A Novel Genetic Algorithm for Degree-Constrained Minimum Spanning Tree Problem

***Lixia Hanr*† *and Yuping Wang*††,**

†School of Science, Xidian University, Xi'an 710071, China
†† School of Computer Science and Technology, Xidian University, Xi'an 710071, China

Summary
A novel genetic algorithm is proposed for degree-constrained Minimum Spanning Tree (short for *d*-MST) problem in this paper. First, a novel model that transfer *d*-MST problem into a preference two objective MST problem is presented. Based on this model, a new crossover operator, a local search scheme, a mutation operator and a new selection operator are designed based on the preference of the two objectives. Then, a new genetic algorithm (short for GA) is proposed. Furthermore, the convergence of the proposed algorithm to globally optimal solution with probability one is proved. The simulation results indicate the proposed algorithm is effective.

Key words:
degree-constrained minimum spanning tree, genetic algorithm, convergence.

Introduction

The minimum spanning tree (MST) of a graph is an important concept in the communication network design and other network-related problem. Given a graph with cost (or weight) associated with each edge, the MST problem is to find a spanning tree of the graph with minimal total cost. When the graph's edge costs are fixed and the search is unconstrained, the well-known algorithm of Krushal [1] and Prim [2] can identify MST in times that are polynomial in the number of nodes. However, the MST usually satisfies some additional constraints, for example, a bound on the degree, or a bound on the diameter of the tree, which often makes the problem NP-hard. Thirteen such NP-complete variants were listed by Gary and Johnson [3]. Camerini, Galbiati, and Maffioli [4~6] catalogued eighteen spanning tree parameters that one might wish to optimize.

In fact, the *d*-MST problem can be regarded as a generalization of the Traveling Salesman Problem when at most only two edges are allowed incident to each node in MST. Because of its complexity, we can apply exact optimization algorithms only to small instances of them. For larger instances, we turn to heuristic techniques, including genetic algorithms (GAs).

Recently, some heuristic algorithms for the *d*-MST have been proposed. Boldon et al [7] present dual simplex approach which performs very well on some benchmarks based on Prim's algorithm. Zhou and Gen [8] described a simple evolutionary algorithm using a Prüfer based encoding and applied it to some small *d*-MST problem In [9], Knowles and Corne introduced a novel tree construction algorithm called the Randomized Primal Method (RPM) with employed a modified version of Prim's algorithm call d-Prim. This method built degree-constrained trees of low cost from solution vectors. Meanwhile, Raidl and Julstrom [10] presented a novel coding of spanning tree in a genetic algorithm. In the coding, chromosomes are strings of numerial weights associated with the target graph's nodes. The weights temporarily bias the graph's edge costs, and an extension of Prim's algorithm, applied to the biased costs, identifies the feasible spanning tree a corresponding to a chromosome. This generally outperformed RPM. The one of the best techniques so far according to the best of our knovledge, however, seems to be Raidl's more recent work [11] which used a direct spanning tree encoding, with associated specialized mutation and recombination operators .

All algorithms above treated the *d*-MST problem as single-objective problem. In this paper, we presented a bi-objective genetic algorithm for *d*-MST problem with the violation degree as the second objective. This bi-objective MST problem with the violation degree as the second objective. This bi-objective MST problem is a preference one. Based on this model, we design a new crossover operation, a local search scheme, a mutation operator and a new selection operator based on the preference of the two objectives. The convergence of the proposed algorithm to globally optimal solution with probability one is proved. The simulation results indicate the proposed algorithm is effective.

Manuscript received July 5, 2006.
Manuscript revised July 25,, 2006.

2. Transformation of *d*-MST Problem

Given a connected, undirected graph G with n nodes, a spanning tree T is a subgraph of a G that connects all of G's nodes and contains no cycles. When every edge (i, j) is associated with a numerical costs c_{ij}, a minimum spanning tree (MST) is a spanning tree of the smallest possible total edge cost $C = \sum_{(i,j)\in T} c_{ij}$.

The degree (denoted by d_i) of a node $i(i = 1, \cdots, n)$ is the number of incident edges, and the degree of a graph is the maximum degree of its nodes. The degree-constrained MST problem is to determine a spanning tree (ST) of the minimum total edge cost and degree no more than a given value *d*: *d*-MST. The mathematic model of *d*-MST problem is as follows.

$$\begin{cases} \min \sum_{(i,j)\in T, T\in S} c_{ij} \\ s.t. \quad d_i \le d, i = 1, \cdots n \end{cases} \quad (1)$$

where $T \in S$ (Let S be the spanning tree space). If a spanning tree satisfies all constraints of problem (1), then it is called a feasible solution. The set $FS = \{T | d_i(T) \le d, i = 1 \sim n, T \in S\}$ is called the set of feasible solution of problem (1) (simply the feasible set).

Define function $f_1(T) = f(T)$ and

$f_2(T) = \max\{0, d_i(T) - d\}, i = 1 \sim n$.

Obviously, for any $T \in S$, we have $f_2(T) \ge 0$, and $f_2(T) = 0$ if and only if $T \in FS$. Therefore, *d*-MST problem (1) can be transformed into the following two objective MST problem

$$\min\{f_1(T), f_2(T)\} \quad (2)$$

Optimizing $f_1(T)$ means searching for an MST the lowest total edge cost, and optimizing $f_2(T)$ means searching for the feasible solution for *d*-MST problem (1). Thus simultaneously optimizing both $f_1(T)$ and $f_2(T)$ means looking for not only a feasible solution for problem (1), but also a solution minimizing the object function of problem (1).

It is obvious that the transformation from problem (1) into (2) converts single-objective constrained MST problem with n constraints (n is the number of nodes) into a two-objective unconstrained MST problem. And the search space of problem (2) is larger than that of problem(1).

In order to illustrate the relationship of problem (1) and (2), we introduce the concept of Pareto optimal solution as follows.

Definition 1. For $T^* \in S$, if there exists no ST $T \in S$ such that $f_i(T) \le f_i(T^*)$ for $i = 1,2$, and $f_j(T) < f_j(T^*)$, for some $j = 1$ or 2, then T^* is called a Pareto optimal solution of problem (2).

Lemma 1. Suppose that there exists at least one optimal solution of problem (1), then T^* is an optimal solution of problem (1) if and only if it is a Pareto optimal solution of problem (2), and $T^* \in FS$

Proof. If T^* is an optima solution of problem (1), then $T^* \in FS$. Thus, $f_2(T)$ attains its minimum value at T^*, and $f_1(T)$ attains the minimum value at T^* in FS. Therefore, T^* is a Pareto optimal solution of problem (2).

If $T^* \in FS$ is a Pareto optimal solution of problem (2), suppose that T^* is not an optimal solution of problem (1), then there exist some $T \in FS$ such that $f_1(T) < f_1(T^*)$. Note that $f_1(T) = f_2(T^*) = 0$. This contradicts the fact that T^* is a Pareto optimal solution of problem (2). This proof is completed.

Note that the second objective in problem (2) is more important than the first one for infeasible individuals because it is crucial to improve these individuals into feasible ones. On the other hand, the first objective is more important than the second one for feasible individuals because in this case it is the key issue to get better and better individual. Due to this characteristic of problem (2), the difference evolution schemes are adopted. For feasible solution, the key issue in evolution is to optimize the first objective function so that we can get the optimal solution for problem (1) as soon as possible. In this case, we will use some evolution scheme to improve the first objective function $f_1(T)$. While for infeasible solution, the key issue is to optimize the second objective function so that we can get feasible solution for problem (1) as soon as possible. Based on this idea, we design a new crossover operator that evolves the individual toward feasible solution. Also, some evolution scheme to decrease the second objective function $f_2(T)$ is presented.

3 The Proposed Genetic Algorithm

In this section, we focus on the design of GA for the *d*-MST problem. Among the several tree encodings([10~13]), only the Prüfernumber encoding explicitly contain the information of node degree that any node with degree d will apper exactly $(d-1)$ times in the encoding. Thus, we adopt Prüfer number encoding to represent the solution.

3.1 Prüfer Number Encoding

One of the classical theorems in graphical enumeration is Cayley's theorem that there are n^{n-2} distinct spanning tree on a complete graph with n nodes [14],[15]. Prüfer presented a constructive proof of Cayley's theorem by establishing an one-to-one mapping between such trees and vectors of length $(n-2)$ in which each element of vectors is a positive integer no greater than n. These vectors are called Prüfer number, and we can use only $(n-2)$ integers permutation to uniquely represent a tree with n nodes where each integer is an digit between 1 and n inclusive.

Encoding Procedure

Step 1) Let i be the smallest leaf node and node j be incident to node i. Set j be the first digit in the encoding. The encoding is built by appending digits to the right.

Step 2) Removenode i and the edge from i to j.

Step 3) Repeat above operation until only one edge is left.

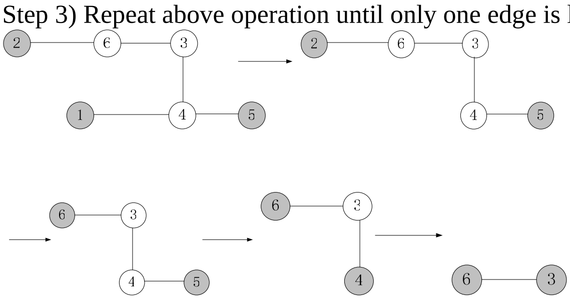


Fig.1 The process of the encoding procedure.

Fig. 1 is an example to illustrate this encoding procedure. The spanning tree T in Fig. 1 corresponds to Prüfer vector [4 6 4 3] on a complete graph with 6-node represent in Fig. 1. Node 1 is the smallest leaf node, and node 4 is incident to node 1. Therefore, 4 is the first digit in the Prüfernumber. Remove node 1 and the edge (1,4) from T. Repeat the process on the subtree until edge (3,6) is left. The Prüfer number of this tree is produced.

In a Prüfer number encoding, a tree is encoded as a Prüfer vector P and a set of its eligible nodes $\overline{P}$ (the set of all nodes not included in P). For a give P and $\overline{P}$, the corresponding tree represented by P and $\overline{P}$, denoted as T, can be obtained by the following procedure.

Decoding Procedure:

Step 1) Let node i be the smallest eligible node of $\overline{P}$ and node j be the leftmost element of P. If $i \neq j$, add the edge (i, j) into the tree T. If i is no longer eligible, then remove node i from $\overline{P}$. Delete j from P. If j does not occur anywhere in the remaining part of P, then put it into $\overline{P}$. Repeat the process until P is empty.

Step 2) For the remaining last two nodes u and v of $\overline{P}$, add the edge (u,v) into the tree T.

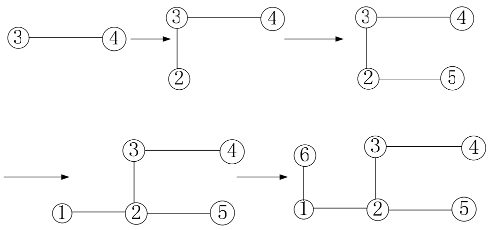


Fig. 2 The process of the decoding procedure

Fig 2 illustrate the decoding procedure. Prüfervector [3 2 2 1] corresponds to a spanning tree on a complete graph with 6-node represent in Fig. 2.The construction of the spanning tree is described as follows.

For the Prüfer number with $P=[3\ 2\ 2\ 1]$, the set of its eligible nodes $\overline{P}=[4\ 5\ 6]$. It is obvious node 4 is the smallest eligible node and node 3 is the leftmost element of P. Add edge (3,4) to the tree, and remove 4 from $\overline{P}$, 3 from P respectively. Because node 3 is no longer present in the remaining P, put it into $\overline{P}$. Thus $P=[2\ 2\ 1], \overline{P}=[3\ 5\ 6]$. Repeat the above process until P is empty and add the edge (1,6) into the tree T for the remaining last two node 1 and 6 of $\overline{P}$.

The encoding is deceptively appealing. Prüfernumbers can be encoded and decoded in times that are $O(n \log n)$.

3.2 Selection Operator with Preference

Due to the special structure of problem (2), we design a new selection operation which is different from that for usual multiobjective optimization problem. The detail is as follows.

- If the second objective values of two solutions are both zero, we prefer to select the one with the smaller first objective value. This selection is in accordance with the concept of Pareto dominance.
- If the second objective value of one solution is zero, and that of the other is nonzero, we prefer to choose the one with the zero second objective value.
- If the second objective values of two solutions are both nonzero, we prefer to select the one with the smaller second objective value.

In fact, the selection scheme ensures the best feasible solution for problem (1) to inherit to the next generation. Meanwhile, it prefers to choose the feasible solutions for problem (1) and the infeasible solutions with smaller second objective function.

3.3 Crossover Operator

In fact, the existence of infeasible solutions for problem (1) improves the opportunity of finding the optimal solution for the *d*-MST problem. The ST T in Fig.3 is an infeasible solution for problem (1), while it is a feasible solution for problem (2) with degree constraint 3 in a 6-node complete graph. Let ST $T^{'}$ be the MST for problem (1). It is obvious that ST T differs from ST $T^{'}$ only one edge. Based on this fact, we design new crossover operator to evolve toward this direction.

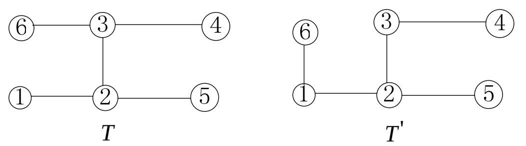


Fig. 3 Two spanning trees

Because of the preference of the feasible solutions for problem (1), a new crossover method called partially multi-point crossover is modified to induce the solution to decrease the second object $f_2(x)$.

It is now appropriate to present the detailed procedure of the crossover method.

Let $a=(a_1a_2\cdots a_{n-2})$, $b=(b_1b_2\cdots b_{n-2})$ be the parents be a pair of parents for crossover.

Step 1) Choose two positions at random on these strings, say, i and $j(i<j)$;

Step 2) Interchange the partially string $(a_{i+1}\cdots a_j)$ with $(b_{i+1}\cdots b_j)$ in a and b to get two new individuals $a^{'}=(a_1^{'}\cdots a_{n-2}^{'})$ and $b^{'}=(b_1^{'}\cdots b_{n-2}^{'})$.

Step 3) Set $h=(i+1)$ and repeat the following procedures;

- If the number of times $a_h^{'}$ appears in $a^{'}$ exceeds $(d-1)$, then select an integer $c\in\{1,\cdots,n\}$ randomly which appears less than $(d-1)$ times in $a^{'}$ and replace $a_h^{'}$ by c. set $h=h+1$;
- If $h>j$, stop and let $a^{'}$ be the offspring of a.

Step 3 is carried out for $b^{'}$ in the same manner, then the offspring of b is obtained. Fig. 4 illustrates this crossover operator with degree constraint 3 in a 6-node complete graph.

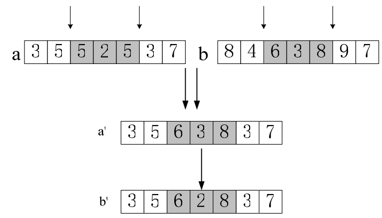


Fig. 4 Illustration of crossover operation

For this crossover operator, we have the following conclusions:

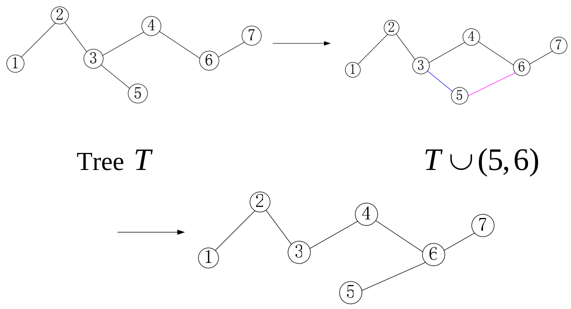
1) If the parents to participate crossover are both feasible solution for problem (1), the offspring generated by crossover are both feasible solution for problem (1).
2) If one parent is feasible solution for problem (1) and the other is infeasible solution for problem (1), at least one offspring is feasible solution for problem (1).
3) If the parents to participate crossover are both infeasible solution for problem (1), the offspring may be feasible solution for problem (1).

In a word, the crossover operator improves the number of feasible solutions in genetic algorithm.

3.4 Local Search Scheme

Local search algorithm is an important approach for the most successful meta-heuristics to solve a wide variety of single objective combinatorial problems. It can be easily revised to be applicable to multi-criteria problems. A new local search scheme is designed to improve the offspring generated by crossover operator. It is not an exact local optimization algorithm; instead it uses relatively small number of individuals in each search process. Thus it usually cannot generate the local optimal solution. However, it can generate the solution good enough using much less computation than general exactly local optimization search algorithm, and simultaneously improve multiple objectives. The detail is as follows.

Let S be any offspring generated by crossover operator. The offspring generated by local search scheme can be easily generated by the following pseudo code.



Delete edge (3,5)

Fig.5 Illustration of local search scheme

For $i = 1$ *to* $(n-2)$ *do*

Substitute a_i *with a integer* c *generated randomly, denoted the new individual as* $a^{'}$

Choose the Pareto one as a *from* $\{a, a^{'}\}$ *according the new selection operator.*

Enddo

Select the last $a^{'}$ *as the result of local search from* S .

3.5 Discrete Multi-uniform Mutation Operator

Mutation operator plays the important role of local random search in evolutionary algorithm. For an individual $a = (a_1 \cdots a_{n-2})$ to undergo mutation, a mutation scheme called discrete multi-uniform mutation operator is designed as follows.

For each component a_i of a , it is changed to a randomly generated integer $c_i \in \{1, \cdots, n\}$, $i = 1, 2, \cdots, n$ .The offspring of a is $b = (b_1 b_2 \cdots b_{n-2})$. It is illustrated in Fig. 6.

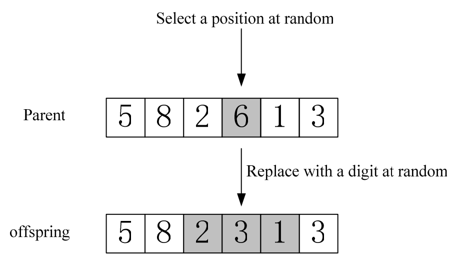


Fig. 6. Illustration of mutation operation.

4. The Proposed Algorithm

Novel Genetic Algorithm (NGA):

Step 1. (Initialization) Choose population size N , proper crossover probability p_c and mutation probability p_m , respectively. Generate initial population $P(0)$. Let the generation number $t = 0$.

Step 2. (Crossover) Choose the parents for crossover from $P(t)$ with probability p_c . If the number of parents chosen is odd, then randomly choose additional one from $P(t)$. Afterwards, randomly match every two parents as a pair and use the proposed crossover operator to each pair to generate two offspring. All these offspring constitute a set denoted by O .

Step 3. (Local Search) For each offspring generated by crossover, the proposed local search scheme is used to it to generate an improved offspring. All these improved offspring constitute a set denote by O_1 .

Step 4. (Mutation) Selection the parents for mutation from set O_1 with probability p_m . For each chosen parent, the proposed mutation operator is used to it to generate a new offspring. These new offspring constitute a set denoted by O_2 .

Step 5. (Selection) Select the best N individuals among the set $P(t) \cup O_1 \cup O_2$ as the next generation population $P(t+1)$, let $t = t + 1$.

Step 6. (Termination) If termination conditions hold, then stop, and keep the best solution obtained as the approximate global optimal solution of the problem; otherwise, go to step 2.

5. Global Convergence

To analyze the properties of genetic algorithms easily, we use a brief and general framework to describe genetic algorithm as follows: in each iteration the population is modified by a number of successive probabilistic transformation. Evidently, the resulting new population only depends on the state of the current population in a probabilistic manner. This property, known as the Markov property, reveals that Markov processes are appropriate models for the probabilistic behavior of genetic algorithms. Notice that the deterministic concept of “the convergence to the optimum” is not appropriate because the state transitions of a genetic algorithm are of stochastic nature. In order to clarify the exact semantic of a phrase like “the convergence to the global optimum” one has at first to define the exact stochastic convergence.

Let Ω is the Prüfer number encoding search space for the degree-constrained MST problem. In order to prove the global convergence of the algorithm with probability one, it is required to introduce the following concept:

Definition 1. For two chromosomes a and b if $\Pr ob\{MC(a)=b\}>0$, then chromosome b is called to be reachable from chromosome a by crossover and mutation, where $MC(a)$ represents the offspring that was generated from a by crossover operator and mutation operator..

For the proposed algorithm NGA, we have the following conclusion:

Theorem 2 For any two chromosomes , b is reachable from a by crossover and mutation.

Proof. In fact, note that the probability of choosing a to take part in crossover is $p_c > 0$.

Suppose that c is any offspring generated from a by crossover operator and e is the individual generated from c by local search, then the probability of e being chosen to take part in mutation is $p_m > 0$. Thus the probability of b being generated form a by crossover and mutation satisfies

$$\Pr ob\{MC(c)=b\} \geq p_c \cdot p_m \cdot \Pr ob\{M(e)=b\}$$

It only needs to prove that b is reachable from e by mutation, i.e., to prove

$$\Pr ob\{M(e)=b\}>0$$

where $M(e)$ represents the offspring of e by mutation. Suppose that e and b have the following form

$e=(e_1,e_2,\cdots,e_{n-2})$ $b=(b_1,b_2,\cdots,b_{n-2})$

It can be known from the mutation operator that the probability of generating b_i from e_i by mutation is $\frac{1}{n}$.

Therefore,

$$\Pr ob\{M(e)=b\}=\frac{1}{n}\cdot\frac{1}{n}\cdots\frac{1}{n}=\frac{1}{n^{n-2}}>0$$

Thus,

$$\Pr ob\{MC(c)=b\} \geq p_c \cdot p_m \cdot \Pr ob\{M(e)=b\}=\frac{p_c \cdot p_m}{n^{n-2}}>0$$

This proves that b is reachable from a by crossover and mutation.

Theorem3 Thepopulationsequen $P(0),P(1),\cdots,P(t),\cdots$ is Pareto monotone, i.e., $P(t+1)$ is better or at least no worse than $P(t)$ for any t.

It is obvious according to the new selection operator that the population sequence is Pareto monotone.

Theorem 4 The proposed genetic algorithm converges to the global optimal solution with probability one.

6. The numerical experiments

In order to examin the proposed algorithm, we compute two kinds of problems respectively.

he first numerical problem is selected from [12] and the optimal solution is 2256. The numerical example is made by Zhou and Gen in [4]. The problem is a 9-node complete graph and the degree constraint is 3 for all nodes. Edge weights of this problem is given in Table 1.

Zhou and Gen [4] reported that their GA found the optimal solution on 66.7% (Fig. 7) times of total runs in 25000 function evaluations (a population of 50 for 500 generations). The proposed algorithm found the optimal solution on this problem on greater than 83.3% times of total runs only in 10000 function evaluations. This demonstrate that the proposed algorithm is more effective than that in [4].

The second kind of problems is randomly generated. Random graphs are those in which the weight of each edge has been generated randomly from a uniform distribution within some predefined range. Boldom et. Al. [7] have found that when reasonably large graphs are generated the maximum degree of a node in the underlying MST rarely exceed four. Hence, they employed a means of generating biased random complete graphs in which a high maximum-node degree is present in the underlying MST. For details of these problems readers can refer to [9].

Results are given in terms of the ratio of the best *d*-MST cost found to the known cost of the MST of the graph. The latter is found by simply running Prim's algorithm, which guarantees finding a minimum cost spanning tree without degree constraints. We compared the proposed algorthm with BF2 [7], GA1 [8] on an M-graph of 250 nodes at various degree constraints. Table 2 represents the best, mean and worst solutions found by each of the algorithms in terms of the *d*-MST/MST ration.

It is clear from the above results that the proposed algorithm finds the lower-cost solutions than any of the other alogorithms. This result is presented graphically in Fig. 9, which shows the mean (over all nine graphs) of the best solution found by each algorithm.

7 Conclusion

In this paper, we presented a new genetic algorithm for the degree-constrained minimum spanning tree problem. First, we transform the *d*-MST problem into bi-objective preference MST problem. Based on this model, we design a new crossover operation, a local search scheme, a mutation operator and a new selection operator based on the preference of the two objectives. The convergence of the proposed algorithm to globally optimal solution with probability one is proved. The simulation results indicate the proposed algorithm is effective.

Acknowledgments

This work was jointly support by National Natural Science Foundation of China (No. 60374063) and the Natural Science Foundation of Shanaxi Province of China (No.2001SL06).

References

[1] J. B. Kruskal, On the Shortes Spanning Tree of a Graph and the Traveling Salesman Problem.. Amer. Math. Soc., vol. 7, pp. 48-50, 1956.
[2] R. Prim, Shortest Connection Networks and Some Generalization. Bell Syst. Tech. J., vol. 36, pp. 1389-1401, 1957.
[3] M. R. Garey and D. S. Johnson, Computers and Intractability: A Guide to the theory of NP-completeness, W. H. Free3man, San Francisco, 1979.
[4] P. M. Comerini, G. Galbiati and F. Maffioli, Complexity of spanning tree problem: Part Ⅰ, Eur. J. Oper. Res., 5, pp. 346-352,1980.
[5] P. M. Comerini, G. Galbiati and F. Maffioli, On the complexity of finding multi-constrained spanning trees, Disc. Appl. Math., 5, pp. 39-50, 1983.
[6] P. M. Comerini, G. Galbiati and F. Maffioli, The complexity of weighted constrained spanning trees, In: Proc. Colloq. On the theory of Algorithms, 1984.
[7] B. Boldom, N. Deo, and N. Kumar, Minimum-Weight Degree-Constrained Spanning Tree Problem: Heuristics and Implementation on an SIMD Parallel Machine, Technical Report CS-TR-95-02, Department of Computer Science, University of Central Florida, Orlando, FL 32816, 1995.
[8] G. Zhou and M. Gen, A Note on Genetic Algorithms for Degree-Constrained Spanning Tree Problems, Networks, 30, pp. 91-95, 1997.
[9] J. Knowles and D. Corne. A New Evolutionary Approach to the Degree-Constrained Minimum Spanning Tree Problem. IEEE Transactions on Evolutionary Computation, 4(2),pp 125-134, 2000.
[10] G. R. Raidl and B. A.Julstrom. A Weighted Coding in a Genetic Algorithm for the Degree-Constrained Minimum Spanning Tree Problem. In Proceeding of the 2000 ACM Symposium on Applied Computing, ACM Press, 2000.
[11] G. R. Raidl, An Efficient Evolutionary Algorithm for the Degree-Constrained Minimum Spanning Tree Problem, In Proceeding of the 2000 Congress on Evolutionary Computation (CECOO), pp. 104-111, 2000.
[12] Günther R. Raidl and Bryant A. Julstrom. Edge Sets: An Effective Evolutionary Coding of Spanning Tree. IEEE Transactions on Evolutionary Computation, vol. 7, no. 3, pp.225-239, 2003.
[13] PrüferH., Neuer Beweis eines satzes uber permutation, Arch. Math. Phys., vol. 27, pp. 742-744, 1918.
[14] A. Cayley, A Theorem on Trees. Quarterly Journal of Mathematics, 23, pp. 376-378, 1889.
[15] P. Piggott and F. Suraweera. Encoding graphs for genetic algorithms: An investigation using the minimum spanning tree problem. In X. Yao, ed., Progress in Evolutionary Computation, LNAI 956, pp. 305-314, Springer, Berlin, 1995.
[16] David A. Van Veldhuizen, Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations. Ph.D. Air Force Institute of Technology 1999.
[17] Savelsbergh M. and T. Volgenant, Edge exchanges in the degree-constrained spanning tree problem, Comput. & Ops. Res, vol. 12, no. 4, pp. 341-348, 1985.

Table 1. Edge weights of the 9-node *d*-MST problem

i	2	3	4	5	6	7	8	9
1	224	224	361	671	300	539	800	943
2	-	200	200	447	283	400	728	762
3		-	400	566	447	600	922	949
4			-	400	200	200	539	583
5				-	600	447	781	510
6					-	283	500	707
7						-	361	424
8							-	500

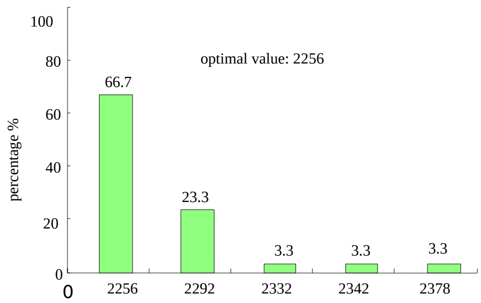


Fig.7 Solution distribution for *d*-MST using GA

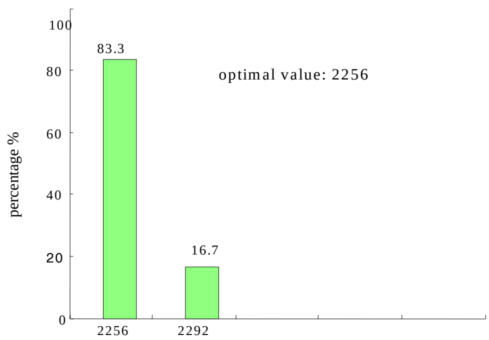


Fig. 8 Solution distribution for *d*-MST using the proposed algorithm

Table 2 Comparing NGA with GA1 and BF2

Graph	n	Degree		BF2	GA1	NGA
1	50	9	Best	2.5673	3.3540	1.6728
2	50	10	Best	2.8126	3.9985	1.7246
3	50	10	Best	3.3899	3.8800	1.9038
4	100	10	Best	2.0330	4.2999	1.2718
5	100	11	Best	1.8584	4.7599	1.3509
6	100	11	Best	1.5869	4.7639	1.3147
7	200	12	Best	1.4133	6.2079	1.1952
8	200	13	Best	1.3870	6.9406	1.2159
9	200	13	Best	1.3487	6.2461	1.1879

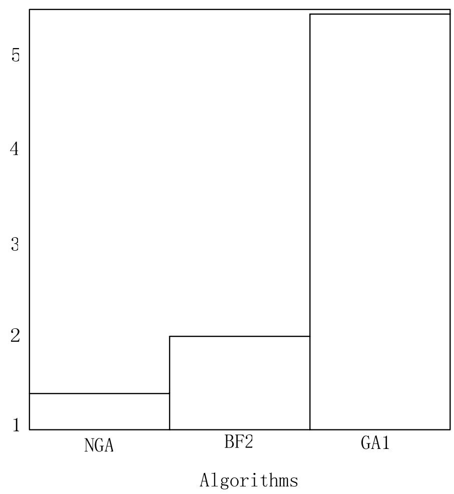


Fig.9 Mean over all graphs of solutions found

Lixia Han received the B.S. in mathematics department from Ludong University, China, in 2002, and the M.S. degrees in Shool of Science from Xidian University, China, in 2005. She is currently a Ph. D student with the School of Science, Xidian University, Xi'an, China. Her current research interests include evolutionary computation, optimization theory, algorithms.

Yuping Wang received the B.Sc. degree in mathematics from Northwest University, China, in 1983, and the Ph.D. degree in computational mathematics from Xi'an Jiaotong University, China, in 1993. He is currently a Professor with the School of Computer Science and Technology, Xidian University, Xi'an, China. He was a Visiting Research Scholar with the Chinese University of Hong Kong from January 1997 to July 1997, and at Hong Kong Baptist University from April 1998 to December 1998. His current research interests include evolutionary computation, optimization theory, algorithms, and applications.