Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2015, Article ID 601782, 13 pages
http://dx.doi.org/10.1155/2015/601782

Research Article

Novel Degree Constrained Minimum Spanning Tree Algorithm Based on an Improved Multicolony Ant Algorithm

Xuemei Sun, Cheng Chang, Hua Su, and Chuitian Rong

School of Computer Science and Software, Tianjin Polytechnic University, Tianjin 300387, China

Correspondence should be addressed to Xuemei Sun; seesea_sun@163.com

Received 22 October 2014; Accepted 12 January 2015

Academic Editor: Kun Liu

Degree constrained minimum spanning tree (DCMST) refers to constructing a spanning tree of minimum weight in a complete graph with weights on edges while the degree of each node in the spanning tree is no more than d ($d \geq 2$). The paper proposes an improved multicolony ant algorithm for degree constrained minimum spanning tree searching which enables independent search for optimal solutions among various colonies and achieving information exchanges between different colonies by information entropy. Local optimal algorithm is introduced to improve constructed spanning tree. Meanwhile, algorithm strategies in dynamic ant, random perturbations ant colony, and max-min ant system are adapted in this paper to optimize the proposed algorithm. Finally, multiple groups of experimental data show the superiority of the improved algorithm in solving the problems of degree constrained minimum spanning tree.

1. Introduction

Minimum spanning tree (MST) is a classic combinatorial optimization problem. Many engineering problems in common cases, for example, pipeline, circuit design, and transportation network, can be transformed into minimum spanning tree. Minimum spanning tree means to construct a minimum cost spanning tree with weight graph by adapting proven algorithms like avoid cycle method, tear cycle method, and so forth. However, things will be different when values, which bring about restrictions to degree of each vertex in a tree, are given in advance. It is in fact a problem of so-called degree-constrained minimum spanning tree (DCMST). As a matter of fact, it means to find out the minimum spanning tree with minimum weight while complying with the vertex constraints in all spanning trees. With the development of science and society, there is a growing need for solving the problems of DCMST in different fields such as computer communication, telephone communication, transportation, and allocation of resources, thus making DCMST optimization a problem worth studying.

As NP hard problem, various algorithms with high time complexity such as branch and bound method, approximate algorithm, iterative method, and multistart hill-climbing are exploited in DCMST in previous study. Heuristic algorithms, for instance, genetic algorithms, simulated annealing algorithm, and ant colony algorithm, have been a research focus to generate acceptable solutions in a valid time. Among these modern optimization algorithms, ant colony algorithm comes as a novel bionics algorithm and converges to the shortest path by information transferring and updating among ants. It combines distributed computation, positive feedback mechanism, and greedy search which enables fast approaching of better solutions.

Many DCNST problems are solved by ant colony algorithm [1–5]; these applications adopt ant colony algorithm with single colony. It comes with shortcomings of long search time and stagnation. Ant colony operates through organized cooperation and task-sharing in nature and their pheromone regulation mechanisms differ from one colony to another. These corporations show great significance when ant colony deals with complex tasks. Algorithms that employ multicolony ant algorithm will converge in a shorter period of time compared to those applied with single colony but feature the same ant population in total especially when the problem is in large scale. Taking all these factors into account,

the paper proposes a degree-constrained minimum spanning tree based on improved multiant colonies. The proposed algorithm features multipaths search through multiant colonies under setting conditions. Moreover, the thinking of dynamic ant colony is utilized here to dynamically adjust volatilization factor and enhancement factor during the running process in order to increase the diversity of solutions. Upper and lower bounds of pheromone are configured on the basis of the thinking of max-min ant system during its updating process. What is more, the tabu list is applied in case of local circling while random disturbance factor is set here to increase the possibility of selection among different paths so as to promote objective and diverse solutions.

2. Background

Providing G is a connected complete graph with weights on edges while the values of the weight are not negative, supposing T is the spanning tree of G, $w(T)$ means the sum of weights of T. Then construct a spanning tree with a minimum $w(T)$ under the condition that the degree of each node is no more than d, where $d \geq 2$. Spanning trees meeting the above description are named degree-constrained minimum spanning tree (DCMST).

DCMST was first proposed by Narula and Ho in 1980. Then branch and bound algorithm [6, 7], which enables optimal solutions search for DCMST with less nodes, was given at the same time. After that, a branch and cut algorithm [8] that promoted a more objective construction when searching for optimal solutions was raised by Caccetta and Hill, and so forth. In particular, Volgenant presented double solution information based Lagrangian algorithm [9] for DCMST. Soon after that, Behle et al. further optimized this problem by presenting preferential branch and cut algorithm [10] by adopting separation standards and principles of 0/1 integer programming.

With further exploration of DCMST, evolutionary algorithm [11–13] taking genetic immune thinking as a main representative emerged. During that period of time Japanese scholar Zhou and Gen achieved a major breakthrough by solving DCMST using evolutionary algorithm and encoding spanning tree by Prufe. Nevertheless, the information of degree cannot be reflected in Prufe coding and infeasible solutions generated in this procession. Since then, many evolutionary thinking based algorithms have been proposed to solve the problem of DCMST. These algorithms mainly focused on bringing improvements in evolutionary coding and these improvements are insufficient for a DCMST task. In addition, randomized primal method (RPM) [12], which is a dynamic table structure, integrated multistart hill-climbing (MHC) [14], stimulated annealing (SA), and genetic algorithm (GA) [12, 13] and the node weight information as a whole was presented by Knowles to find the optimal solutions of DCMST. Many heuristic algorithms, for instance, ant colony algorithm [3, 15–17] and particle swarm optimization algorithm [18, 19], have been applied to study DCMST from different ways and have gained positive effects.

A typical example was given in [2] to solve DCMST by an improved ant colony algorithm. In this example, a group of ants were asked to explore the given graph; then a group of signed edge set which is related to the pheromone on paths was selected as a candidate set. Afterwards, Kruskal's algorithm was adopted to construct a DCMST according to the candidate set. Finally, local optimization was carried out in DCMST to better improve the spanning tree. The algorithm proposed in this paper is basically based on ant colony algorithm and further optimizes the solutions by introducing multiant colonies.

3. Mathematical Model of DCMST

In the mathematical model of DCMST, graph $G = (V, E, W)$ means a connected undirected graph, wherein $V = \{v_1, v_2, \ldots, v_n\}$ is the vertex set, the edge set is $E = \{e_1, e_2, \ldots, e_m\}$, and $W = (\omega_{ij})_{n\times n}$ is the weight matrix. Supposing $\omega_{ij} = \omega_{ji}$, $\omega_{ii} = \infty$, $i, j = 1, 2, \ldots, n$, if $(v_i, v_j) \in E(G)$, then $\omega_{ij} = W(v_i, v_j)$; if $(v_i, v_j) \notin E(G)$, then $\omega_{ij} = \infty$. The mathematical model of DCMST can be described as

$$\min \quad Z = \sum_{i=1}^{n}\sum_{j=1}^{n}\omega_{ij}X_{ij} \tag{1}$$

$$\text{s.t.} \quad \sum_{i=1}^{n}X_{ij} \leq b_i \tag{2}$$

$$\sum_{i,j\in S}^{n} X_{ij} \leq |S| - 1, \quad \forall S \subset V,\ S \neq \varnothing \tag{3}$$

$$\sum_{i=1}^{n}\sum_{j=1}^{n}X_{ij} = n - 1 \tag{4}$$

$$X_{ij} \in \{0, 1\}, \quad i, j \in V, \tag{5}$$

wherein variable

$$X_{ij} = \begin{cases} 1, & \text{edge } (i, j) \text{ lies on the optimal tree} \\ 0, & \text{other.} \end{cases} \tag{6}$$

Condition (2) limits the number of edges that connected to node I which ensures the restrictions of degree constraint. Condition (3) makes sure that the obtained subgraph is a spanning tree of graph G. Condition (4) ensures that there are $n - 1$ edges in the final subgraph.

4. Degree-Constrained Minimum Spanning Tree Based on Multiant Colonies

In this paper, multiple ant colonies are arranged to search multiple paths following given conditions. They are supposed to search for optimal solutions independently and to communicate with each other through information entropy. Entropy is mainly used for description of the disordered relation in thermodynamics. In here, information entropy which can be expressed as $s = -k\sum_{i=1}^{n} p_i \ln p_i$ is employed to display the diversity of solutions, wherein p_i is the possibility that condition i can be determined while $p_i \geq 0$, $\sum_{i=1}^{n} p_i = 1$. The size of information entropy dominates the diversity of

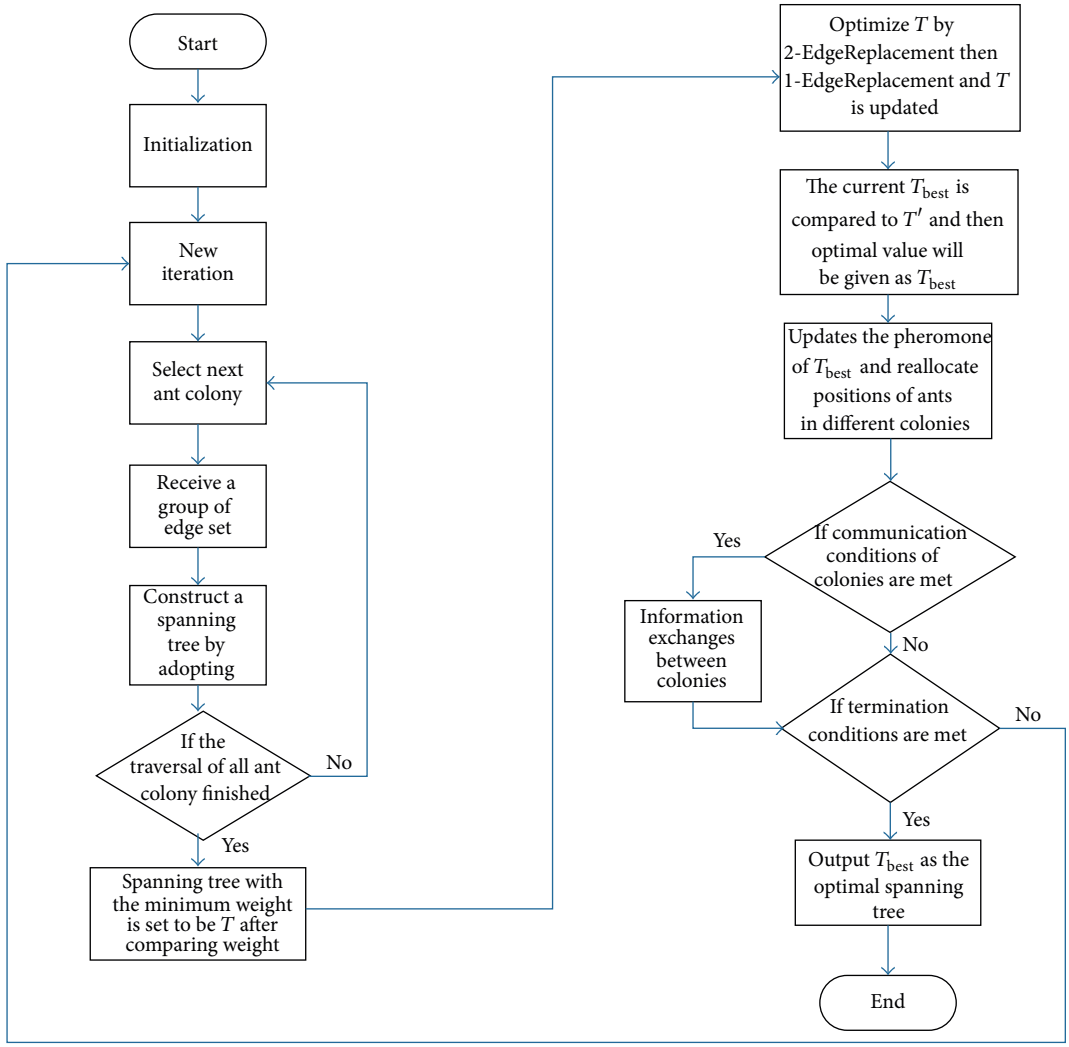

Figure 1: Flow chart of the algorithm.

solutions and it is also an important basis for pheromone updating between ant colonies.

Besides, the improved algorithm enlarged search area of ant and added to the diversity of solutions by dynamically adjusting volatilization factor and enhancement factor to affect the distribution of pheromone. By introducing adaptive optimization strategy in case of local circling and random disturbance factor to increase the possibility that each path is being selected, the obtained solutions are supposed to be objective and diverse.

At the beginning of the experiment, each ant is allocated with an initial position; then initial values of pheromone of edges are given. After DCMST T_{best} is constructed, ant colonies start the search process and a set of edge set TE will be obtained. Kruskal algorithm then will be applied to construct a spanning tree according to TE. After all rge spanning trees are constructed by each colony, all these spanning trees will be compared with each other and the optimal spanning tree with the least weight will be note into T_{better}. Algorithms 2-EdgeReplacement and 1-EdgeRepalcement will be exploited to optimize T_{better}. Then, comparing T_{better} with T_{best} after iteration, the one that features the smaller weight is set to be T_{best}. Next, globally update the pheromone of T_{best} and reallocate positions of each ant. If communication requirements are met between colonies, then pheromone updating will be conducted between colonies, or next iteration will be conducted. Final solution T_{best} will not be obtained until all the iterations are complete. See Algorithm 1.

The flow chart is as shown in Figure 1.

4.1. The Initialization Stage. The edge weight will be computed according to the distances between node sets when initialization begins. Maximum weight M and minimum weight m will be recorded. Then the initial pheromone concentration of each edge is acquired by analyzing obtained weight. Assuming the range of weight lies between $(M - m)/3$ and $4(M - m)/3$, the initial edge set will be obtained according to initial pheromone matrix. After that, each ant will be allocated with an initial position. See Algorithm 2.

```
Input:
  G = (V, E): a complete graph with weight;
  ω: function to calculate weight;
  d: degree constraint of vertex, d ≥ 2;
  MA: the amount of ant colonies
Output:
  A degree constrained minimum spanning tree T
Begin
  //Initialization
  i ← 1; i_best ← 0; i_restart ← 0
  Create the Multiple set of ants A[MA] of size |V| with the number of colonies k
  InitAntsEdges(V, E, ω)
  T_best ← ConstructSpanningTree(E, d)
  while i < i_max and i − i_best < i_stop do
    for ma = 1 to MA do
      AntsMove(A[ma], E) //Exploration
      s ← Calculation Entropy
      T ← ConstructSpanningTree(E, d)
      If ω(T) < ω(T_better) then
        T_better ← T
        ma_better ← ma
      //local optimization
      T_better ← 2-EdgeReplacement(T_better, E, V, ω)
      T_better ← 1-EdgeReplacement(T_better, E, V, ω)
    If ω(T_better) < ω(T_best) then
      T_best ← T_better
      i_best ← i
    foreach e ∈ T_best do //enhance
      e·phm ← γ · e·phm
    if i_max(i_best, i_restart) > R then
      i_restart ← i
      Restart(T_best)
    if i mod gap(s) = 0 then
      ExchangePheromone()
    i ← i + 1
    ResetAnts(A)
    Update parameters γ and η
  return T_best
```

Algorithm 1

```
InitAntsEdges(V, E, ω)
begin
  for i = 0 to |V| − 1 do
    Ant[i]·location ← V[i]
    Ant[i]·tabuList ← ∅
  foreach e ∈ E do
    e·initPhm ← (M − ω(e)) + (M − m)/3
    e·phm ← e·initPhm
    e·nVisited ← 0
```

Algorithm 2

4.2. The Search Stage. The transfer rule of an improved ant colony algorithm is employed when ants start their search for next target position in the search stage.

The algorithm integrates the characteristics of optimized algorithms like dynamic ant colony algorithm and random disturbance ant colony algorithm as a whole and each ant is allocated with its own tabu list. Ants will select next target according to transfer rule in

$$C_{ij}^{k} = \begin{cases} sp_{ij}^{k} & q \le q_0 \\ p_{ij}^{k} & q > q_0 \\ 0 & \text{other.} \end{cases} \tag{7}$$

In this formula, q is a random number within 0~1. q_0 is a constant; it is set to be 0.5 in here. When q is less than or equal to q_0, according to random disturbance strategy, the next target position will be the node when sp_{ij}^{k} takes the largest value. Consider

$$sp_{ij}^{k} = \frac{\left(\tau_{ij}{}^{\alpha} \cdot \eta_{ij}{}^{\beta}\right)^{\gamma}}{\sum \tau_{ij}{}^{\alpha} \cdot \eta_{ij}{}^{\beta}}, \tag{8}$$

```
AntsMove(A[ma], E)
begin
  for i = 1 to maxSteps do
    if i mod updatePeriod = 0 then
      UpdatePheromone(E)
    foreach a ∈ A[ma] do
      nAttempts ← 0
      moved ← False
      while not moved and nAttempts < 5 do
        v1 ← a·location
        Select an edge (v1, v2) at random and proportional to (v1, v2)·phm
        if v2 ∉ a·tabuList then
          add v2 into a·tabuList
          a·location ← v2
          (v1, v2)·nVisited++
          moved ← True
        else nAttempts++
  s ← Calculation Entropy of the current colony
```

Algorithm 3

wherein α is equal to 1 and β changes with the number of iterations, recorded as mn, by a kind of dynamic ant colony strategy. $\beta = 5$ when the number of iterations lies in $1 \sim mn/3$. When it comes to be in $mn/3 \sim mn/2$, $\beta = 4$. $\beta = 3$ when the number lies in $mn/2 \sim mn2/3$. Similarly, $\beta = 2$ when this number lies in $mn2/3 \sim mn$.

Γ calculated by $\gamma = a \cdot e^{b/k}$ is the disturbance factor. a is a random number within 0~2. b is equal to 4 and k is the number of iterations.

When q is larger than q_0,

$$p_{ij}^{k} = \frac{{\tau_{ij}}^{\alpha} \cdot {\eta_{ij}}^{\beta}}{\sum {\tau_{ij}}^{\alpha} \cdot {\eta_{ij}}^{\beta}}. \tag{9}$$

p_{ij}^{k} means the possibility that j is being selected; the possibility increases as p_{ij}^{k} grows rather than forcing sp_{ij}^{k} to be the maximum node. Then next target position will be determined and add this position to tabu list of this ant in case of being selected again; then add 1 to the number of visits.

Information entropy of a colony will be calculated as (10) according to obtained p_{ij}^{k} of each ant after a colony completes searching:

$$s = -k \sum_{i=1}^{m} p_i \ln p_i. \tag{10}$$

m represents the number of ants in a colony. The calculated information entropy of colonies will be a basis for colonies selection when pheromone exchanges. See Algorithm 3.

4.3. Updating of Pheromone. All ant colonies search for edges following a given step length. All ants will not stop moving in parallel in each step and updating pheromone on edges that have been searched until all the steps have been finished. Pheromone updating is set to be operated after finishing a group of specific steps instead of updating pheromone concentration after each step in order to promote more effective search. Pheromone is supposed to be updated when an ant has finished 1/3 of its maximum step.

```
UpdatePheromone(E)
begin
  foreach e ∈ E do
    e·phm ← (1 − η) × e·phm + e·nVisited × e·initPhm
    e·nVisited ← 0
    if e·phm > maxPhm then
      e·phm ← maxPhm − e·initPhm
    if e·phm < minPhm then
      e·phm ← minPhm + e·initPhm
```

Algorithm 4

Pheromone on each path ranges between minPhm and maxPhm. It is similar to max-min ant system, which will be of help to prevent the solution from falling into local optimal:

$$\begin{aligned} \text{maxPhm} &= 1000 \cdot (M - m) + \frac{(M - m)}{3}, \\ \text{minPhm} &= \frac{(M - m)}{3}. \end{aligned} \tag{11}$$

M and m are maximum weight and minimum weight referred to at the beginning.

A counter will be set on each edge to know how many times it has been visited after an update is complete. Local update of pheromone is arranged as the formula lists as follows:

$$\text{phm} = (1 - \eta) \cdot \text{phm} + n\text{Visited} \cdot \text{initPhm}. \tag{12}$$

```
ConstructSpanningTree(E, d)
begin
  T_n ← ∅
  Sort E in order of descreasing pheromone level
  C ← top nCandidates edges from E
  Sort C in order of increasing edges cost
    while |T_n| < |V| − 1 do
      if C ≠ ∅ then
        Let e be the next edge in C
        Remove e from C
        if adding e into T_n with meet the degree constraint then
          T_n = T_n ∪ {e}
        else
        C ← next nCandidates edges from E
        Sort C in order of increasing edges cost
return T_n
```

Algorithm 5

η means the volatilization factor and it is updating with the operating of algorithm. Consider

$$\eta = \eta \cdot \left(\frac{n\text{Visited}}{\text{phm}}\right)^{k}. \tag{13}$$

initPhm is the initial pheromone value of edge and nVisited implies the number on edges being visited. k is a constant and, in here, it is set to be 0.5. It should be noted that when the value of pheromone on a path exceeds threshold value, the value of pheromone will be adjusted by adjusting initial pheromone values of edges for better solution. See Algorithm 4.

4.4. Constructing Spanning Tree. A group of edges will be obtained after an ant colony completes searching. Pheromone matrix is adopted to construct a corresponding DCMST due to the absolute connectivity of obtained edge sets and they are all labeled with high pheromone values.

Sort edges sets by pheromone values, from largest to smallest. Construct a candidate set by choosing the top nCandidates edges and arrange them from smallest to largest by weight values. Next, a spanning tree in which the degrees of nodes meet constraints will be constructed. If the selected nCandidates set does not meet the requirements to construct a spanning tree, another nCandidates set will be selected to construct a spanning tree until a degree constrained spanning tree was constructed. After all ant colonies constructed their own spanning tree, the one that owns the smallest weight will be selected as the optimal spanning tree in this iteration, labeled as T. Then its pheromone matrix and colony number will be recorded. See Algorithm 5.

4.5. Local Optimization Stage. Spanning tree T will be processed by two local optimization algorithms of 2-EdgeReplacement and 1-EdgeReplacement. 2-EdgeReplacement means to replace any two edges of spanning tree with two other edges, wherein the weights of two other edges are smaller than two previous edges being replaced. Furthermore, spanning tree has to meet degree constraints after two edges being replaced. What makes 1-EdgeReplacement different from 2-EdgeReplacement is that only one edge rather than two edges is replaced in a spanning tree in 1-EdgeReplacement. Finally, optimized spanning tree is noted as T'. See Algorithms 6 and 7.

4.6. Global Optimization Stage. Spanning tree T' after iterative optimization is compared with optimal spanning tree T_{best} generated in latest iteration; the one that gets the smaller weight will be noted as T_{best}. Then pheromone will be updated globally in T_{best} and ants will be allocated with new positions. Only if communication requirements between colonies are met will pheromone be updated between colonies, or a new iteration will be conducted.

4.7. Information Exchanges between Colonies. Information exchanges between colonies are conducted at certain time intervals, which is determined according to information entropy of all colonies; in other words, it changes by the convergence of all colonies. Time intervals gap of information exchanges between colonies is in accordance with the following formula:

$$\text{gap} = k_1 \cdot e^{\sum_{i=1}^{h} s_i / h}, \tag{14}$$

wherein k_1 is a constant, h is the number of subcolonies, and s_i is the information entropy of the ith subcolony.

Selection criteria of colonies, where information exchanges existed between colonies, are determined by information entropies of each colony. Provided that colony j is the selected colony of colony i for information exchanging, then j is determined by the following formula:

$$j = \arg \max_{1 \le j \le h} \left(\left|s_i - s_j\right|\right), \tag{15}$$

in which s_i and s_j are information entropies of colonies i and j, respectively. Colonies with greater entropy will choose colonies with smaller entropy as objects to exchange information. Then colonies with small information entropy are

```
2-EdgeReplacement(T_better, E, V, ω)
begin
  T_n ← T_better
  nTries ← 0
  while nTries < |V|/2 do
    e_1 ← a random edge in T_n
    e_b ← e_1; c_b ← 0
    foreach e_2 ∈ T_better and e_1, e_2 are not adjacent do
      {e_r1, e_r2} ← rep2Opt(e_1, e_2)
      if ω(e_1) + ω(e_2) − ω(e_r1) − ω(e_r2) > c_b then
      c_b ← ω(e_1) + ω(e_2) − ω(e_r1) − ω(e_r2)
      e_b ← e_2
    if c_b > 0 then
      T_n ← (T_n − {e_1, e_b}) ∪ rep2Opt(e_1, e_b)
      nTries ← 0
    else
      nTries ← nTries + 1
  return T_n
```

Algorithm 6

```
1-EdgeReplacement(T_better, E, V, ω)
begin
  T_n ← T_better
  Sort E in order of increasing edges cost
  repeat
    changed ← False
    Sort T_n in order of increasing edges cost
    j ← |V| − 1
    while j ≥ 0 do
      k ← 0
      while ω(T_n[j]) > ω(E[k]) do
        if swapping E[k] and T_n[j] creates no cycle in T_n and satisfies degree constraint then
          T_n ← T_n ∪ {E[k]} − {T_n[j]}
          changed ← True
          break
        k ← k + 1
      j ← j + 1
until not changed
return T_n
```

Algorithm 7

concentratedly distributed and information exchanges with high entropy colonies will help to improve their pheromone amount. Similarly, colonies with high information entropy will be distributed scatteredly to help to improve colonies with small information entropy.

Pheromone updates will be conducted depending on formula (16) when colony i selects colony j as an object to exchange information. Consider

$$\tau_{uv}^{i} = \tau_{uv}^{i} + \lambda \Delta \tau_{uv}^{i}. \tag{16}$$

$\lambda = s_i - s_j$, $\lambda_{\min} < \lambda < \lambda_{\max}$, and $\lambda_{\min}$ and $\lambda_{\max}$ are constants and present the minimum and maximum update coefficient, respectively. $\Delta\tau_{uv}^{i}$ is the pheromone of colony j on path (u, v). See Algorithm 8.

5. Simulation and Analysis

5.1. Experimental Data. The data in Table 1 was selected as experimental subjects. Single colony ant algorithm was compared with multicolony ant algorithm. Experimental parameters were set as follows: maximum number of loop iterations $i_{\max} = 100$, $i_{\text{stop}} = 25$. The number of ant colonies is 3, the maximum search step of ant $m_{\text{Steps}} = 75$, update cycle of local pheromone updatePeriod = $m_{\text{Steps}}/3$, and the degree of vertex $d = 2, 3, 4, 5$.

5.2. Experimental Platform. The algorithm is coded with standard Java and has been compiled and debugged on the platform of Eclipse. A computer installed with 32-bit Windows 7 operating system and equipped with 3 G RAM

```
ExchangePheromone(A, E)
begin
  λ ← 0
  for i = 0 to |MA| − 1 do
    for j = 0 to |MA| − 1 do
      if |s_i − s_j| > λ then
        λ ← |s_i − s_j|
        mj ← j
    for k = 0 to |E| − 1 do
      E_i[k]·phm ← E_i[k]·phm + λ × Emj[k]·phm
```

Algorithm 8

Table 1

Sample graph (G)	Number of vertexes $\|V\|$	Sample graph (G)	Number of vertexes $\|V\|$
att48	20	rand100	100
bier127	127	rat99	99
eil51	51	pr107	107
eil76	76	pr136	136
eil101	101	pr152	152
ulysses16	16	ulysses22	22
kroA100	100	chn31	31
kroA200	200	ch130	130
kroA150	150	ch150	150
rand50	50	st70	70
rand75	75	overil30	30
danzig42	42	bayg29	29
gr96	96	burma14	14
gr120	120	berlin52	52
gr137	137		

and 2.20 GHz CPU was used to operate the algorithm. Experimental results are obtained by operating the proposed algorithm 20 times for each sample data set.

5.3. Parameters Configuration. Parameters are configured as Table 2 in experiment.

5.4. Evaluation Index. In experiment, the optimal value, average value, standard deviation, coefficient of variation CV, and mean deviation value Gap were selected as evaluation indexes of experimental results, wherein coefficient of variation CV reflects the robustness of the algorithm: $CV = \sigma/\mu \times 100$. The proposed algorithm tends to be more stable as the value of CV gets smaller. Mean deviation value Gap illustrates the possibility of better solution obtaining $Gap = (C_{mean} - C_{opt})/C_{opt} \times 100$.

5.5. Experimental Results and Analysis. Experiment was conducted when degree of vertex d was set to be 2, 3, 4, 5, respectively. Each sample graph run 20 times; then a minimum spanning tree and its weight was gained. If the value d of the minimum spanning tree is not within the scope of the above set, the output answer is invalid.

Table 2

Parameters	Values	Notes
i_{max}	100	Maximum number of iterations
i_{stop}	30	Maximum iterations without improvement
maxSteps	75	Number of steps that an ant traverses each iteration
nCandidates	$5\|V\|$	Candidate set cardinality
η	0.5	Initial pheromone evaporation factor
γ	1.5	Initial pheromone enhancement factor
updatePeriod	maxSteps/3	Pheromone update period
R	10	Number of iterations without improvement before escaping

Tables 3–6 provide the statistical results of optimal value, average value, standard deviation, coefficient of variation CV, and mean deviation value Gap after 20 times of operation for all the sample graphs under various conditions.

According to the deviation coefficient CV and mean deviation value Gap listed in the table, these values are large when $d = 2$. Significant changes can be observed comparing to optimal solution and the changes in optimal search are obvious.

With the increase of d, the restrictions of spanning tree construction will not be tightly controlled. It can be drawn from the table that optimal solution, taking graph att48 as an example, tends to be stable while changes in deviation coefficient and mean deviation value approach zero. Optimal tree weight tends to be stable and algorithm optimization gets improved with changes of d by adopting improved algorithm. With the increase of degree in sample graph, variation ranges of deviation coefficient and standard deviation value were kept within 0.00%–6.00%, even close to 0.00%. The above results show advantages of improved ant colony algorithm in solving DCMST of dense graph and advantages of ant colony algorithm in solving combinatorial optimization problem in minimum spanning tree.

When d is equal to 4 or 5, obtained solutions optimized by ant colony algorithm are basically stable in a certain range and optimized performance of algorithm and optimized values of results tend to be constant. The results imply that there are no more promotions on solution searching when degree constraints go over a certain range. Therefore, in practical applications, the degree of limitation on the spanning tree should be more considered. Excessive restrictions on spanning tree will not result in an expected optimization effect.

Table 3: Experimental results when $d = 2$.

Sample graph (G)	Optimal value (best)	Average value (μ)	Standard deviation (σ)	Coefficient of variation (CV)	Mean deviation value (Gap)
att48	32699.57	32865.26	193.35	1.00	1.00
bier127	117484.46	118228.23	0.00	1.00	1.00
eil51	402.00	413.50	9.19	2.00	4.00
eil76	524.00	537.50	0.71	0.00	3.00
eil101	519.00	534.50	2.12	0.00	3.00
ulysses16	43.00	45.50	2.12	5.00	6.00
ulysses22	43.00	45.00	2.83	6.00	5.00
kroA100	21802.88	22214.5	89.54	0.00	2.00
kroA200	32001.91	32139.63	145.86	0.00	0.00
rand50	5450.00	5450.00	0.00	0.00	0.00
rand75	7329.46	7384.14	0.00	0.00	1.00
rand100	8212.65	8398.32	262.58	3.00	2.00
rat99	1203.00	1250.00	41.01	3.00	4.00
pr107	38405.70	38943.71	59.19	0.00	1.00
pr136	105046.60	105417.40	405.11	0.00	0.00
pr152	67577.00	67880.68	312.56	0.00	0.00
chn31	13870.83	14041.91	175.49	1.00	1.00
ch130	6268.00	6389.03	171.16	3.00	2.00
ch150	6787.99	6875.00	0.00	0.00	1.00
overil30	367.23	372.11	6.91	2.00	1.00

Table 4: Experimental results when $d = 3$.

Sample graph (G)	Optimal value (best)	Average value (μ)	Standard deviation (σ)	Coefficient of variation (CV)	Mean deviation value (Gap)
att48	27855.00	27855.00	0.00	0.00	0.00
bier127	94680.00	94680.00	0.00	0.00	0.00
eil51	362.00	363.50	0.71	0.00	0.00
eil76	456.00	459.00	2.83	1.00	1.00
eil101	455.00	458.00	0.00	0.00	1.00
ulysses16	43.00	45.50	2.12	5.00	6.00
ulysses22	43.00	45.00	2.83	6.00	5.00
kroA100	18731.00	18731.00	0.00	0.00	0.00
kroA200	15853.00	25853.00	0.00	0.00	0.00
rand50	4967.00	4967.00	0.00	0.00	0.00
rand75	5996.00	5996.00	0.00	0.00	0.00
rand100	6907.00	6907.00	0.00	0.00	0.00
rat99	1082.00	1082.00	0.00	0.00	0.00
pr107	34890.00	34890.00	0.00	0.00	0.00
pr136	88940.00	88940.00	0.00	0.00	0.00
pr152	59105.00	59105.00	0.00	0.00	0.00
chn31	12111.00	12111.00	0.00	0.00	0.00
ch130	5102.00	5102.50	0.71	0.00	0.00
ch150	5812.00	5812.00	0.00	0.00	0.00
overil30	334.00	334.50	0.71	0.00	0.00

Table 5: Experimental results when $d = 4$.

Sample graph (G)	Optimal value (best)	Average value (μ)	Standard deviation (σ)	Coefficient of variation (CV)	Mean deviation value (Gap)
att48	27623.00	27623.00	0.00	0.00	0.00
bier127	94680.00	94680.00	0.00	0.00	0.00
eil76	456.00	457.50	2.12	0.00	0.00
eil51	359.00	359.00	0.00	0.00	0.00
eil101	456.00	458.00	1.41	0.00	0.00
ulysses16	41.00	42.00	0.00	0.00	2.00
ulysses22	41.00	43.00	0.00	0.00	0.00
kroA100	18731.00	18731.00	0.00	0.00	0.00
kroA200	25844.00	25844.00	0.00	0.00	0.00
rand50	4665.00	4665.00	0.00	0.00	0.00
rand75	5996.00	5996.00	0.00	0.00	0.00
rand100	6889.00	6889.00	0.00	0.00	0.00
rat99	1082.00	1082.00	0.00	0.00	0.00
pr107	34756.00	34756.00	0.00	0.00	0.00
pr136	88940.00	88940.00	0.00	0.00	0.00
pr152	59105.00	59105.00	0.00	0.00	0.00
chn31	12111.00	12111.00	0.00	0.00	0.00
ch130	5103.00	5103.50	0.00	0.00	0.00
ch150	5809.00	5809.00	0.00	0.00	0.00
overil30	334.00	334.50	0.71	0.00	0.00

Table 6: Experimental results when $d = 5$.

Sample graph (G)	Optimal value (best)	Average value (μ)	Standard deviation (σ)	Coefficient of variation (CV)	Mean deviation value (Gap)
att48	27623.00	27623.00	0.00	0.00	0.00
bier127	94680.00	94680.00	0.00	0.00	0.00
eil76	455.00	460	0.00	0.00	1.00
eil51	360.00	360.00	0.00	0.00	0.00
eil101	456.00	458.50	2.12	0.00	1.00
ulysses16	41.00	41.50	0.71	2.00	1.00
ulysses22	41.00	42.50	0.71	2.00	4.00
kroA100	18731.00	18731.00	0.00	0.00	0.00
kroA200	25844.00	25844.00	0.00	0.00	0.00
rand50	4665.00	4665.00	0.00	0.00	0.00
rand75	5996.00	5996.00	0.00	0.00	0.00
rand100	6889.00	6889.00	0.00	0.00	0.00
rat99	1082.00	1082.00	0.00	0.00	0.00
pr107	34756.00	34756.00	0.00	0.00	0.00
pr136	88940.00	88940.00	0.00	0.00	0.00
pr152	59105.00	59105.00	0.00	0.00	0.00
chn31	12111.00	12111.00	0.00	0.00	0.00
ch130	5102.00	5102.50	0.71	0.00	0.00
ch150	5809.00	5809.00	0.00	0.00	0.00
overil30	334.00	334.50	0.71	0.00	0.00

Table 7: Results of comparison for multiple and single colony ant algorithm.

Sample graph	$d = 2$			
	Average solution (μ)		Optimal solution (best)	
	Multiple	Single	Multiple	Single
eil51	374.43	413.50	398.00	402.00
eil76	534.98	537.50	519.00	524.00
chn31	14012.02	14041.91	13870.86	13870.83
overil30	371.92	372.11	365.00	367.23
ulysses16	45.00	45.50	43.00	44.00
ulysses22	46.50	45.00	43.00	44.00
rand75	7390.14	7384.14	7326.04	7329.46
rand100	8248.18	8398.32	8166.56	8212.65
gr96	462.50	463.70	445.00	449.00
gr120	1678.90	1692.41	1631.00	1663.00
gr137	686.80	690.30	682.00	683.00
bayg29	8107.22	8597.52	8432.44	8439.00
beilin52	7702.51	7761.52	7442.58	7627.00
burma14	19.70	19.90	18.00	19.00
danzig42	647.16	656.39	640.65	642.92
st70	662.73	663.90	643.00	650.00
kroA150	27446.79	27611.06	27363.01	27471.08

Table 8: Results of comparison for multiple and single colony ant algorithm.

Sample graph	$d = 3$			
	Average solution (μ)		Optimal solution (best)	
	Multiple	Single	Multiple	Single
eil51	363.25	363.50	362.00	363.00
eil76	459.00	460.00	454.00	456.00
gr96	398.40	399.75	395.00	397.00
bayg29	7456.81	7459.00	7415.16	7459.00
danzig42	565.75	566.15	565.00	566.00
ulysses22	42.60	45.00	41.00	43.00
st70	544.50	544.65	542.00	543.00

By comparing multicolony ant algorithm with single colony ant algorithm [2], multicolony ant algorithm is obviously more effective in solution optimization which proves that multicolony ant algorithm shows greater advantages in DCMST optimal searching. Conditions under $d = 2$ and $d = 3$ are selected here to illustrate the proposed opinion. Solution assessment of these two algorithms is evaluated by average solution (μ) and optimal solution (best). Solution comparisons of multicolony ant algorithm and single colony ant algorithm under $d = 2$ and $d = 3$ are listed in Tables 7 and 8.

Experimental data as shown in Tables 7 and 8 show that average weight and optimal weight of DCMST obtained by multicolony ant algorithm are obviously smaller than those solved by single colony ant algorithm. Multicolony ant algorithm yields better optimization results than single colony ant algorithm which implies that coordination and cooperation through information entropy in multiant colonies promotes better convergence of algorithm and expands search space which adds to the diversity of solution. By comparing experimental results of $d = 2$ and $d = 3$, it can be seen that the tougher the degree constraints of minimum spanning tree are, the more accurate the solution searched by multicolony ant algorithm is. The above analysis proved that the algorithm has strong search ability.

It can be drawn from the above analysis that multicolony ant algorithm shows greater superiority in solving DCMST. It features outstanding ability of optimal solution finding and also it will extensively distribute the solutions and obtain new optimal solutions, thus making it a better algorithm for searching DCMST with smallest weight.

6. Conclusion

The paper carried out studies on applications of ant colony algorithm in DCMST optimization. Many problems remain

unresolved in this study since researches on swarm intelligence of ant colony algorithm are still in a primary stage. Works to be carried out next are as follows: to find out links among genetic algorithm, hill-climbing algorithm, and particle swarm optimization algorithm and integrate them into a novel improved algorithm. By adopting it to optimizations like DCMST, practical application demands can be met.

Another work to be done is to improve parameters selection criteria of algorithm optimization and assessment standard of algorithm stability on account of the lack of related study in depth while ant colony algorithm is a heuristic search algorithm and also due to few works being carried out in this field to perfectly prove its correctness.

Owing to the complexity and diversity of practical problems and variations in working environment, multilayered algorithms have not been reached in this paper. In addition, complex theory has become a powerful framework to deal with the complexity problems in real-world systems [20–23]. Thus, concurrent collaborations of multiant colonies will be a key emphasis in work so as to increase the speed and efficiency of solving tasks.

In a word, related works presented in this paper reflect smart features of intelligent computing and dynamic execution in parallel, and significant achievements have been made by it in DCMST.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

The project is supported by the National Natural Science Foundation of China (Grant no. 61173032), the Young Scientists Fund of the National Natural Science Foundation of China (Grant no. 61402329), and the Natural Science Foundation of Tianjin, China (Grant nos. 13JCYBJC15500 and 12JCYBJC31900).

References

[1] M. N. Doan, "An effective ant-based algorithm for the degree-constrained minimum spanning tree problem," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '07)*, pp. 485–491, IEEE, September 2007.

[2] T. N. Bui, X. Deng, and C. M. Zrncic, "An improved ant-based algorithm for the degree-constrained minimum spanning tree problem," *IEEE Transactions on Evolutionary Computation*, vol. 16, no. 2, pp. 266–278, 2012.

[3] F. Neumann and C. Witt, "Ant colony optimization and the minimum spanning tree problem," *Theoretical Computer Science*, vol. 411, no. 25, pp. 2406–2413, 2010.

[4] T. N. Bui and C. M. Zrncic, "An ant-based algorithm for finding degree-constrained minimum spanning tree," in *Proceedings of the 8th Annual Genetic and Evolutionary Computation Conference*, pp. 11–18, ACM, July 2006.

[5] Y.-T. Bau, C.-K. Ho, and H.-T. Ewe, "Ant colony optimization approaches to the degree-constrained minimum spanning tree problem," *Journal of Information Science and Engineering*, vol. 24, no. 4, pp. 1081–1094, 2008.

[6] S. C. Narula and C. A. Ho, "Degree-constrained minimum spanning tree," *Computers and Operations Research*, vol. 7, no. 4, pp. 239–249, 1980.

[7] M. Savelsbergh and T. Volgenant, "Edge exchanges in the degree-constrained minimum spanning tree problem," *Computers and Operations Research*, vol. 12, no. 4, pp. 341–348, 1985.

[8] L. Caccetta and S. P. Hill, "A branch and cut method for the degree-constrained minimum spanning tree problem," *Networks*, vol. 37, no. 2, pp. 74–83, 2001.

[9] A. Volgenant, "A Lagrangean approach to the degree-constrained minimum spanning tree problem," *European Journal of Operational Research*, vol. 39, no. 3, pp. 325–331, 1989.

[10] M. Behle, M. Jünger, and F. Liers, "A primal branch-and-cut algorithm for the degree-constrained minimum spanning tree problem," in *Experimental Algorithms*, vol. 4525 of *Lecture Notes in Computer Science*, pp. 379–392, Springer, Berlin, Germany, 2007.

[11] G. R. Raidl, "An efficient evolutionary algorithm for the degree-constrained minimum spanning tree problem," in *Proceedings of the Congress on Evolutionary Computation*, vol. 1, pp. 104–111, La Jolla, Calif, USA, 2000.

[12] G. R. Raidl and B. A. Julstrom, "A weighted coding in a genetic algorithm for the degree-constrained minimum spanning tree problem," in *Proceedings of the ACM Symposium on Applied Computing (SAC '00)*, vol. 1, pp. 440–445, ACM, March 2000.

[13] J. Knowles and D. Corne, "A new evolutionary approach to the degree-constrained minimum spanning tree problem," *IEEE Transactions on Evolutionary Computation*, vol. 4, no. 2, pp. 125–134, 2000.

[14] M. Dorigo and C. Blum, "Ant colony optimization theory: a survey," *Theoretical Computer Science*, vol. 344, no. 2-3, pp. 243–278, 2005.

[15] T. Stützle and H. H. Hoos, "MAX-MIN ant system," *Future Generation Computer Systems*, vol. 16, no. 8, pp. 889–914, 2000.

[16] B. Doerr and D. Johannsen, "Refined runtime analysis of a basic ant colony optimization algorithm," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '07)*, pp. 501–507, Singapore, September 2007.

[17] M. Krishnamoorthy, A. T. Ernst, and Y. M. Sharaiha, "Comparison of algorithms for the degree constrained minimum spanning tree," *Journal of Heuristics*, vol. 7, no. 6, pp. 587–611, 2001.

[18] H. T. T. Binh and T. B. Nguyen, "New particle swarm optimization algorithm for solving degree constrained minimum spanning tree problem," in *PRICAI 2008: Trends in Artificial Intelligence*, vol. 5351 of *Lecture Notes in Computer Science*, pp. 1077–1085, Springer, Berlin, Germany, 2008.

[19] A. T. Ernst, "A hybrid Lagrangian particle swarm optimization algorithm for the degree-constrained minimum spanning tree problem," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '10)*, IEEE, 2010.

[20] Z.-Q. Ma, C.-Y. Xia, S.-W. Sun, L. Wang, H.-B. Wang, and J. Wang, "Heterogeneous link weight promotes the cooperation in spatial prisoner's dilemma," *International Journal of Modern Physics C*, vol. 22, no. 11, pp. 1257–1268, 2011.

[21] J.-J. Zhang, H.-Y. Ning, Z.-Y. Yin et al., "A novel snowdrift game model with edge weighting mechanism on the square lattice," *Frontiers of Physics*, vol. 7, no. 3, pp. 366–372, 2012.

[22] C. Y. Xia, S. W. Sun, Z. X. Liu, Z. Q. Chen, and Z. Z. Yuan, "Epidemics of sirs model with nonuniform transmission on scale-free networks," *International Journal of Modern Physics B*, vol. 23, no. 9, pp. 2203–2213, 2009.

[23] C. Y. Xia, Z. X. Liu, Z. Q. Chen, and Z. Z. Yuan, "SIRS epidemic model with direct immunization on complex networks," *Control and Decision*, vol. 23, no. 4, pp. 468–472, 2008.