Swarm Intelligence Inspired Multicast Routing: An Ant Colony Optimization Approach*

Xiao-Min Hu, Jun Zhang**, and Li-Ming Zhang

Department of Computer Science, Sun Yat-Sen University, Guangzhou, P.R. China
junzhang@ieee.org

Abstract. The advancement of network induces great demands on a series of applications such as the multicast routing. This paper firstly makes a brief review on the algorithms in solving routing problems. Then it proposes a novel algorithm called the distance complete ant colony system (DCACS), which is aimed at solving the multicast routing problem by utilizing the ants to search for the best routes to send data packets from a source node to a group of destinations. The algorithm bases on the framework of the ant colony system (ACS) and adopts the Prim's algorithm to probabilistically construct a tree. Both the pheromone and heuristics influence the selection of the nodes. The destination nodes in the multicast network are given priority in the selection by the heuristics and a proper reinforcement proportion to the destination nodes is studied in the case experiments. Three types of heuristics are tested, and the results show that a modest heuristic reinforcement to the destination nodes can accelerate the convergence of the algorithm and achieve better results.

1 Introduction

With the development of network technologies, application demands and quality requests are getting higher and higher. As the topologies of networks are complex, efficient packet routing methods are quite meaningful to the real world [1].

The complexity of the network makes deterministic algorithms incapable to react in time. So a variety of heuristic algorithms for network routing emerged in recent years. Di Caro and Dorigo [2] introduced an approach called AntNet to the adaptive learning of routing tables in a communication network. Gelenbe et al. [3] recently proposed an experimental investigation of path discovery using genetic algorithms (GAs). There are also some algorithms for network routing, such as some agent-based algorithms (eg., the ant-based control (ABC) system [4]), neural heuristics [5]. These algorithms are promising but still need enhancements.

Multicast routing [6]-[11] first appeared in ARPANET in packet-switched store-and-forward communication networks. The packets in multicast routing

* This work was supported by NSFC Joint Fund with Guangdong, Key Project No. U0835002, NSF of China Project No.60573066 and the Scientific Research Foundation for the Returned Overseas Chinese Scholars, State Education Ministry, P.R. China.

** Corresponding author.

M. Giacobini et al. (Eds.): EvoWorkshops 2009, LNCS 5484, pp. 51–60, 2009.

are sent from a source node to several destinations without or with certain constraints such as the requirements of the quality-of-service (QoS) and real-time communications. The unconstrained multicast routing problem, which is known to be NP-complete [10], is equivalent to the Steiner tree problem in graphs.

This paper aims at solving the unconstrained multicast routing problem using an ant algorithm. The proposed algorithm, which is called the distance complete ant colony system (DCACS), bases on an ant colony system framework [12] and takes advantage of the distance complete graph (DCG) to find the multicast tree with the lowest cost in the network. Each ant starts from a randomly chosen destination nodes (including the source node) and follows the structure of the Prim's minimum spanning tree (MST) [13] algorithm to construct the multicast tree. The ants select a next node in a deterministic way, or explore one by a probabilistic selection. After an ant has finished its tour, the solution is checked further for a potentially better tree by the classical Prim's MST algorithm and the redundancy trimming strategy. The pheromone in the network is updated by the local and global pheromone update mechanism.

The genetic algorithm presented in [10] and the ant algorithm described in [15] are used to compare the performance of the proposed DCACS algorithm. The ant algorithm in [15] has been applied to solve Steiner tree problems. However, their algorithm differs from ours in which: 1) Their algorithm bases on the ant system (AS), but ours on the ant colony system (ACS). 2) The ants in their algorithm start from all the destination nodes and merge together when an ant steps on the route that another ant has passed. Our ants start from a randomly chosen destination node, and each ant builds their own tree. 3) Their algorithm has conflict detection and avoidance mechanism as their ants may conflict with each other resulting in erroneous results. But ours do not need to do so. 4) The pheromone update methods are different between the two algorithms.

The other contribution of this paper is that it analyzes the results of giving priority in the selection of destination nodes by the heuristics. Three types of heuristic methods are tested. Proper heuristic methods and parameters settings to reinforce pheromone values to the destination nodes and the other nodes are concluded by experiments on the Steiner cases from the OR-Library [14].

The paper is organized as follows. Section 2 introduces the description of multicast routing problems and their mathematical definition. The construction method of the distance complete graph (DCG) is also introduced. Section 3 describes the implementation and main techniques of the proposed DCACS algorithm. In Section 4, the proposed algorithm is tested by some cases to analyze its performance. The conclusion of the paper is made in Section 5.

2 Multicast Routing Problem and Distance Complete Graph

In this section, the formal description and definition of the multicast routing problem will be made and the distance complete graph topology on which the proposed algorithm based will also be described.

2.1 Definition of the Multicast Routing Problem

A network graph is denoted as $G = (V_G, E_G, \Omega)$, where V_G is the set of nodes in the network, E_G is the set of edges that connect two nodes in V_G, and Ω is the set of constraints. If the problem is constraint-free, then $\Omega = \emptyset$. In a multicast routing problem, the V_G is divided into three subsets as V_S, V_T, V_I, where V_S is the set of source nodes, V_T is the set of destinations, and V_I is the set of intermediate nodes. Each edge e which belongs to V_G has a positive cost $c(e)$ for passing the edge. If there is no edge connecting two nodes in V_G, the corresponding cost will be set as ∞, which indicates that the edge does not exist. The multicast routing problem is to find a tree with the minimum cost connecting the nodes in V_S and V_T , and the resulting tree must comply with the constraints in Ω. The formal definition of a multicast routing problem is to find a tree $T = (V_S + V_T + V_\theta, E_\theta)$, where $V_\theta \subseteq V_I$ and $E_\theta \subseteq E_G$, with

$$\min \textstyle\sum_{e \in T} c(e), \ T \text{ satisfies constraints in } \Omega.$$

Fig. 1 shows an example of the multicast routing. There are twelve nodes and some edges. The gray solid node 1 is the source node, and the black solid nodes 2, 6, and 11 are the destination nodes. Now the problem is to find a tree which has all the destination nodes connected to the source node with the minimum total cost within the constraints. The black edges between nodes 2, 1, 5, 6, 7, 11 construct the result tree in the example via the intermediate nodes 5 and 7. As the problem considered in this paper is the one without constraints, which is an equivalence of the Steiner tree in graphs, we do not differentiate the source nodes and the destination nodes. We call both of the nodes as destination nodes. The sets V_S and V_T are combined to be denoted as V_D.

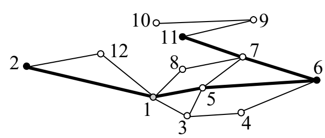

Fig. 1. An example of a multicast routing network

2.2 Distance Complete Graph

The network in multicast routing is not a fully-connected graph, which means that some nodes are not directly connected by a single edge, though they are attainable via other nodes and the corresponding edges. A distance complete graph (DCG) of a network is the one that each pair of nodes is logically connected and the cost (or distance) on that edge is the cost of the shortest path between the nodes in the actual network.

One of the classical algorithms for generating an undirected DCG is the Floyd algorithm [16]. Fig. 2 shows an example of a topology with six nodes. The solid edges in Fig. 2 are the actual edges existing between the two nodes, and the numbers beside the edges are the corresponding cost values. We can see that

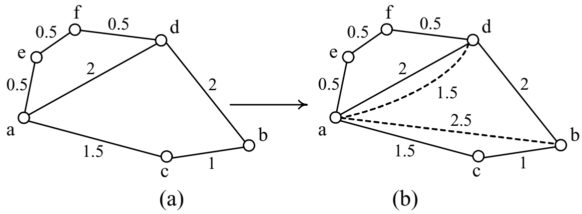

Fig. 2. An example of transforming edges (a,b) and (a,d) in the DCG

nodes a and b are not adjacent, but they can reach each other via node c or d. The algorithm is to change the original graph in Fig. 2(a) to the DCG in Fig. 2(b), where a logical edge (shown as a dashed line) is added with the minimum cost from a to b. The shortest route from a to b is via node c, so the cost $d_{\mathtt{ab}}$ of the logical edge in the DCG is 2.5. Because the graph is undirected, we have $d_{\mathtt{ba}} = d_{\mathtt{ab}}$. To nodes a and d, although they are connected directly, there is a shorter route via nodes e and f. So $d_{\mathtt{ad}} = d_{\mathtt{da}} = 1.5$.

The time complexity of the Floyd algorithm is $O(|V_G|^3)$. The ant algorithm proposed in this paper is based on the DCG, so the construction of the DCG must be preprocessed. After constructing the DCG, the ants can be dispatched to construct multicast trees.

3 Distance Complete ACS for Multicast Routing Problem

The structure of the proposed distance complete ACS (DCACS) is based on the procedure of constructing a minimum spanning tree (MST) by the Prim's algorithm [13]. The following section will present how to solve a multicast routing problem.

3.1 The Prim's Algorithm and the Pheromone and Heuristics Mechanism

Suppose S is a set with one node and V is the set of nodes in an undirected connected graph excluding the node in S. The classical Prim's algorithm [13] is to add the node j to S, provided that the edge (i, j) has the minimum cost, $i \in S$, and $j \in V - S$. The algorithm terminates when $S = V$. In the proposed algorithm, the selection criteria for the next node are not simply based on the cost of the edges, but the product of the pheromone value and the heuristic value.

The product of pheromone and heuristics for an edge (i, j) is denoted as $\tau(i, j)\eta(i, j)$, where $\tau(i, j)$ is the pheromone value and $\eta(i, j)$ is the heuristic value. The initial pheromone value τ_0 for every edge is discussed in 3.2. Three types of heuristic values are tested in this paper.

$$H_a : \eta(i, j) = \begin{cases} [\mu_D/(1/d_{ij})]^{\beta}, \text{ if } j \in V_D \\ [\mu_I/(1/d_{ij})]^{\beta}, \text{ otherwise} \end{cases} \tag{1}$$

$$H_b : \eta(i,j) = \begin{cases} \mu_D/(1/d_{ij})^\beta, \text{ if } j \in V_D \\ \mu_I/(1/d_{ij})^\beta, \text{ otherwise} \end{cases} \tag{2}$$

$$H_c : \eta(i,j) = \begin{cases} (\mu_D)^\beta/(1/d_{ij}), \text{ if } j \in V_D \\ (\mu_I)^\beta/(1/d_{ij}), \text{ otherwise} \end{cases} \tag{3}$$

where μ_D and μ_I are the reinforcement proportion to the destinations and intermediate nodes respectively. $\mu_D = \max(|V_I|/|V_G|, |V_D|/|V_G|)$. $\mu_I = \min(|V_I|/|V_G|, |V_D|/|V_G|)$. d_{ij} is the cost of the logical edge in the DCG between nodes i and j. The parameter β is a positive constant.

The destination nodes are more likely to be selected by using the above heuristic methods. With the same cost by the edges, the greedy operation prefers the nodes in V_D. The performance of the algorithm with different heuristic methods will be tested in Section 4 with a discussion on the design of the heuristics.

3.2 The Ant's Search Behavior

In this part, the process on how an ant searches for routes to build a tree is described step by step. It is a probabilistic Prim's algorithm in the framework of ACS [12] and the algorithm operates in the DCG.

Step 1: Initialization

At first, an ant k is placed on a randomly chosen destination node i. Then it begins its search.

Step 2: Exploration or Exploitation

The ant probabilistically chooses to do exploration or exploitation to select the next node j based on the state transition rule [12], which is controlled by (4)

$$j = \begin{cases} \arg\max_{r \in \Theta_k} \{\tau(i,r)\eta(i,r)\}, \text{ if } q \le q_0 \text{ (exploitation)} \\ J, \text{ otherwise (exploration)} \end{cases} \tag{4}$$

where i is the node that the ant has visited, Θ_k is the set of nodes that haven't been visited by the current ant k, q_0 is a predefined parameter in [0,1] controlling the proportion of doing exploitation and exploration, J is a random node chosen by (5). $\tau(i,r)$ stands for the pheromone value on the edge connecting nodes i and r. $\eta(i,r)$ is the heuristic value for choosing node r from node i. Note that the DCG is constructed, so any pairs of nodes are logically connected. If a random number q is smaller than or equal to q_0, the ant will choose the next unvisited node with the maximum product of pheromone and heuristic value. This is called the exploitation step. Otherwise, the next node j is chosen by (5) with a probability distribution, which is called the exploration step.

$$p_k(i,j) = \begin{cases} \frac{\tau(i,j)\eta(i,j)}{\sum_{r \in \Theta_k} \tau(i,r)\eta(i,r)}, \text{ if } j \in \Theta_k \\ 0, \text{ otherwise} \end{cases} \tag{5}$$

Step 3: Edge Extension and Local Pheromone Update

When an ant is building a solution, the pheromone value on the visited edges is decreased. It is done for the avoiding too many ants that tail after the same route and letting the other ants search the network diversely. The local pheromone update is done in two sub-steps, one is the update of pheromone on logical edges, and the other is on actual edges. The pheromone value on an edge (i, j) is updated as (6).

$$\tau(i, j) = (1 - \rho)\tau(i, j) + \rho\tau_{\min} \tag{6}$$

where $\rho \in (0, 1]$ is the pheromone evaporation rate. $\tau_{\min}$ is the lower boundary value of the pheromone on every edge. Since $\tau(i, j) \geq \tau_{\min}$, the updated pheromone value is smaller than or equal to the original one.

After choosing the next node j, an ant k moves from node i to node j. Since the ant moves on the DCG, the logical edge (i, j) can be extended to the actual route. The logical and actual edges in the route have their pheromone updated by (6). Fig. 3 shows an example of updating an edge (1, 6), which means that the ant just moved from node 1 to node 6. The logical edge (1, 6) can be extended to be a route 1, 2, 3, 4, 5, 6. Then the pheromone values on the corresponding edges (1, 2), (1, 3), (1, 4), (1, 5), (1, 6), (2, 3), (2, 4), (2, 5), (2, 6), (3, 4), (3, 5), (3, 6), (4, 5), (4, 6), (5, 6) and their symmetric edges (e.g. (2, 1)) are also updated. After the local pheromone update, the pheromone density on the edges corresponding to the logical edges decreases.

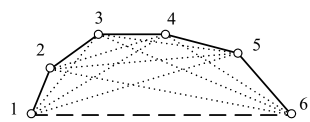

Fig. 3. Example of extending a logical edge (1, 6)

Step 4: Has the ant finished the job?

The nodes walked by the ant and the ones that are extended from the logical edges are regarded as having been visited. The termination condition for an ant is whether it has visited all the destination nodes. If the termination condition is unsatisfied, go to Step 2 for a further search. Otherwise, the ant has finished searching for a multicast tree.

3.3 Redundancy Trimming

After an ant has finished building a multicast tree connecting the source node and all the destination nodes, the tree must be checked for redundancy.

Firstly, we apply the classical Prim's algorithm to the visited nodes. If the cost of the generated MST is smaller than that of the tree built by the ant, the ant's solution is replaced by the MST.

Secondly, we check for useless intermediate nodes. The intermediate nodes that have only one degree of connectivity are deleted (e.g. node 10 will be deleted in Fig. 1).

The above two processes are repeated until the tree cannot be optimized any more. The redundancy trimming can also be used as a deterministic algorithm for multicast routing problems. Take the whole set of nodes in V_G as the input, the tree can be gradually trimmed and reduced. The resulting tree is a MST, and it is a feasible solution for the problem. However, the experiments in Section 4 show that this method can not generate the best tree. It is a sub-optimal result, which can be used to initialize the pheromone value. Hence, the initial pheromone value on every edge is set as

$$\tau_0 = 1/(|V_G| T_s) \tag{7}$$

where T_s is the cost of the tree generated by the deterministic redundancy trimming method.

3.4 Global Pheromone Update

After the m ants have performed the tree construction, the global pheromone update is applied to the best tree ever been found in order to reinforce the pheromone values. The pheromone values on the actual edges of the best-so-far tree are updated as (8).

$$\tau(i,j) = (1-\rho)\tau(i,j) + \rho \Delta\tau \tag{8}$$

where $\Delta\tau = 1/T_{\text{best}}$, T_{best} is the total cost of the best-so-far tree.

The pheromone values on the logical edges in the best-so-far tree are also updated. Suppose an actual route between a pair of nodes i and j in the best tree is $(a_0, a_1, a_2, ..., a_\psi)$, where $a_0 = i$ and $a_\psi = j$. ψ is the number of edges in the route, then the new pheromone value on the edge (i, j) equals to

$$\tau(i,j) = \sum_{l=0}^{\psi-1} \tau(a_l, a_{l+1})/\psi \tag{9}$$

4 Experiment and Discussions

The test cases used in this paper are the Steiner problems in group b from the OR-Library [14]. The eighteen problems are tabulated in Table 1, with the graph size from 50 to 100. In the table, $|V_G|$ stands for the number of nodes, and $|V_D|$ is the number of destination nodes. $|E_G|$ is the number of actual edges in the graph. OPT is the optimum of the problem. T_s is the cost of the tree generated by using only the redundancy trimming method. It can be seen that the value of T_s is not good compared to the best value.

The values of the parameters are set empirically as $q_0 = 0.9$, $\rho = 0.1$, $\tau_{\text{min}} = \tau_0$. The maximum iteration number is set as 500, but once the ants have found the optimum, the algorithm terminates. The same combination of parameters values are tested for ten times independently.

Table 1. Eighteen Steiner b test cases

No.	$\|V_G\|$	E_G	V_D	OPT	T_s	No.	$\|V_G\|$	E_G	V_D	OPT	T_s	No.	$\|V_G\|$	E_G	V_D	OPT	T_s
1	50	63	9	82	89	7	75	94	13	111	112	13	100	125	17	165	189
2	50	63	13	83	96	8	75	94	19	104	111	14	100	125	25	235	243
3	50	63	25	138	144	9	75	94	38	220	227	15	100	125	50	318	333
4	50	100	9	59	63	10	75	150	13	86	95	16	100	200	17	127	171
5	50	100	13	61	68	11	75	150	19	88	103	17	100	200	25	131	144
6	50	100	25	122	130	12	75	150	38	174	186	18	100	200	50	218	227

Table 2. Comparisons of DCACS with the GA in [10] and the ant algorithm in [15]

No.	GA[10]	ant[15]	DCACS	No.	GA[10]	ant[15]	DCACS	No.	GA[10]	ant[15]	DCACS
1	82	82	82	7	111	–	111	13	165	167.3	168
2	83	83	83	8	104	104	104	14	235	235.3	235
3	138	138	138	9	220	220	220	15	318	318	318
4	59	59	59	10	86	87.4	86	16	127	133	127
5	61	61	61	11	88	89	88	17	131	–	131
6	122	–	122	12	174	–	174	18	218	225.5	218

– The values are not available in [15].

4.1 Analysis on the Heuristic Method

The basic structures of H_a, H_b, and H_c are the same, except for the role of the parameter β. The reinforcement proportion to the destination nodes and the intermediate nodes is normalized to be in [0, 1] and $\mu_D + \mu_I = 1$. In the traditional ACS for the traveling salesman problem by Dorigo and Gambardella [12], the heuristics is defined as $(1/c_{ij})^{\beta}$, and the β ($\beta > 0$) is used to determine the relative importance of pheromone versus cost. In our algorithms in solving multicast routing problems, a series of tests is made to judge in which way the parameter β should be used and what its value should be.

The results by the three H_a, H_b, and H_c are compared in Fig. 4. Each subfigure is composed of three parts with the corresponding heuristic methods, and each method has been tested using 10, 50 and 100 ants. The figures illustrate the success rates of the algorithm to the corresponding Steiner tree problems. It can be seen that H_a and H_c are more robust than H_b, which means that the emphasis by β to μ_D and μ_I is more significant than to the cost of the edges. The best β for $H2_a$ is 1, 2, 3 and for $H2_c$ is 1, 2, 3, 4 in most cases. The best value of μ^{β} is shown to be from 8 to 16 in the test cases.

4.2 Comparison with Other Algorithms

We use the genetic algorithm in [10] and the ant algorithm in [15] to compare the performance of the proposed algorithm. The parameters used in the proposed DCACS with $H2_c$ are set as $m = 10 \sim 100$, $\beta = 3$, $\rho = 0.1$, and $q_0 = 0.9$.

The obtained results are tabulated and compared in Table 2. The genetic algorithm presented in [10] can find the optimal tree with 100% success to all

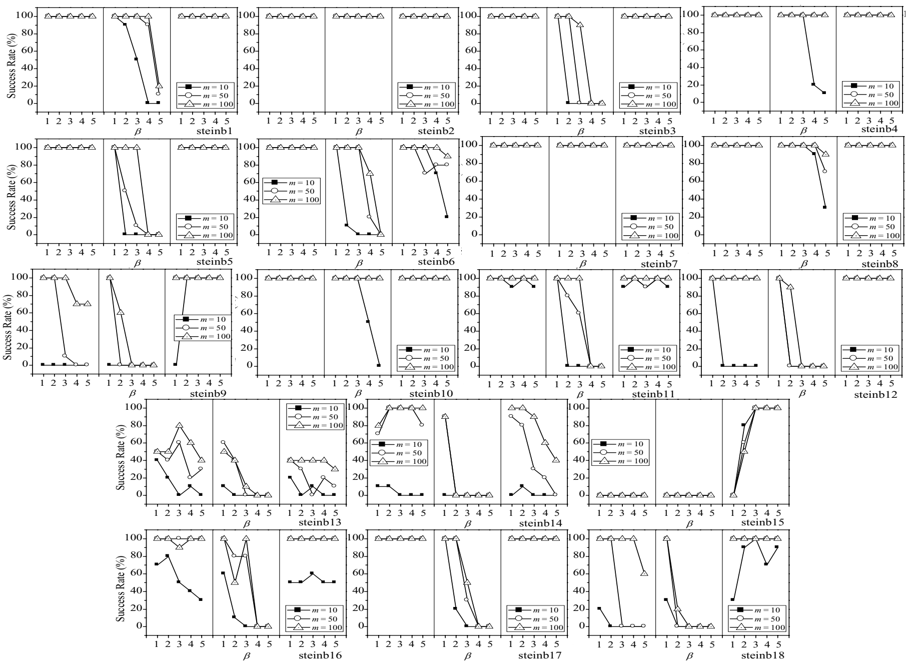

Fig. 4. Comparison between H2$_a$, H2$_b$ and H2$_c$

the eighteen test problems. Our proposed DCACS also can achieve 100% success to the test cases except for Stein b13. In [15], only the Stein b1∼b5, b8∼b11, b13∼b16, and b18 are tested. However, the results obtained by our algorithm are much better and faster according to the descriptions in [15].

5 Conclusion

This paper proposed an algorithm DCACS according to the ant colony system on the distance complete graph to solve the multicast routing problems without constraints. The ants base on the structure of Prim's algorithm and probabilistically select the nodes to construct a multicast tree. After the tree has been generated, the redundant nodes are deleted. Pheromone update and heuristic reinforcement to the destination nodes efficiently guide the algorithm to the optimum. The proposed algorithm is tested by eighteen Steiner cases and three kinds of heuristic methods are evaluated. By comparing the results with other algorithms, the proposed algorithm is very promising in solving multicast routing problems.

References

1. Tanenbaum, A.: Computer Networks. Prentice-Hall, Englewood Cliffs (1996)
2. Di Caro, G., Dorigo, M.: Antnet: Distributed Stigmergetic Control for Communications Networks. Journal of Artificial Intelligence Research 9, 317–365 (1998)

3. Gelenbe, E., Liu, P.X., Lain, J.: Genetic Algorithms for Autonomic Route Discovery. In: IEEE Workshop on Distributed Intelligent Systems: Collective Intelligence and its applications, DIS 2006, pp. 371–376 (2006)
4. Schoonderwoerd, R., Holland, O., Bruten, J., Rothkrantz, L.: Ant-like Agents for Load Balancing in Telecommunications Networks. In: Proceedings of the First International Conference on Autonomous Agents (Agents 1997), Marina del Rey, CA, pp. 209–216 (1997)
5. Gelenbe, E., Ghanwani, A., Srinivasan, V.: Improved Neural Heuristics for Multicast Routing. IEEE Journal on Selected Areas in Communications 15(2), 147–155 (1997)
6. Salama, H.F., Reeves, D.S., Viniotis, Y.: Evaluation of Multicast Routing Algorithms for Real-time Communication on High-speed Networks. IEEE Journal on Selected Areas in Communications 15(3), 332–345 (1997)
7. Wang, B., Hou, J.C.: Multicast Routing and Its Qos Extension: Problems, Algorithms, and Protocols. IEEE Network 14(1), 22–36 (2000)
8. Low, C.P., Song, X.-Y.: On Finding Feasible Solutions for the Delay Constrained Group Multicast Routing Problem. IEEE Trans. on Computers 51(5), 581–588 (2002)
9. Charikar, M., Naor, J., Schieber, B.: Resource Optimization in QoS Multicast Routing of Real-time Multimedia. IEEE/ACM Trans. on Networking 12(2), 340–348 (2004)
10. Leung, Y., Li, G., Xu, Z.B.: A Genetic Algorithm for the Multiple Destination Routing Problems. IEEE Trans. on Evolutionary Computation 2(4), 150–161 (1998)
11. Bharath-Kumar, K., Jeffe, J.M.: Routing to Multiple Destination in Computer Networks. IEEE Trans. on Communication 31(3), 343–351 (1983)
12. Dorigo, M., Gambardella, L.M.: Ant Colony System: a Cooperative Learning Approach to the Traveling Salesman Problem. IEEE Trans. on Evolutionary Computation 1(1), 53–66 (1997)
13. Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C.: Introduction to Algorithms, 2nd edn. MIT Press, Cambridge (2001)
14. Beasley, J.E.: OR-Library: Distributing Test Problems by Electronic Mail. J. Opl. Res. Soc. 41(11), 1069–1072 (1990)
15. Singh, G., Das, S., Gosavi, S., Pujar, S.: Ant Colony Algorithms for Steiner Trees: an Application to Routing in Sensor Networks. In: Recent Developments in Biologically Inspired Computing, pp. 181–206. Idea Group Publishing (2005)
16. Floyd, R.W.: Shortest Path. Communications of the ACM, 345 (1962)