The Degree-Constrained Multicasting Algorithm Using Ant Algorithm*

Ying LIU* Jianping WU Ke Xu Mingwei Xu

Dept. of Computer Science, Tsinghua Univ. Beijing, China 100084

*liuying@csnet1.cs.tsinghua.edu.cn

Abstract— **In this paper, we deal with obtaining multicast trees in packet-switched networks such as ATM networks, when there exist constraints on the packet replication capabilities of the individual switching nodes. This problem can be formulated as the Steiner tree problem with degree constraints on the nodes, so we call it the Degree Constrained Steiner Tree (DCST) problem. An ant heuristic algorithm is proposed to solve this DCST problem. We experimentally compare our algorithm with the previous ones. The experimental results show that the proposed approach can find optimal solution satisfying the degree constraints.**

Keywords—**multicast; Degree constraint; Steiner trees; Packet-switched networks**

I. INTRODUCTION

The importance of multimedia services such as teleconferencing, VOD (video on demand), and advertising in telecommunication networks is growing rapidly. These services usually involve a single source and a set of destinations that require point-to-multipoint connections between them. Routing these connections is generally called multicasting. To accommodate multicast services adequately, efficient routing algorithms should be offered together with the technological development of related hardware. The methods to establish a multicast route are largely classified into two categories. One is to set up a point-to-point connection from the source to each destination. This method is quite simple, but may spend network resources excessively. The other is to make use of multicast trees, which span the source and all destinations. By using multicast trees, it is possible to avoid unnecessary replications of data, hence utilize network resources more economically. However, obtaining multicast trees to satisfy a specified objective is not easy in general. When the goal of multicasting is to minimize the total tree cost defined by a single parameter (e.g. bandwidth cost) assigned to each link, the routing problem is typically formulated as the Steiner tree problem in networks [1]. To find a minimum cost Steiner tree is known to be NP-hard. Most of the applications in network are related to the (unconstrained) Steiner trees. In real situations, however, some constraints possibly exist, due to the technological restrictions, e.g. the delay constraint on the path from the source to each destination for satisfying an user QoS (Quality of Service) requirement, or the degree constraints on the nodes imposed by the restriction of the packet-copying capability of each switch, and so on. For the degree constrained case, as far as we know, the only heuristics are those that simply modify the unconstrained Steiner tree algorithms by inserting a subroutine of recalculating the shortest-path distances at each iteration while respecting the degree constraints [2].

In this paper, we focus on finding multicast trees with degree bounds on the nodes. In high-speed packet-switched networks such as ATM, the switches will suffer the packet copying restrictions driven by the necessity of achieving fast transmission of data. In low-speed networks such as the current *Internet*, though, it is also essential to keep the degree of each switch small, to enhance the network survivability, i.e. to cope with the link or node failures adequately. The packet-copying capability of each switching node can be represented as a degree constraint on that node, hence the problem of finding a multicast tree under these constraints can be modeled as a Degree-Constrained Steiner Tree problem (DCST) in networks, i.e. a Steiner tree problem with the degree bounds on the individual nodes. More formal description of the DCST is as follows.

The packet-switched computer network is modeled as a simple, undirected graph $G = (V, E)$ where V is a set of nodes and E is a set of edges. The cost of edge $(u, v) \in E$ is denoted by $c(u, v)$, which is a positive real number. The cost may measure utilization cost or the degree of congestion. The degree-constraint of each node i is denoted by k_i.

The DCST is defined as follows:

Given: A simple, undirected, connected graph $G = (V, E)$, $s \in V$ denotes the source node and D denotes the set of destination nodes (for convenience, the source and destination nodes are all referred to as destination nodes in the discussion to follow), and each

* Supported by the National Science Foundation of China under Grant Num. 90104002, 69972036, and 69725003.

0-7803-7661-7/03/$17.00©2003 IEEE.

non-leaf node degree constraints $k_i \geq 2$.

Find: A multicasting tree such that the degree of each node $d_i \leq k_i$ and its total cost is minimum among all possible choices of trees satisfying the degree-constraints.

The DCST is of course, NP-hard, since the Steiner tree problem is NP-hard. As mentioned before, however, few researches [2] have been made for the DCST, when compared with its unconstrained version. The work of Bauer and Varma [2] was focused on the practical or experimental aspects of the DCST. They proposed several heuristics which modified the (unconstrained) Steiner tree algorithms by simply inserting a subroutine that recalculates the shortest paths at every iteration, after eliminating the already degree constrained nodes and their incident links from the network. They presented the simulation results from the sparse networks with 200 nodes and reported that most of the modified heuristics performed very satisfactorily. In this paper, we use ant algorithm to solve the DCST. The ant heuristic is a new general-purpose heuristic algorithm which can be used to solve different combinatorial optimization problems. The remainder of this paper is organized as follows. In Section II, we introduce the ant algorithm. In Section III, we describe our heuristics. Experimental results from various environments are shown in Section IV. Finally Section V summarizes the results and concludes the paper.

II. BACKGOUND

A. Ant Algorithm

The natural metaphor on which ant algorithms are based is that of ant colonies. Real ants are capable of finding the shortest path from a food source to their nest [3], without using visual cues by exploiting pheromone information. While walking, ants deposit pheromone on the ground and follow, in probability, pheromone previously deposited by other ants. In Fig.1, we show a way ants exploit pheromone to find a shortest path between two points. Consider Fig.1 (a): ants arrive at a decision point in which they have to decide whether to turn left or right. Since they have no clue about which is the best choice, they choose randomly. It can be expected that, on average, half of the ants decide to turn left and the other half to turn right. This happens both to ants moving from left to right (those whose name begins with an L) and to those moving from right to left (name begins with an R). Fig. 1(b) and (c) shows what happens in the immediately following instants, supposing that all ants walk at approximately the same speed. The number of dashed lines is roughly proportional to the amount of pheromone that the ants have deposited on the ground. Since the lower path is shorter than the upper one, more ants will visit it on average, and therefore pheromone accumulates faster. After a short transitory period the difference in the amount of pheromone on the two paths is sufficiently large so as to influence the decision of new ants coming into the system [this is shown by Fig. 1(d)]. From now on, new ants will prefer in probability to choose the lower path, since at the decision point they perceive a greater amount of pheromone on the lower path. This in turn increases, with a positive feedback effect, the number of ants choosing the lower, and shorter, path. Very soon all ants will be using the shorter path. The above behavior of real ants has inspired *ant system*, an algorithm in which a set of artificial ants cooperate to the solution of a problem by exchanging information via pheromone deposited on graph edges. The ant system has been applied to combinatorial optimization problems such as the traveling salesman problem (TSP) [4], [5].

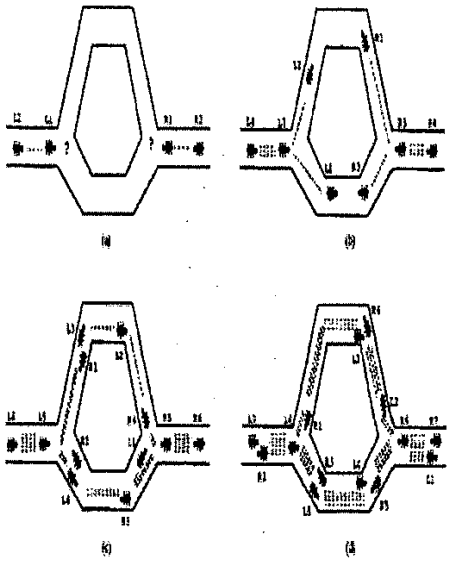

Fig. 1. How real ants find a shortest path. (a) Ants arrive at a decision point. (b) Some ants choose the upper path and some the lower path. The choice is random. (c) Since ants move at approximately a constant speed, the ants which choose the lower, shorter, path reach the opposite decision point faster than those which choose the upper, longer, path. (d) Pheromone accumulates at a higher rate on the shorter path. The number of dashed lines is approximately proportional to the amount of pheromone deposited by ants.

B. Ant algorithm in networks

Based on the principle above mentioned, a network model has been implemented by creating artificial ant population. To associate communication networks with the theory of ant algorithm, we established a table of probabilities, called pheromone table (as shown in Fig.2), in which pheromone strengths are represented by probabilities. Every node has an entry (row in the pheromone table) for each possible destination in the network, and every table has an entry for each neighboring node. Pheromone tables give the probabilities of alternative choices between neighboring links. They are updated based

on the rules defined. Equation (1) gives the probability with which ant k in node u chooses to move to the node v.

$$p_k(u,v)=\frac{[\tau(u,v)]\cdot[\theta(u,v)]^{\mu}}{\sum_{n\in J(u)}[\tau(u,n)]\cdot[\theta(u,v)]^{\mu}} \quad (1)$$

where τ is the pheromone, $J(u)$ is a set of all the neighboring node of node u, $\theta=\frac{1}{c(u,v)}$ is the inverse of the cost $c(u,v)$. μ is the parameter which determines the relative importance of pheromone versus cost . In (1) we multiply the pheromone on link (u,v) by a heuristic value $\theta(u,v)$. In this way, we favor the choice of outgoing links that have less cost, as well as a greater amount of pheromone.

Destination node	Preferred Neighboring node			
	a_1	a_2	...	a_m
d_1	P(1,1)	P(1,2)	...	P(1,m)
d_2	P(2,1)	P(2,2)	...	P(2,m)
⋮	...	...	...	⋮
d_n	P(n,1)	P(n,2)	...	P(n,m)

Fig2 the pheromone table

Informally, ant system works as follows. Each ant generates a complete tour by choosing the nodes according to a probabilistic state transition rule; ant prefer to move to nodes where are connected by links which have less cost with a high amount of pheromone updating rule (global updating rule, for short) is applied; then each ant deposits an amount of pheromone on edges which belong to its tour in proportion to how short its tour was. The process is then iterated. We give the rules used in our algorithm.

①When ant traversed a link, its pheromone level is changed by applying the local updating rule as follows:

$$\tau(u,v)\leftarrow(1-\alpha)\tau(u,v)+\Delta\tau(u,v) \quad (2)$$

where $\Delta\tau(u,v)=1/c(u,v)$, $0<\alpha<1$ is a pheromone decay parameter. The effect of the local updating is to make the desirability of edges change dynamically. Every time an ant uses an edge this become slightly less desirable (since it loses some of its pheromone). In this way ants will make a better use of its pheromone information.

② Once all of ants have built their tours, assuming that the path ant traveling is AP, pheromone is updated on all edges according to:

If the nodes in the path AP satisfy their degree constraints, and $(u,v)\in AP$:

$$\tau(u,v)\leftarrow(1-\alpha)\tau(u,v)+\sum_{k=1}^{m}\Delta\tau_k(u,v) \quad (3)$$

otherwise: $\tau(u,v)\leftarrow(1-\alpha)\tau(u,v)$ (4)

where: $\Delta\tau_k(u,v)=\frac{1}{1+\cos t(AP)}$ (5)

$0<\alpha<1$ is a pheromone decay parameter. $\cos t(AP)$ is the total cost of the path AP ant k traveling and m is the number of the ants.

III. ANT ALGORITHM FOR THE DCST

A. *Basic idea of the algorithm*

The algorithm is to find a multicast tree satisfying the degree constraints by launching ant in the source node.The algorithm assumes that every node knows its own degree constraint, and every node records its current degree in the established tree. Each node has a pheromone table, for simplicity, we denote the pheromone table by *Phertable*[•] . The *Phertable* of each node in the network is first initialized using an initialized pheromone value τ_0 by equation (1). In order to avoid the loop, every ant record the node which it traversed using an array *ant.taboo* in the ant-packet.

To establish the connections with all the destinations, the algorithm first randomly select a destination node in multicast destination set, and launch a number of ants to find the optimal route. When the ants arrive at the destination node, if the nodes in the route satisfy the degree constraints, we update the globally pheromone value on the path ants selected and add the route to the established tree.

B. *The details of the algorithm*

Step1: a destination d_1 is randomly selected from destination set D . Before launching ant from source node s, *ant.taboo* is designated as $\{s\}$. Ant moves from source to the next node that has maximum probability in *Phertable*[s].

Step2: if an ant be at node u, when $u=d_1$ we send ant-packet back to source node s along the path ant-packet coming, while we update the pheromone value of all the links on this path ant traveling by equation (3). When $u\neq d_1$ ant will move to the next node $v\in J(u)$ by comparing *Phertable*[u] in the node u; and if v belong to *ant.taboo*, ant dies, while the pheromone value on link is updated by

setting $\Delta\tau(u,v)=0$ in equation (2). When the ant arrives at node v, pheromone value on link (u,v) is updated by equation (2). The *ant.taboo* is updated by *ant.taboo* $\leftarrow$ {*ant.taboo*, v}.

Step3: we update each entry in the *Phertable*[u] of each node u, and launching the next ant, repeat step 1 and step 2 until all the ants with the address of destination d_1 are return to source. Then we select the path $p(s,d_1)$, along which the probability in *Phertable*[$\bullet$] of each node is maximum and every node in the path satisfy its degree constraint, establish a connection between source node and destination node. After this, every node updates its current degree in the established tree as follows: If a node is added to the tree, its current degree adds two; if the node has already been in the tree, a branch is added from it, the degree of this node adds one. Letting a set including all the nodes on the path $p(s,d_1)$ as M1.

Step 4: in set $D-d_1$, we randomly reselect the destination node d_2. When we launch each ant-packet with the address of destination node d_2 in the source node s. The ant moves to the next node v_1 by *Phertable*[s] we take two status into account as follows: i.e., $v_1 \in ant.taboo$ and $v_1 \notin ant.taboo$.

Case1: when $v_1 \notin ant.taboo$, the pheromone value on link (s,v_1) is updated locally by (2), and ant-packet moves to the next node v_2 by *Phertable*[v_1] in node v_1; if $v_2 \in ant.taboo$, ant dies; otherwise repeat the process above. From now on, each time ant encounters the node $u \in ant.taboo$, ant dies, otherwise continue.

Case2 : If $v_1 \in ant.taboo$: for ease of the description letting the node t be the first node subject to $t \notin ant.taboo$. Then repeat step (1-3).

Step 5: if all of the destinations are connected in the multicast tree, the connection process is completed, otherwise repeat step4.

The algorithm finally can find the optimal multicast trees satisfying the degree constraints. By using the *ant.taboo*, the algorithm can also avoid loop. The algorithm is simple and easy to realize.

IV. SIMULATION

We implemented the ant algorithm of the previous section. The network graph used in the experiment is constructed using the approach given in [6]. This method first creates a list of nodes (at random location) and then creates links between these nodes. The probability of a link to exist between two nodes u and v is given by:

$$P(u,v) = \beta \exp(-l(u,v)/L\alpha)$$

where $l(u,v)$ is the distance between u and v, and L is the maximum distance between any two nodes, α and β are parameters between (0,1). A large value α increases the ratio of the number of long links to short links, and a large β increases the average node degree of the networks. The simulation networks were always connected. In the simulation, the link cost was set to the distance between two nodes multiplied a random number. The degree constraint of each node is assigned to be equal to or smaller than its real degree value in the graph.

In order to evaluate the robustness of our heuristic SPH for the degree-constrained multicasting tree problem proposed in [2] was compared with the proposed ant algorithm. We perform two different simulations by varying the multicast group size and the network size. In the simulation, the average multicast tree cost was used as a measure of performance. We generated twenty different networks and calculate the average tree cost.

Fig.3 shows the comparison of the results obtained by the proposed algorithm and SPH in terms of tree cost and the network size. In the simulation, the destination number is set to be 8% of the number of the network nodes. The results show that the proposed algorithm performs better than SPH and with the increase of the destination node number The relationship between the tree cost and the number of destination node is shown in Fig.4. From Fig.4 we can see that the ant heuristic outperforms SPH.

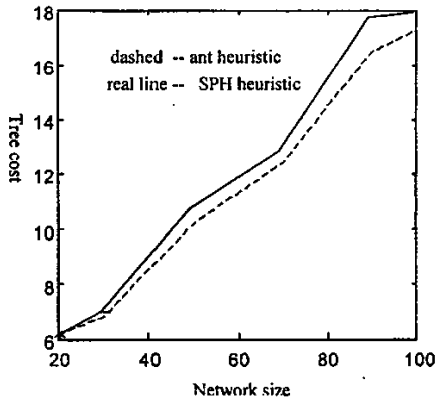

Fig3 tree cost v.s. network size

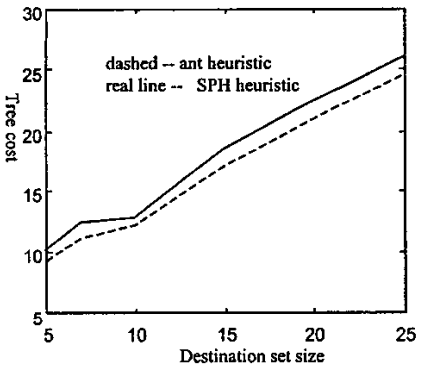

Fig4 tree cost v.s. destination set size

Finally, we compare the number of the solved instances of the two algorithms. Table 1indicate the number of solved cases from 100 instances by each heuristic, tested on the network with random degree constraints. Ant algorithm found feasible trees for most instances. On the contrary, SPH did not find feasible tree in most situations.

TABLE I PERCENTAGE OF SOLVED INSTANCES OF THE TWO ALGORITHM

	Network size	50	70	90	100
Destination size is 15% of total network size	Ant algorithm	96%	94%	96.5%	97%
	SPH	96%	93%	95%	95%
Destination size is 30% of total network size	Ant algorithm	97%	95%	97%	94%
	SPH	97%	94%	95.5%	93%

V. CONCLUSION

In this paper, we concern with the problem of finding degree-constrained multicast tree with minimum cost. An ant heuristic algorithm is used to deal with the NP-hard problem. The ant heuristics provide a novel way to solve the problem of multicasting in the networks. It has the desirable characteristics such as versatility, robustness and the ant algorithm can be outperformed by more specialized algorithms. And as far as we know, few researchers use the ant algorithm to solve the degree-constrained multicast routing problem. The experimental results show that the approach proposed can find optimal solution satisfying the degree constraints.

REFERENCES

[1] Stefan Vob, "Steiner's problem in graphs: heuristic methods", Discrete Applied Mathematics vol.40, pp. 45-72,1992.

[2] Fred Bauer and Anujan Varma, "Degree-constrained multicasting in point-to-point networks", IEEE INFOCOM, pp.369-376, 1995.

[3] Marco Dorigo, Vitttorio Maniezzo and Alberto Colorni, "Ant system: optimization by a colony of cooperating agents", IEEE Trans on Systems, Man, and Cybernetics-part B: cybernetics, vol.26, No.1, February, 1996.

[4] Marco Dorigo and Luca Maria Gambardella, "Ant colony system: a cooperative learning approach to the travelling salesman problem", IEEE Trans on Evolutionary Computation, vol.1, no.1, April, 1997.

[5] T.Stutzle and H Hoos, "Max-min ant system and local for traveling salesman problem", IEEE International Conference on Evolutionary Computation (ICEC 97), 1997, 309-314.

[6] B.M.Waxman, "Routing of multiple connections", IEEE Journal on Selected Areas in Communications, vol.6, no.9, pp.1617-1622, 1988.