of 19 districts in Bangkok, Thailand (performed 30 trails)
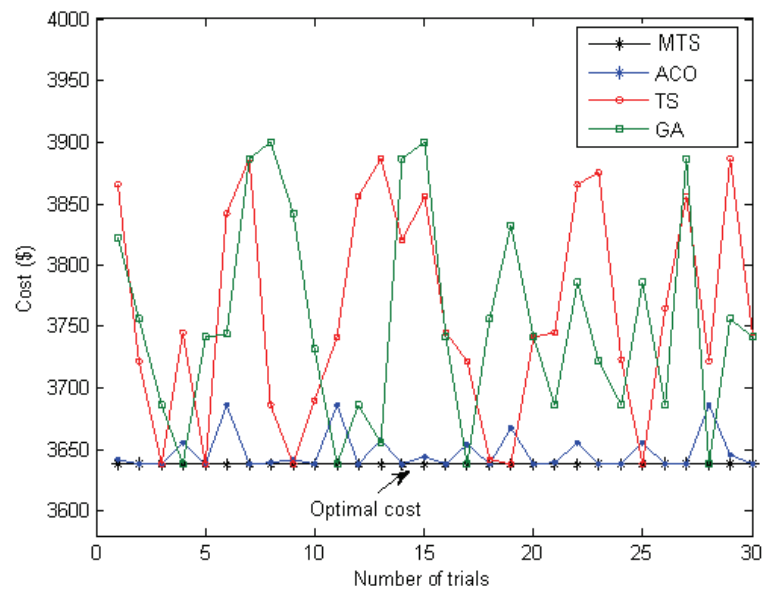
Fig. 6 Distribution of cost for test problem 1 ($R_0 = 0.990$)
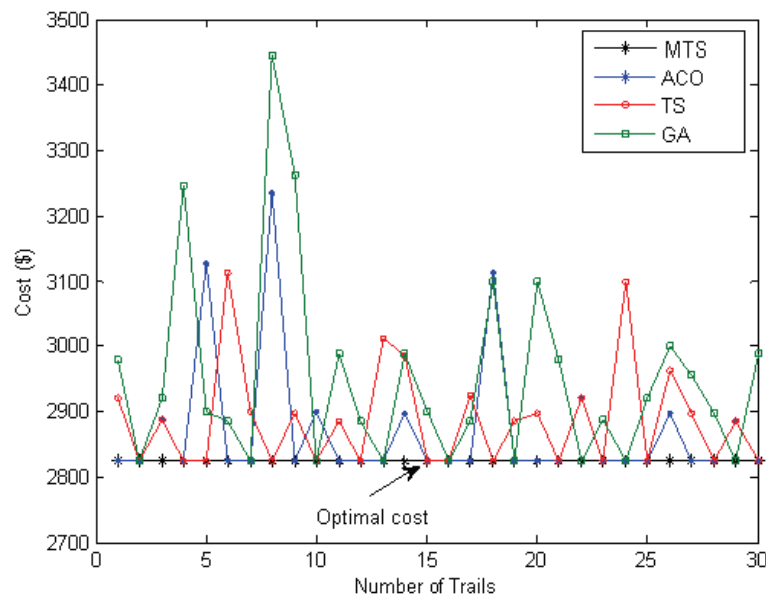
Fig. 7. Distribution of cost for test problem 2 ($R_0 = 0.950$)

5. Conclusion and discussion

In this chapter, the MTS method has been applied to solve the topology network design problem with considering both economics and reliability. The MTS algorithm shows superior features such as high-quality solution, stable convergence characteristic, and good computation efficiency. The studied results confirm that the proposed MTS are indeed capable of obtaining higher quality solution efficiently, convergence characteristic and computation efficiency in comparison with GA, TS and ASO methods.

6. References

Aggarwal, K. K.; Chopra, Y. C. & Bajwa, J. S. (1982). Reliability evaluation by network decomposition : *IEEE Transactions on Reliability*, Vol. R-31, pp. 355-358.

Ball, M. & Van Slyke, R. M. (1977). Backtracking algorithms for network reliability analysis: *Annals of Discrete Mathematics*, Vol. 1, pp. 49-64.

Bell, J. & McMullen, P. (2004). Ant colony optimization techniques for the vehicle routing problem: *Advanced Engineering Informatics*, Vol. 18, pp. 41-48.

Beltran, H. F. & Skorin-Kapov, D. (1994). On minimum cost isolated failure immune networks: *Telecomm. Sys*, Vol. 3, pp. 183-200.

Bland, J. A. & Dawson, G. P. (1991). Tabu Search and Design Optimization: *Computer - aided design*, Vol. 23, No. 3, pp. 195-201.

Chelouah, R. & Siarray, P. (2000). Tabu Search applied to global optimization: *European Journal of Operational Research,*. Vol. 123, pp. 256-270.

Coloni, A; Dorigo, M. & Maniezzo, V. (1991). Distributed optimization by ant colonies: *Processding of Eur. Conf. Artificial Life*, pp. 134-142.

Darren, L. D. & Smith, A. E. (1998). Economic Design of Reliable Networks: *IEEE Transactions on Economic of Reliability Engineering*, July.

Dorigo, M.; Maniezzo, V. & Colorni, A. (1996). Ant system: Optimization by a colony of cooperative agents: *IEEE Trans. Syst., Man, Cybern. B, Cybern.,* Vol. 26, No. 1, pp. 29-41.

Dorigo, M. & and Gambardella, L. M. (1997). Ant colonies for the traveling saleman problem: *Biosystems*, Vol. 43, pp. 73-81.

Dorigo, M. & Gambardella, L. M. (1997). Ant colony system: a cooperative learning approach to the traveling saleman problem: *IEEE Trans. Evolut. Comput.*, Vol. 1, No. 1, pp. 53-66.

Fanni, A. et al., (1999). Tabu Search metaheuristics for electromagnetic problems optimization in continuous domain: IEEE Trans. On Magnet. Vol. 34, no. 3, pp. 1694-1697.

Fishman, G. S. (1986). A comparison of four Monte Carlo method for estimating the probability of s-t connectedness: *IEEE Transactions on Reliability,* Vol. 35, No. 2, pp. 145-155.

Gambardella, L. M.; Taillard, E. & Dorigo, M. (1999). Ant colonies for the quadratic assignment problem*: J. Oper. Res. Soc.,* Vol. 50, pp. 167-176.

Glover, F. (1989). Tabu Search Part I: *ORSA J. Comput*, Vol. 1 No. 3, pp. 190-206.

Glover, F. (1990). Tabu Search Part II: *ORSA J. Comput*, Vol. 2 No. 1, pp. 4-32.

Glover, F.; Lee, M. & Ryan, J. (1991). Least-cost network topology design for a new service: An application of a tabu search: *Ann. Oper. Res.,* Vol. 33, pp. 351-362.

Ho, S. L.; Yang, S.; Wong, H. C.; Cheng, K.W.E. & Ni, G. (2005). An Improved Ant Colony Optimization Algorithm and Its Application to Electromagnetic Devices Designs: *IEEE Transactions on Magnetics,* Vol. 41, No. 5, pp. 1764-1767.

Jan, R. H (1993). Design of reliable networks: *Computers and Operations Research*, Vol. 20, No. 1, pp. 25-34.

Jan, R. H.; Hwang, F. J. & Chen, S. T. (1993). Topology optimization of a communication network subject to a reliability constraint: *IEEE Transactions on Reliability*, Vol. 42, No. 1, pp. 63-70.

Kumar, A.; Pathak, R. M. ; Gupta Y. P. & Parsaei, H. R. (1995). A genetic algorithm for distributed system topology design: *Comp. Ind. Eng.,* Vol. 28, pp. 659-670.

Mathur, M.; Karale, S. B.; Priyee, S.; Jayaraman, V. K. & Kulkarni, B. D. (2000). Ant colony approach to continuous function optimization: *Ind. Eng. Chem. Res.,* Vol. 39, pp. 3814-3822.

Pierre, S; Hyppolite, M. A. ; Bourjolly, J.M. & Dioume, O. (1995). Topology design of computer communication network using simulated annealing: *Eng. Applicat. Artificial Intell.*, Vol. 8, pp. 61-69.

Pothiya, S.; Ngamroo, I.; Runggeratigul, S. & Prinya Tantaswadi, P. (2006). Design of Optimal Fuzzy Logic based PI Controller using Multiple Tabu Search Algorithm for Load Frequency Control: *International Journal of Control, Automation, and Systems*, Vol. 4, No. 2, pp. 155-164.

Pothiya, S.; Ngamroo, I. & Kongprawechnon, W. (2008). Application of multiple tabu search algorithm to solve dynamic economic dispatch considering generator constraints: *Energy Conversion and Management* , Vol. 49, pp. 506–516.

Ramirez-Marquez, J. & David Coit, D. (2005). A Monte Carlo Simulation Approach for Approximating Multistate Two-Terminal Reliability: *Reliability Engineering & System Safety,* Vol. 87, No. 2.

Shinmori, S.; Koide, T. & Ishii, H. (1995). On lower bound for network reliability by edge-packing: *Transactions of the Japan Society for Industrial and Applied Mathematics,* Vol. 5, No. 2, pp. 139-151.
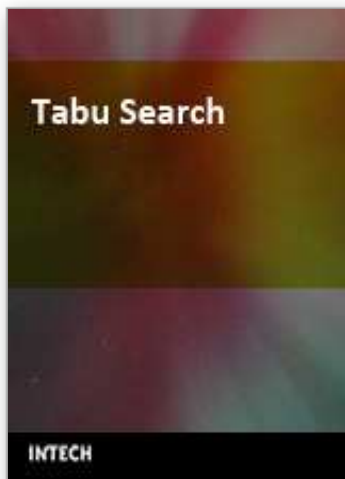
Tabu Search

Edited by Wassim Jaziri

The goal of this book is to report original researches on algorithms and applications of Tabu Search to real-world problems as well as recent improvements and extensions on its concepts and algorithms. The book' Chapters identify useful new implementations and ways to integrate and apply the principles of Tabu Search, to hybrid it with others optimization methods, to prove new theoretical results, and to describe the successful application of optimization methods to real world problems. Chapters were selected after a careful review process by reviewers, based on the originality, relevance and their contribution to local search techniques and more precisely to Tabu Search.

INTECH
open science | open minds