Optimal State Allocation for Multicast Communications with Explicit Multicast Forwarding

De-Nian Yang, *Member*, *IEEE*, and Wanjiun Liao, *Senior Member*, *IEEE*

Abstract—In this paper, we propose a scalable and adaptive multicast forwarding mechanism based on Explicit Multicast (Xcast). This mechanism optimizes the allocation of forwarding states in routers and can be used to improve the scalability of traditional IP multicast and Source-Specific Multicast. Compared with previous work, our mechanism needs fewer routers in a multicast tree to store forwarding states and therefore leads to a more balanced distribution of forwarding states among routers. We focus on two problems and formulate each of them as an optimization problem. The first problem, referred to as MINSTATE, minimizes the total number of routers that store forwarding states in a multicast tree. The second problem, referred to as BALANCESTATE, minimizes the maximum number of forwarding states stored in a router for all multicast groups, which is proved to be an NP-hard problem. We design a distributed algorithm that obtains the optimal solution to the first problem and propose an approximation algorithm for the second problem. We also prove that the approach adopted by most existing works to allocate forwarding states in the branching routers of a multicast tree is a special case of our mechanism. The simulation results show that the forwarding state allocation provided by previous work is concentrated on the backbone routers in the Internet, which may cause the scalability problem. In contrast, our mechanism can balance forwarding states stored among routers and reduce the number of routers that store the forwarding states for a multicast tree.

Index Terms—Explicit multicast, forwarding state, scalability.

1 Introduction

MULTICAST is an efficient way of realizing one-to-many and many-to-many communications [1]. Traditional IP multicast is provided with the host group model [2] and multicast routing protocols [3], [4], [5], [6]. Each multicast group is associated with a class-D IP address, which serves as the destination addresses of data packets. Multicast addresses are assigned in a way that guarantees the global uniqueness of each class-D address [7]. Unlike IP multicasting, Source-Specific Multicast (SSM) [8] treats each one-to-many connection as one multicast channel. Each multicast channel is associated with a channel identifier composed of the sender's address and a class-D address. The class-D address is assigned by the sender and is not required to be globally unique. Both SSM and IP multicast adopt the shortest path tree to deliver multicast data. The routing of a shortest path tree is the union of the shortest paths from all receivers in the group to the tree root. For SSM, the root is the sender, and the tree is a source-based tree. For IP multicast, the root is a router called the core in CBT [5] or RP in PIM-SM [6], and the tree is a shared tree. Each sender first sends data to the root via unicast, from where the data is relayed to all the receivers.

Each router in SSM or IP multicast needs to store a forwarding state for each multicast group. Each state, identified by a channel ID or a group address, specifies the adjacent routers in the tree. Multiple forwarding states cannot be aggregated into one state, because their IDs may not be contiguous, and their next-hop routers may be different. Therefore, routers may not have enough memory to store all of the multicast states when there are a larger number of multicast groups [9]. Moreover, a router may take a long time to look up the forwarding state for each arriving data. Such problem is worse in SSM than in IP multicast, because SSM may need more multicast trees. SSM uses an individual tree for each sender in a group, but IP multicast can use a single shared tree to deliver the data from all senders in the group.

Several research efforts [9], [10], [11], [12], [13], [14], [15], [16], [17], [18], [19] have attempted to reduce the number of multicast forwarding states in a router. In [9], [10], [11], [12], a single multicast tree is built to deliver data of multiple groups with similar receivers. The problem with this approach is that receivers may receive undesired data from multicast groups that they did not join. Hence, one must carefully choose the single tree so as to reduce the amount of undesired data. In [13], [14], [15], [16], [17], [18], only the branching routers of a multicast tree are assigned to store forwarding states, where the branching router is a router with at least two child routers in the tree. A multicast packet is not duplicated on the path from a branching router to its nearest downstream branching router. Thus, packets are sent via unicast between two routers, and intermediate routers on the path do not store multicast states of the tree.

So far, existing work has focused only on reducing the number of forwarding states in all routers, and no related work has discussed the distribution of forwarding states among different routers. Previous research [15] analyzes the distribution of multicast states and points out that the forwarding states will be concentrated on backbone routers once multicast services become popular, since backbone

• *The authors are with the Department of Electrical Engineering and the Graduate Institute of Communication Engineering, National Taiwan University, Taipei, Taiwan. E-mail: {dnyang, wjliao}@cc.ee.ntu.edu.tw.*

Manuscript received 1 May 2006; revised 3 Jan. 2007; accepted 20 June 2007; published online 26 July 2007.
Recommended for acceptance by P. Mohapatra.
For information on obtaining reprints of this article, please send e-mail to: tpds@computer.org, and reference IEEECS Log Number TPDS-0108-0506.
Digital Object Identifier no. 10.1109/TPDS.2007.70754.

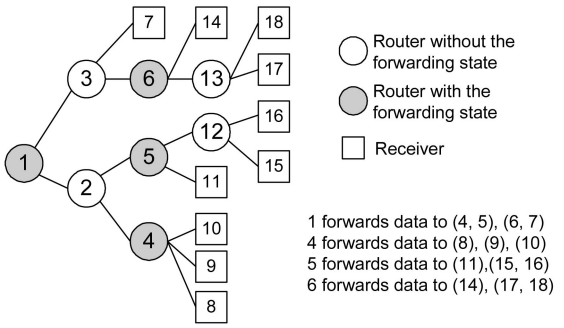


Fig. 1. An example of our mechanism.

routers are used by more multicast trees. Moreover, the authors find that the routers with more adjacent routers or hosts tend to store more forwarding states. It is reasonable, because they more likely act as branching routers of a multicast tree. Thus, the distribution of forwarding states among routers is unbalanced.

In this paper, we propose a new multicast forwarding mechanism based on Explicit Multicast (Xcast) forwarding for SSM and IP multicast. Each IP packet in Xcast can include multiple receiver addresses in the header. Upon receiving an Xcast packet, the router encapsulates multiple receiver addresses in a packet and uses an existing unicast routing protocol to find the neighboring routers to which the packet must be delivered. The router duplicates the packets that must be delivered to more than one neighboring router.

The objective of our mechanism is to optimize the distribution of forwarding states among routers. No routers in the tree need to store forwarding states for small groups (that is, with few receivers). When more receivers have joined a tree, some of the routers are dynamically chosen to store forwarding states. When the number of receivers becomes small, some of the routers discard their forwarding states. The forwarding state of each downstream interface records the addresses of a set of downstream receivers and routers that also store forwarding states for the tree. A router, on the receipt of a data packet, duplicates the packet to each downstream interface associated with a forwarding state and replaces the destination addresses in the header with the addresses of the downstream receivers or routers that also store forwarding states. The router then delivers the packet via Xcast to the downstream receivers and routers. Consider Fig. 1, for example. Node 1 is the root of the multicast tree, and nodes 1, 4, 5, and 6 store forwarding states for this tree. The nodes within a parenthesis denote the destinations of an Xcast packet. It is clear in Fig. 1 that not all branching routers in our mechanism need to store forwarding states. Therefore, our mechanism can flexibly allocate forwarding states to the routers.

In this paper, we formulate two optimization problems to effectively distribute forwarding states among routers. The first problem, referred to as MINSTATE, minimizes the number of routers that store forwarding states for each tree. We design a distributed algorithm that can find the optimal solution to MINSTATE. The second problem, referred to as BALANCESTATE, minimizes the maximum number of forwarding states stored in a router for all groups. The second problem balances the distribution of forwarding states among routers, because routers that store too many forwarding states tend to move some of the states to other routers in order to reduce the objective value of the problem. We prove that BALANCESTATE is NP-hard, and we propose an approximation algorithm and a distributed algorithm for this problem.

The proposed mechanism can cooperate with previous work to reduce the number of forwarding states for a tree. Our mechanism is orthogonal to the approach that adopts a single multicast tree to serve multiple multicast groups (for example, [9], [10], [11], and [12]), and can also be integrated with that approach to further reduce forwarding state stored in each router. Another approach that stores forwarding states only in branching routers of each tree (for example, [13], [14], [15], [16], [17], and [18]) is proved to be a special case of our mechanism, because our mechanism can more flexibly distribute forwarding states among routers. We also show that with the second approach (that is, storing states only in branching routers), forwarding states may be concentrated on backbone routers, which are branching routers in most cases, even when the group size is very small. When the number of group increases, backbone routers may not have enough memory to store forwarding states, causing a large forwarding delay to each multicast packet at each backbone router. On the contrary, our mechanism can reduce the number of forwarding states stored in a router and balance the distribution of forwarding states among routers. Thus, it allows only a portion of branching routers to store forwarding states and also allows nonbranching routers to store states. Therefore, our mechanism can increase the scalability of both SSM and IP multicast with respect to the number of members in a multicast group and the number of multicast groups in a network.

The rest of this paper is organized as follows: In Section 2, the problem description and definition are addressed. In Section 3, the algorithms for MINSTATE are presented, and the proposed algorithms are proved to be optimal. In Section 4, BALANCESTATE is proved to be NP-hard, and an approximation algorithm and a distributed algorithm are proposed. In Section 5, our mechanisms are validated by simulations. Finally, this paper is concluded in Section 6.

2 Problem Description

In this paper, the network is modeled as a connected directed graph $G(V, A)$, where V and A are the sets of vertices and arcs, respectively. Each vertex is either a host or a multicast router. Each arc is a point-to-point link, and any multiaccess link can be represented as multiple arcs. For SSM, the root is the sender. For IP multicast, the root is a relay node, like the core in CBT and RP in PIM-SM. We assume that each receiver of a group is a host connected to a Designated Router (DR). Therefore, each receiver must be a leaf node of a multicast tree, and all leaf nodes of the tree are the receivers. For each tree, a vertex m is upstream to another vertex n if m is on the path from the root to n. In this case, n is downstream to m. In this paper, vertex and node are interchangeably used.

For each multicast tree, a set of multicast routers is selected to store forwarding states for the tree. Multicast data are sent between routers via Xcast. For each multicast tree, a multicast router d that stores the forwarding state for the tree is referred to as a *state node* of the tree. The nearest state node u upstream to d is the *upstream state node* of d, and d is a *downstream state node* of u. All intermediate nodes on the path from u to d are *stateless nodes*. Each receiver is a *downstream receiver* of its upstream state node. A state node may have more than one downstream state node and downstream receiver from each interface. Consider Fig. 1 for example. Nodes 1, 4, 5, and 6 are state nodes of the multicast tree. Nodes 4 and 5 are the downstream state nodes of node 1 from the downstream interface to node 2. Nodes 15 and 16 are the downstream receivers of node 5 from the interface to node 12. Node 1 is the upstream state node of node 6.

In this paper, we formulate the distribution of forwarding states in routers as two optimization problems. The first problem MINSTATE is defined for a multicast tree t that comprises a set of nodes V_t and a set of arcs A_t. The problem is to decide whether each node in V_t should be a state node such that the total number of state nodes in tree t is minimized. To balance the distribution of forwarding states among routers, the second problem BALANCESTATE is defined for a network that comprises a set of routers V, a set of links A, and a set of multicast trees T, where two multicast trees are allowed to use some common routers in V. For each router used in each multicast tree t in T, the problem is to decide whether the router should store a forwarding state for t such that the maximum number of forwarding states stored in a router is minimized.

The two optimization problems share a common constraint, namely, a limit on the packet forwarding delay in each stateless node. For each node, the number of destination addresses in an Xcast packet sent from a downstream interface is identical to the total number of downstream state nodes and downstream receivers from the interface. Each stateless node looks up more addresses in its unicast forwarding table, since the header of an Xcast packet contains more destination addresses. For each state node, the two problems have a common constraint that the number of destination addresses in each Xcast packet cannot exceed δ. In other words, a node can have at most δ destinations from each downstream interface, where a *destination* is either a downstream state node or a downstream receiver. Therefore, we limit the packet forwarding delay in each node by the constraint. The network operators can adjust δ to find the best trade-off between the packet forwarding delay and the memory consumption according to the forwarding speed of a router.

For each multicast tree, the root is a state node. Each receiver is a stateless node, since it does not deliver data to any other node. The input parameters and decision variables are summarized in Table 1.

TABLE 1
Input Parameters and Decision Variables

Symbol	DESCRIPTION
V	the set of routers in the network
A	the set of links in the networks
T	the set of multicast trees
t	a multicast tree, $t \in T$
V_t	the set of nodes in multicast tree t
A_t	the set of arcs in multicast tree t
p_m^t	the parent node of m in multicast tree t
C_m^t	the set of child nodes of m in multicast tree t
r_t	the root of multicast tree t
R_t	the set of receivers in multicast tree t
δ	the maximum number of destinations of a state node from each downstream interface
T_m^t	a set of sub-trees of t, where each sub-tree is rooted at m
f_m^t	a sub-tree of t; sub-tree f_m^t is rooted at m and must have more than δ leaves, $f_m^t \in T_m^t$; each leaf node is a downstream receiver or a downstream router of m in t
$V_{f_m^t}$	the set of nodes of f_m^t except the leaves of f_m^t
σ_m^t	a binary variable; σ_m^t is one if node m is a state node in t; otherwise, σ_m^t is zero
u_m^t	the upstream state node of m in multicast tree t
D_m^t	the set of destinations of m from all downstream interfaces in multicast tree t
$D_{m,n}^t$	the set of destinations of m from the downstream interface to n in multicast tree t
$\phi_{\min}^t$	the minimum number of state nodes in multicast tree t; the objective value of MINSTATE
$\phi_{\max}$	the maximum number of forwarding states stored in a router, the objective value of BALANCESTATE
$\phi_{\max}^*$	the maximum number of forwarding states stored in a router in the optimal assignment of BALANCESTATE
$\sigma_m^{t,LP}$	the optimal solution of m in t in the linear relaxation on BALANCESTATE-ILP
$\phi_{\max}^{LP}$	the optimal objective value of the linear relaxation on BALANCESTATE-ILP

3 Minimizing the Number of State Nodes in Each Multicast Tree

In this section, we propose a distributed algorithm called MINSTATE-DISTRIBUTED to find the optimal solution to MINSTATE. At any instant, each state node independently decides whether it can perform two operations. The node first tries to remove its forwarding state. If it fails, then it tries moving the forwarding state to its parent node. The former operation reduces the number of state nodes, and the latter operation packs the state nodes such that more state nodes can remove forwarding states in later operations. The MINSTATE-DISTRIBUTED Algorithm stops when all state nodes are no longer able to perform the above two operations. Each state node in MINSTATE-DISTRIBUTED only stores the addresses of its upstream state node, parent node, child nodes, and destinations from each interface instead of the whole multicast tree. The algorithm does not restrict the sequence of the nodes that perform the above two operations. This merit enables the algorithm to be implemented in a distributed manner.

Fig. 2 gives the details of MINSTATE-DISTRIBUTED. Auxiliary variable x_m^t dictates whether node m in t is required to decide if it can remove or move a forwarding state. The algorithm stops when x_m^t is zero for each node m. Although this algorithm has no restriction on the sequence of nodes that perform the above two operations, the

Given: a tree $t \in T$ with node $V_t \subseteq V$ and arc $A_t \subseteq A$, δ.

Find: $\phi^t_{\min}$, and σ^t_m, $\forall m \in V_t$.

1. Let $\sigma^t_m \leftarrow 1$, $x^t_m \leftarrow 1$, $\forall m \in V_t - R_t - \{r_t\}$; let $\sigma^t_r \leftarrow 1$, $x^t_r \leftarrow 0$; let $\sigma^t_m \leftarrow 0$, $x^t_m \leftarrow 0$, $\forall m \in R_t$.
2. While $\exists m \in V_t - R_t - \{r_t\}$ such that $x^t_m = 1$,
   if $\left|D^t_{u^t_m,m}\right| + \left|D^t_m\right| - 1 \le \delta$, then let $\sigma^t_m \leftarrow 0$,
   $x^t_m \leftarrow 0$, and $x^t_n \leftarrow 1$, $\forall n \in D^t_{u^t_m,m} \cup \{u^t_m\}$;
   else if $\sigma^t_{p^t_m} = 0$ and $\left|D^t_m\right| \le \delta$, then let $\sigma^t_m \leftarrow 0$,
   $x^t_m \leftarrow 0$, $\sigma^t_{p^t_m} \leftarrow 1$, $x^t_{p^t_m} \leftarrow 1$, and $x^t_n \leftarrow 1$,
   $\forall n \in D^t_{p^t_m} \cup D^t_{u^t_{p^t_m},p^t_m} \cup \{u^t_{p^t_m}\}$;
   else let $x^t_m \leftarrow 0$;
   end if;
   end while.
3. Let $\phi^t_{\min} \leftarrow \sum_{m \in V_t} \sigma^t_m$.

Fig. 2. The distributed algorithm MINSTATE-DISTRIBUTED.

following lemmas and theorem prove that our algorithm can find the optimal solution to MINSTATE with $O(\delta d|V|)$ operations, where d is the maximal distance between a branching node and its closest upstream branching node.

Lemma 3.1. *Given an optimal assignment S^*, for any feasible assignment S that is not optimal, we can obtain another optimal assignment by a sequence of operations on S^* and S.*

Proof. We prove this lemma by introducing an algorithm with a sequence of operations on S^* and S. After the algorithm stops, S^* and S are identical, and both are optimal. For each node m in t, we denote the assignment of S^* and S at node m by σ^*_m and σ_m, respectively. The algorithm is specified as follows: First, we number all nodes from 1 to $|V_t|$ with postorder traversal. Then, for m from $|V_t|$ to 2, $m \notin R_t$, if $\sigma^*_m = 0$ and $\sigma_m = 1$, we let $\sigma_m \leftarrow 0$ and $\sigma_{p^t_m} \leftarrow 1$. If $\sigma^*_m = 1$ and $\sigma_m = 0$, we let $\sigma^*_m \leftarrow 0$ and $\sigma^*_{p^t_m} \leftarrow 1$.

The above algorithm compares nodes m in S and S^*. If m is a state node in S only, node m removes its forwarding state when its parent node is a state node. Otherwise, node m moves the forwarding state to its parent node when its parent node is stateless. At the beginning of the iteration, for all node m, we maintain both assignments as feasible solutions to the problem, and both assignments of state nodes in the subtree rooted at m must be identical, except for node m, because all nodes except m in the subtree have been compared. At the end of each iteration, the assignment S must also be feasible. The reason is that the upstream state node of m in S^* is either identical to the one of S (that is, both are p^t_m) or upstream to p^t_m. Therefore, S must be feasible to MINSTATE, since S^* is also a feasible assignment. In other words, both assignments are feasible at the end of the iteration. Because both assignments are feasible at the beginning of the algorithm, the above induction proves that the above algorithm maintains the feasibility of both assignments at the end of the algorithm. Moreover, the assignment of state nodes in S is identical to S^* after the algorithm stops. Since the number of state nodes in S^* is not changed, S^* is still an optimal assignment. Therefore, S is also an optimal assignment after the algorithm stops. □

Lemma 3.2. *An assignment is optimal if no state node can remove its forwarding state or move its forwarding state to its parent node.*

Proof. We assume that the assignment is not optimal. Since it is a feasible assignment, we can obtain an optimal assignment after a sequence of operations according to Lemma 3.1. This contradicts the requirement that no state node can remove or move its forwarding state. □

Theorem 3.3. *MINSTATE-DISTRIBUTED can find an optimal assignment with $O(\delta d|V|)$ operations, where d is the maximal distance between a branching node and its closest upstream branching node.*

Proof. According to Lemma 3.2, MINSTATE-DISTRIBUTED can find an optimal assignment after it stops, because all nodes can no longer remove or move forwarding states. Each forwarding state incurs at most one remove operation. In addition, at least one additional destination address must be added to each Xcast packet sent by a branching node if the forwarding state of the node is moved to its parent node. Therefore, each forwarding state can be moved upstream by at most δ branching nodes and incurs at most $O(\delta d)$ move operations. Since a multicast tree initially has $|V_t|$ state nodes and each forwarding state incurs at most $O(\delta d)$ operations, MINSTATE-DISTRIBUTED requires at most $O(\delta d|V|)$ operations to find the optimal solution. □

In the following, we prove that most of the previous work [13], [14], [15], [16], [17], [18] that allocate forwarding states only to the branching nodes of a multicast tree is a special case of our mechanism.

Corollary 3.4. *If δ is one, the root and branching nodes are the only state nodes in the optimal assignment.*

Proof. In the optimal assignment, if any branching node does not have a forwarding state, its upstream state node must have more than one destination from the downstream interface to the branching node. Therefore, the assignment is infeasible. If any nonbranching node has a forwarding state, we can remove the forwarding state. It implies that the assignment is not optimal. □

In addition to the above corollary, our simulation results show that we use at least 40 percent fewer state nodes than the assignment with only branching nodes storing forwarding states in a multicast tree when δ is two. Our mechanism uses fewer forwarding states when δ is more than two due to the following corollary.

Corollary 3.5. *Given two instances I_1 and I_2 with the same tree but different δ, say, δ_1 and δ_2, if δ_1 is not less than δ_2, then ϕ_1 is not bigger than ϕ_2, where ϕ_1 and ϕ_2 are the optimal objective values of I_1 and I_2.*

Proof. Since the assignment that corresponds to ϕ_2 is feasible to I_1 and the problem is a minimization problem, ϕ_2 is an upper bound on ϕ_1. Therefore, ϕ_1 is not bigger than ϕ_2. □

Finding optimal assignments with arbitrary sequence of nodes that perform the two operations is important for network protocol design, because there is no centralized coordination in real networks. Therefore, no node can obtain the optimal sequence of operations. In addition, our algorithm can still find the optimal assignment, even with dynamic group membership. When a new receiver joins a tree, it is connected to the existing tree via an on-tree state node. To find the optimal assignment, we first let all nodes between the on-tree state node and the new receiver store forwarding states. Then, we remove or move the forwarding states later. The details of our network protocol based on the algorithm can be found in [29].

The processing overhead of our approach in routers includes two parts. The first one forwards each data packet with multiple destination addresses at each state node. The overhead, inherited from Xcast, is proportional to the number of destination addresses in each packet. The second one is the overhead to find the state nodes in each multicast tree in a distributed manner. The overhead is proportional to the number of remove and move operations in our distributed algorithm, because each operation is required to exchange control messages between adjacent routers, whereas the number of required operations is $O(\delta d|V|)$. Therefore, both parts of processing overheads are proportional to δ, and the network operator can control the tradeoff between the overhead and the quality of the solution with different values of δ. In addition, our simulation results show that the performance improvement dramatically decreases as δ increases. In other words, a small δ is sufficient, and the processing overhead is therefore limited for most cases.

4 Minimizing the Maximum Number of Forwarding States Stored in a Router

Although the proposed algorithm for MINSTATE minimizes the number of state nodes in a multicast tree, the distribution of forwarding states among routers may be highly unbalanced. Since it is difficult to aggregate multicast forwarding states, some routers may not have enough memory to store all forwarding states and experience high forwarding delay for multicast packets, but some may be underutilized and able to store more forwarding states. In this section, we consider the problem of balancing forwarding states among routers, namely, BALANCESTATE. We first prove that BALANCESTATE is NP-hard, and then, we design an approximation algorithm and a distributed algorithm to solve this problem. The following lemma shows that the optimal assignment of MINSTATE is highly unbalanced in the worst case.

Lemma 4.1. *In the worst case, the maximum number of forwarding states stored in a router in the optimal assignment of MINSTATE is $|T|$ times the maximum number of forwarding states stored in a router in the optimal assignment of BALANCESTATE.*

Proof. We prove the theorem with a worst case example. In this example, each multicast tree is identical to the one in Fig. 3, and δ is two for each multicast tree. Router m is used by all multicast trees, and any other router is used by at most one multicast tree. Router m stores the forwarding states of all multicast trees in the optimal assignment of MINSTATE. In other words, the maximum number of forwarding states stored in a router in the optimal assignment of MINSTATE is $|T|$. However, router m stores the forwarding state of only one multicast tree in the optimal assignment of BALANCESTATE. Any other tree uses m's parent and child branching routers to store forwarding states. Therefore, the maximum number of forwarding states stored in a router in the optimal assignment of BALANCESTATE is one. □

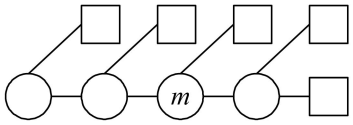


Fig. 3. An example that illustrates the unbalanced distribution of forwarding states in routers, $\delta = 2$.

The example in Fig. 3 shows the importance of BALANCESTATE. In Section 4.1, we prove that BALANCESTATE is NP-hard, and in Section 4.2, we design an approximation algorithm and a distributed algorithm.

4.1 Proof of NP-Hardness of BALANCESTATE

The following theorem proves that BALANCESTATE is NP-hard.

Theorem 4.2. *BALANCESTATE is NP-hard.*

Proof. We prove that BALANCESTATE is NP-hard by reduction from *Minimum Graph Coloring*. We first transform the graph in Minimum Graph Coloring into a network with multicast trees, where the set of routers used in more than one multicast tree is determined by the connectivity of the graph. We then prove that the maximum number of forwarding states stored in a router is one in the decision problem of BALANCESTATE, which is denoted by BALANCESTATE-DECISION, if and only if the graph in Minimum Graph Coloring is 3-colorable. We note that deciding whether a graph is 3-colorable is NP-complete [20]. We first transform the graph G_{MGC} in Minimum Graph Coloring into a network in BALANCESTATE. For each node m in G_{MGC}, we construct seven multicast trees in the network, as shown in Fig. 4a, where r, g, and b correspond to *red*, *green*, and *blue*, respectively. A router is used in two multicast trees if the two nodes in the two multicast trees are connected by dashed lines in Fig. 4a. For example, nodes y_m^r and x_m^r use the same router in the network. In addition, for each color c, where c is r, g, or b, we let node u_m^c have $g_m + 1$ child nodes, where g_m is the degree of m in G_{MGC}. Among the $g_m + 1$ child nodes, one node x_m^r has $g_m + 1$ receivers, and each of the other g_m nodes corresponds to an incident edge of m in G_{MGC}. In other words, for each edge that connects nodes m and n in G_{MGC}, we have two nodes $z_{m,n}^c$ and $z_{n,m}^c$ in T_{m1}^c and T_{n1}^c, and the two nodes use the same router in the network. Fig. 4b gives a network reduced from a graph that consists of nodes 1 and 2 connected by an edge with at most one forwarding state stored in each router. Obviously, we can obtain the network in polynomial time.

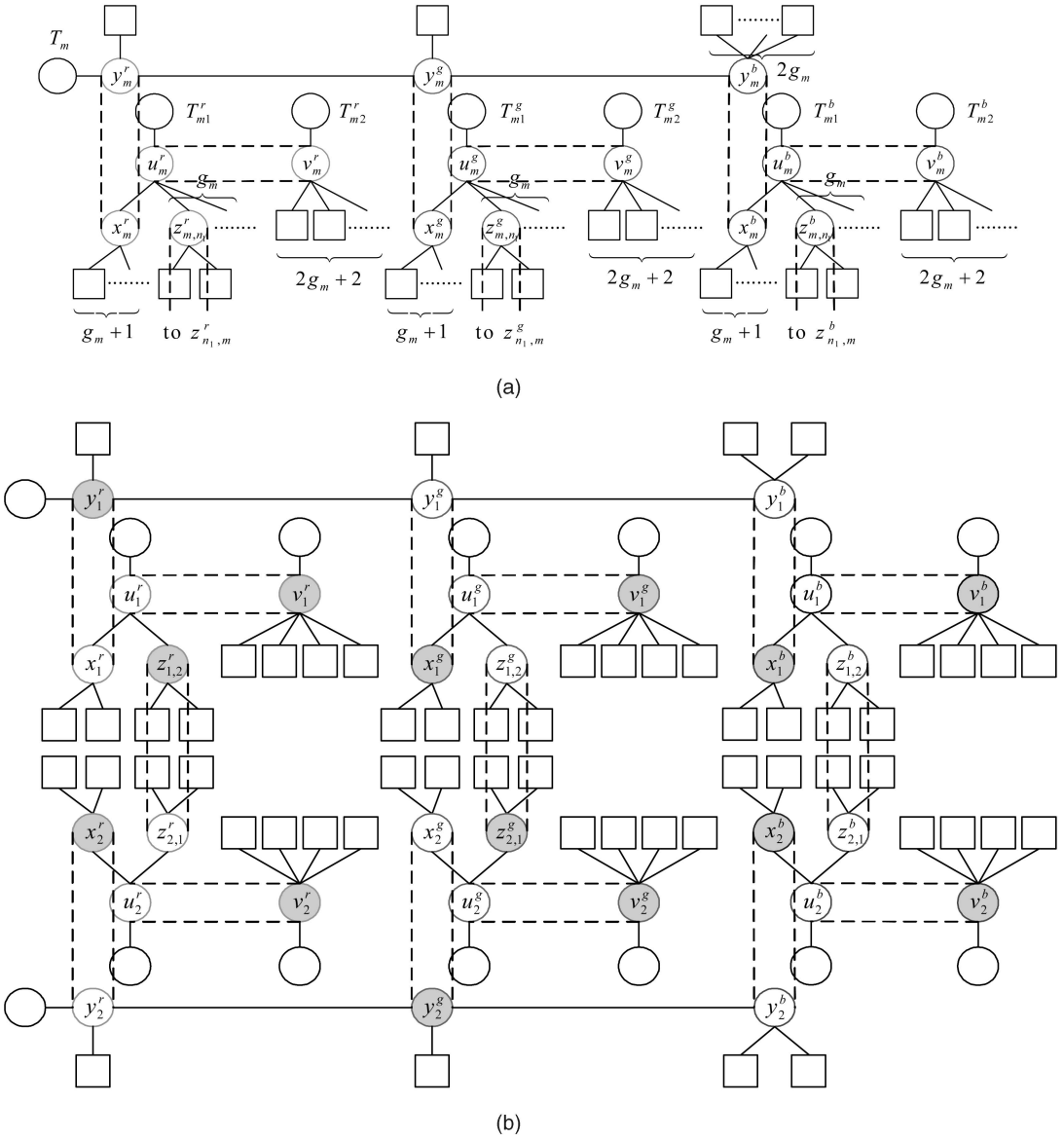


Fig. 4. The BALANCESTATE instance reduced from the instance of the graph coloring problem. Two nodes of two multicast trees are connected by dashed lines if the two nodes use the same router. (a) Multicast trees in the network that correspond to node m and m's incident edges in G_{MGC}. (b) Multicast trees in the network that correspond to two nodes connected by an edge in G_{MGC}.

We let δ be $2g_m + 1$ in BALANCESTATE. Therefore, nodes v_m^r, v_m^g, and v_m^b must act as state nodes, because each of them has more than δ child nodes. For each color c, we can allocate only one forwarding state for either x_m^c or every z_{m,n_i}^c for every feasible solution with node u_m^c as a stateless node in the network due to the following reasons ($1 \le i \le g_m$). First, each node z_{m,n_i}^c must be state nodes if x_m^c is a stateless node. Second, any node z_{m,n_i}^c can be a stateless node if x_m^c is a state node. In addition, we cannot let both x_m^c and any z_{m,n_i}^c be stateless nodes for each feasible solution with node u_m^c as a stateless node in the network, since u_m^c receives packets with more than δ destination addresses in this case. With the above observation, we can prove the sufficient and necessary conditions as follows:

We first prove the sufficient condition. Given the coloring of each node in G_{MGC}, we let u_m^c be a stateless node and v_m^c be a state node for each color c. Therefore, the router used by T_{m1}^c and T_{m2}^c stores only one forwarding state. For each node m that selects a color, say, red, we then let x_m^r be a stateless node and y_m^r and every z_{m,n_i}^r be state nodes, $1 \le i \le g_m$. Therefore, the router used by T_m and T_{m1}^r stores also one forwarding state. For the nodes corresponding to each unselected colors c, green or blue in this case, we let x_m^c be a state node and y_m^c and every z_{m,n_i}^c be stateless nodes, $1 \le i \le g_m$. Therefore, the router used by T_m and T_{m1}^c stores also one forwarding state. Fig. 4b shows an example, with nodes 1 and 2 in G_{MGC} colored with red and green. The above assignment of forwarding states is feasible according to the observations in the last paragraph. Moreover, if G_{MGC} is 3-colorable, every two nodes m and n in G_{MGC} connected by an edge can be assigned two different colors, say, r and g. In this case, only $z_{m,n}^r$ and $z_{n,m}^g$ are state nodes, and the router used by T_{m1}^c and T_{n1}^c thereby stores only one forwarding state for

each color c. Therefore, the maximum number of forwarding states stored in a router is one, and the sufficient condition follows.

We then prove the necessary condition. Considering any assignment with the maximum number of forwarding states stored in a router as one, node u_m^c must be a stateless node for each color c. For each node m in G_{MGC}, we choose any of color c for m, with x_m^c being a stateless node. Note that at least one of x_m^c is a stateless node. Otherwise, if x_m^r, x_m^g, and x_m^b are all state nodes, then y_m^r, y_m^g, y_m^b must be stateless nodes, and this assignment is not feasible for T_m. In the following, we show that each node in G_{MGC} is assigned a color different from each neighboring node. For each edge that connects m and n in G_{MGC}, because at most one of $z_{m,n}^c$ and $z_{n,m}^c$ is a state node for each color c, say, $z_{m,n}^c$ in this case, node $z_{n,m}^c$ must be a stateless node, and x_n^c cannot be a stateless node due to the observations made in the second paragraph of this proof. In other words, x_m^c and x_n^c cannot both act as stateless nodes, and nodes m and n therefore cannot select the same color. The necessary condition follows. □

4.2 δ-Approximation Algorithm

We first propose an Integer Linear Programming (ILP) formulation BALANCESTATE-ILP for the problem. We then design an approximation algorithm of the ILP formulation, where our approximation algorithm is based on rounding the optimal solution to the linear relaxation on the formulation. We limit the number of variables in a constraint such that the linear relaxation of the formulation leads to a performance bound. In the formulation, each node m of a multicast tree t is associated with a set T_m^t. Each element f_m^t of T_m^t is a subtree of t, and f_m^t is rooted at m. Each leaf node of f_m^t is a downstream receiver or a downstream router of m in t. Each subtree f_m^t in T_m^t must have more than δ leaves. In addition, each subtree f_m^t in T_m^t must be *minimal*. In other words, any subtree of f_m^t cannot be an element in T_m^t. The set T_m^t contains all possible subtrees of m that satisfy the above constraints. Let $V_{f_m^t}$ denote the set of nodes of f_m^t, except the leaves of f_m^t. Consider the multicast tree t in Fig. 1, with δ being two for example. A subtree f_2^t in T_2^t includes nodes 2, 4, 5, 11, and 12, and $V_{f_2^t}$ includes nodes 2 and 5. Another subtree $\hat{f}_2^t$, with nodes 2, 4, 5, 11, 12, 15, and 16, is not an element in T_2^t, since it is not minimal, where f_2^t is a subtree $\hat{f}_2^t$, and f_2^t is in T_2^t. We propose an algorithm to find T_m^t for each node m in each tree t later in this section.

Lemma 4.3. *For each node m in each tree t and each subtree f_m^t in T_2^t, the set $V_{f_m^t}$ has at least a state node in an assignment if and only if the assignment is feasible to BALANCESTATE.*

Proof. We first prove the sufficient condition. If an assignment is not feasible to BALANCESTATE, we show that there exists a subtree such that all nodes in the subtree are stateless. Since the assignment is infeasible, there exists a node p_m^t such that p_m^t has more than δ destinations from the downstream interface to a node m. Let $\hat{f}_m^t$ be the subtree rooted at m, and all leaves of $\hat{f}_m^t$ are the destinations of p_m^t from the downstream interface to m. Since the subtree has more than δ leaves, there must be a subtree f_m^t in T_m^t such that f_m^t is either identical to $\hat{f}_m^t$ or is a subtree of $\hat{f}_m^t$, and all nodes in $V_{f_m^t}$ are stateless.

We then prove the necessary condition. If there exists a subtree f_m^t in T_m^t such that all nodes in $V_{f_m^t}$ are stateless, node p_m^t must have more than δ destinations from the interface to m. The reason is that each leaf node of f_m^t must be a state node, a receiver, or a stateless node with at least one destination from all downstream interfaces. Therefore, the assignment is infeasible to BALANCESTATE. □

Based on Lemma 4.3, we have the formulation BALANCESTATE-ILP with the following objective function and constraints:

$$\min \phi_{\max},$$

$$\sum_{n \in V_{f_m^t}} \sigma_n^t \geq 1,\ \forall t \in T,\ \forall m \in V_t,\ \forall f_m^t \in T_m^t, \tag{1}$$

$$\sum_{t \in T: m \in V_t} \sigma_m^t \leq \phi_{\max},\ \forall m \in V, \tag{2}$$

$$\sigma_m^t = 0,\ \forall t \in T,\ \forall m \in R_t, \tag{3}$$

$$\sigma_r^t = 1,\ \forall t \in T. \tag{4}$$

For each node m in each tree t and each subtree f_m^t in T_m^t, (1) enforces that at least one node in $V_{f_m^t}$ must act as a state node. According to Lemma 4.3, any assignment is feasible to BALANCESTATE-ILP if and only if it is feasible to BALANCESTATE. Our approximation algorithm is based on rounding the optimal solution to the linear relaxation on BALANCESTATE-ILP. In this section, we explain our algorithm with the assumption that all nodes, except the leaves in each multicast tree, are branching nodes.

Our approximation algorithm is based on the following observation:

Lemma 4.4. *For each node m in each tree t and each subtree f_m^t in T_m^t, set $V_{f_m^t}$ does not have more than δ nodes if all nodes, except the leaves in tree t, are branching nodes.*

Proof. We prove the lemma by contradiction. Let node n denote a node, with every child node being a leaf of f_m^t. Let $\hat{f}_m^t$ be the subtree of f_m^t, with all leaves of n removed. If the number of nodes in $V_{f_m^t}$ is more than δ, the number of nodes in $V_{\hat{f}_m^t}$ must be equal to or larger than δ. It implies that $\hat{f}_m^t$ has at least $\delta + 1$ leaves, since all nodes in $V_{\hat{f}_m^t}$ are branching nodes. Therefore, $\hat{f}_m^t$ is also a subtree in T_m^t. Since $\hat{f}_m^t$ is a subtree of f_m^t, this contradicts that f_m^t is minimal. □

The additional notation used in the algorithm is listed as follows:

- $\sigma_m^{t,LP}$. This is the optimal solution of m in t in the linear relaxation on BALANCESTATE-ILP.
- $\phi_{\max}^{LP}$. This is the optimal objective value of the linear relaxation on BALANCESTATE-ILP.
- $\phi_{\max}^*$. This is the maximum number of forwarding states stored in a router in the optimal assignment.

Given: a set of multicast trees T, each tree $t \in T$ with node $V_t \subseteq V$ and arc $A_t \subseteq A$, δ.

Find: $\phi_{\max}$, and σ_m^t, $\forall t \in T$, $\forall m \in V_t$.

1. For each node m in each tree t,
   let $T_m^t \leftarrow \{\varnothing\}$; FINDSUBTREE($m$, t, $\{m\}$);
   end for.
2. Find $\phi_{\max}^{LP}$ and $\sigma_m^{t,LP}$, $\forall t \in T$, $\forall m \in V_t$.
3. For each node m in each tree t,
   if $\sigma_m^{t,LP} \geq 1/\delta$, then let $\sigma_m^t \leftarrow 1$; else let $\sigma_m^t \leftarrow 0$;
   end if
   end for.
4. Let $\phi_{\max} \leftarrow \max_{m \in V} \left\{ \sum_{t \in T: m \in V_t} \sigma_m^t \right\}$.

Function: FINDSUBTREE(m, t, S)

1. Let f_m^t denote the sub-tree rooted at m with $V_{f_m^t} = S$.
2. If f_m^t has more than δ leaves, then let $T_m^t \leftarrow T_m^t \cup \{f_m^t\}$;
   else
   for each leaf node n of f_m^t,
   FINDSUBTREE(m, t, $S \cup \{n\}$);
   end for;
   end if.

Fig. 5. Approximation algorithm BALANCESTATE-APX.

Fig. 5 gives our δ-approximation algorithm BALANCESTATE-APX. At Step 1, function FINDSUBTREE finds all subtrees in T_m^t for each node m. Initially, the node of a subtree f_m^t includes only m and m's child nodes. These child nodes are the leaves of f_m^t. We add f_m^t to T_m^t if the number of leaves is more than δ; otherwise, we include each child node of each leaf node in f_m^t in each of the following iterations. We stop expanding the subtree if the number of leaves is larger than δ. Then, Step 2 uses T_m^t to design BALANCESTATE-ILP and finds the optimal solution to the linear relaxation on BALANCESTATE-ILP. Step 3 simply assigns state nodes based on rounding the optimal solution obtained at Step 2. The following lemma proves that our assignment is feasible to BALANCESTATE if all nodes, except the leaves in each multicast tree, are branching nodes.

Lemma 4.5. *The assignment obtained by BALANCESTATE-APX is feasible to BALANCESTATE.*

Proof. For each node m in each tree t and each subtree f_m^t in T_m^t, set $V_{f_m^t}$ has at most δ nodes according to Lemma 4.4. With (1), there is at least one node n in $V_{f_m^t}$ such that $\sigma_n^{t,LP}$ is at least $1/\delta$. Step 3 assigns node n as a state node. Therefore, the assignment is feasible to BALANCESTATE according to Lemma 4.3. □

The following proves that the algorithm is a δ-approximation algorithm:

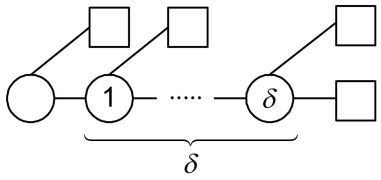


Fig. 6. An example of the multicast tree with the tight approximation ratio.

Theorem 4.6. *Each solution obtained by BALANCESTATE-APX must satisfy the following inequality:*

$$\phi_{\max} \leq \delta \times \phi_{\max}^{*}.$$

Moreover, BALANCESTATE-APX stops in polynomial time.

Proof. Let $m_{\max}$ denote the node storing the most forwarding states in BALANCESTATE-APX. In other words, $\phi_{\max} = \sum_{t \in T: m_{\max} \in V_t} \sigma_{m_{\max}}^t$. The following inequality holds, since the algorithm chooses node $m_{\max}$ as a state node if $\sigma_{m_{\max}}^{t,LP}$ is at least $1/\delta$:

$$\sum_{t \in T: m_{\max} \in V_t} \sigma_{m_{\max}}^t \leq \delta \times \sum_{t \in T: m_{\max} \in V_t} \sigma_{m_{\max}}^{t,LP}.$$

The following inequality holds, since $\phi_{\max}^{LP}$ is the optimal objective value of the linear relaxation:

$$\delta \times \sum_{t \in T: m_{\max} \in V_t} \sigma_{m_{\max}}^{t,LP} \leq \delta \times \phi_{\max}^{LP}.$$

Since BALANCESTATE is a minimization problem, we obtain the performance bound as

$$\phi_{\max} \leq \delta \times \phi_{\max}^{LP} \leq \delta \times \phi_{\max}^{*}.$$

Note that the number of subtrees in T_m^t is a function of δ, irrelevant to the size of multicast tree t, because $V_{f_m^t}$ of each subtree f_m^t in T_m^t has at most δ branching nodes. In this paper, we assume that δ is a constant in our analysis, because δ is determined by the forwarding speed of a router instead of the network size and the number of trees. Moreover, our simulation results show that a small δ is sufficient to obtain a good solution. Therefore, Step 1 requires constant time to find T_m^t for each node m in each tree t. The number of variables and constraints in BALANCESTATE-ILP is polynomial to the network size and the number of multicast tree. Steps 2 and 3 take polynomial time. Therefore, BALANCESTATE-APX takes polynomial time. □

Corollary 4.7. *The performance bound $\phi_{\max} \leq \delta \times \phi_{\max}^{*}$ is tight.*

Proof. We prove the corollary with the example in Fig. 6. Let all multicast trees use the same routers and have the same routing as the one in Fig. 6. Each tree must choose at least one node from nodes 1 to δ as the state node of the tree in any feasible assignment. Obviously, there exists an optimal solution to the linear relaxation on BALANCESTATE-ILP such that $\sigma_m^{t,LP}$ is $1/\delta$ for each node m in each tree t. The bound is tight when the number of multicast trees is $k\delta$, where k is a positive integer. In this case, BALANCESTATE-APX chooses every node in each tree as a state node, and $\phi_{\max}$ is $k\delta$.

However, the optimal solution selects only one state node for each tree, and the forwarding states are evenly distributed among the δ nodes. Therefore, $\phi^*_{\max}$ is k, and we obtain the tight performance bound. □

In this section, we propose a δ-approximation algorithm, assuming that all nodes, except the leaves in each multicast tree, are branching nodes. If any node m is not a branching node in a tree t, there may exist a subtree $f^t_{p^t_m}$ in $T^t_{p^t_m}$ such that the number of nodes in $V^t_{p^t_m}$ is larger than δ. Therefore, the above algorithm based on rounding may fail to find a feasible solution. The modification of the above δ-approximation algorithm to cope with multicast trees with nonbranching nodes can be found in [29]. The modified algorithm obtains a similar performance bound $\phi_{\max} \leq \delta \times \phi^*_{\max} + 1$. To support the protocol design, we design a distributed algorithm BALANCESTATE-DISTRIBUTED based on MINSTATE-DISTRIBUTED. Each node first tries to remove the forwarding state. If it fails, then it tries moving the forwarding state to the upstream stateless node with the least number of forwarding states, instead of to the parent node in MINSTATE-DISTRIBUTED, to avoid concentrating the forwarding states in some routers.

Our algorithms can be extended to consider the capability of each router. In BALANCESTATE-APX, we substitute (2) with the following:

$$\sum_{t \in T: m \in V_t} \frac{\sigma^t_m}{C_m} \leq \phi_{\max}, \ \forall m \in V,$$

where C_m is the capability of router m, and $\phi_{\max}$ denotes the maximum load of a router. Therefore, for a given number of forwarding states, a larger capability leads to smaller load of a router. BALANCESTATE-APX can solve this problem by the above new constraint, and the algorithm is an approximation algorithm, because Theorem 4.6 still holds after we substitute $\sigma^t_{m_{\max}}$ and $\sigma^{t,LP}_{m_{\max}}$ with $\sigma^t_{m_{\max}}/C_{m_{\max}}$ and $\sigma^{t,LP}_{m_{\max}}/C_{m_{\max}}$ in the proof. Similarly, in BALANCESTATE-DISTRIBUTED, we have to move the forwarding state to the upstream stateless node with the least load.

The protocol based on the distributed algorithm supports dynamic membership as follows: When a member leaves a multicast tree, the upstream state node removes the forwarding state for the member after it receives the leave message from the member. When a member joins a multicast tree, however, the number of destination addresses in each packet sent by the upstream state node may exceed δ after the address of the member is added to the packet. In this case, our protocol first creates a state node between the new member and the upstream state node. Then, each state node tries removing or moving its forwarding state, where the two operations are achieved by message exchanges between neighbor routers.

5 Simulation Results

This section shows our simulation results for the above two optimization problems solved by the proposed algorithms. We compare our algorithms with previous work [13], [14], [15], [16], [17], [18] that allocates forwarding states only to the branching routers of a multicast tree, and this allocation is the same as our mechanism, with δ being equal to one, as explained in Section 3. For previous work [9], [10], [11], [12] that adopts a single tree to serve multiple groups, our mechanism is orthogonal to that approach and can be combined with it to further reduce the number of multicast forwarding states stored in each router. For BALANCESTATE, we compare the solutions obtained by our algorithms with the lower bound on the optimal solution to BALANCESTATE. The optimal solutions must be more than the lower bound but less than the solution obtained by any algorithm of BALANCESTATE. We find the lower bound with an algorithm based on Lagrangean relaxation.

In our simulation, we use flat graphs with the Waxman distribution [21] as the network topologies to test our algorithms in networks with different graph characteristics. We also use MBone [22], a real multicast tree traced by Mwalk [23], [24], and the Internet graphs with the power-law distribution generated by Inet [25], [26] to test our algorithms in more realistic networks. The simulation results are averaged over 100 samples. Our simulation includes the following parameters. The first one is the graph characteristic. We use $(\alpha = 0.2, \beta = 0.2)$, $(\alpha = 0.25, \beta = 0.25)$, and $(\alpha = 0.3, \beta = 0.3)$ as the parameters of the Waxman distribution in the flat graphs. The graphs with larger α and β have larger node degrees and smaller graph diameters [27]. The second one is the group size. The group size is the number of receivers in a multicast tree. The third one is δ, which is the maximum number of destinations from a downstream interface. Since δ is an important parameter in our algorithm, we present our simulation results with different δ values in most figures to find the proper δ in different scenarios. When δ is one, all the branching nodes are the only state nodes in a multicast tree. Finally, the last parameter is the distribution of receivers. We use the affinity and disaffinity model [28] to describe the distribution of receivers in a multicast tree. The distribution of receivers is correlated with the topology of a multicast tree. With a positive affinity index, receivers tend to cluster together, and the distribution is suitable for applications such as local news and traffic reports. With a negative affinity index, receivers tend to spread out, and the distribution is proper for applications such as videoconferencing. When the affinity index is zero, all receivers are uniformly chosen at random among all nodes. We present our simulation results in the flat graphs in Section 5.1 and the graphs that represent the realistic networks in Section 5.2.

5.1 Results in Flat Graphs with Different Characteristics

Figs. 7 and 8 compare the results of MINSTATE with different parameters in the graphs with the Waxman distribution. There are 100 nodes in the network. Fig. 7 shows the results of MINSTATE with different δ values, different group sizes, and different α and β values in the Waxman distribution. We measure the average number of state nodes in a multicast tree. In Fig. 7a, a multicast tree with a larger group size has more state nodes. The number of state nodes in a multicast tree decreases as δ increases. Moreover, a small δ is sufficient to eliminate more than 50 percent of forwarding states. Therefore, our mechanism

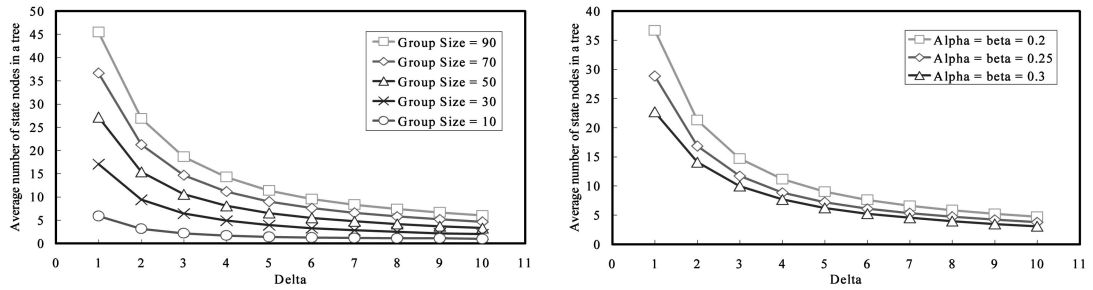


Fig. 7. Average number of state nodes in a multicast tree with different δ values, different group sizes, and different α and β values in the Waxman distribution, where the $\text{affinity index} = 0$. (a) $\alpha = \beta = 0.2$. (b) $\text{Group size} = 70$.

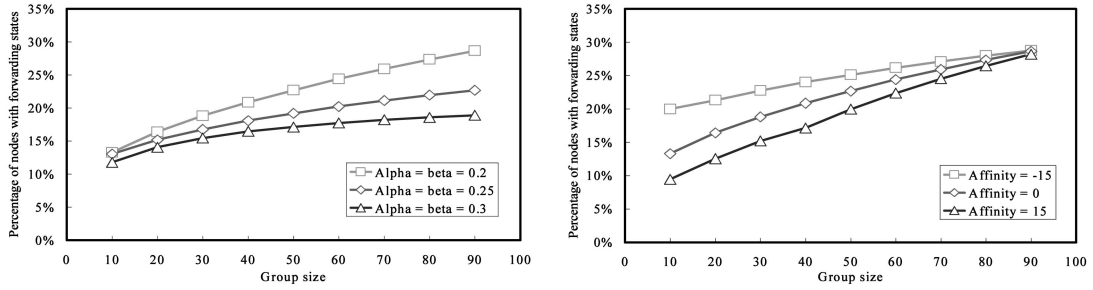


Fig. 8. Percentage of nodes with the forwarding states in a multicast tree with different group sizes, different α and β values in the Waxman distribution, and different affinity indices, where $\delta = 2$. (a) $\text{Affinity index} = 0$. (b) $\alpha = \beta = 2$.

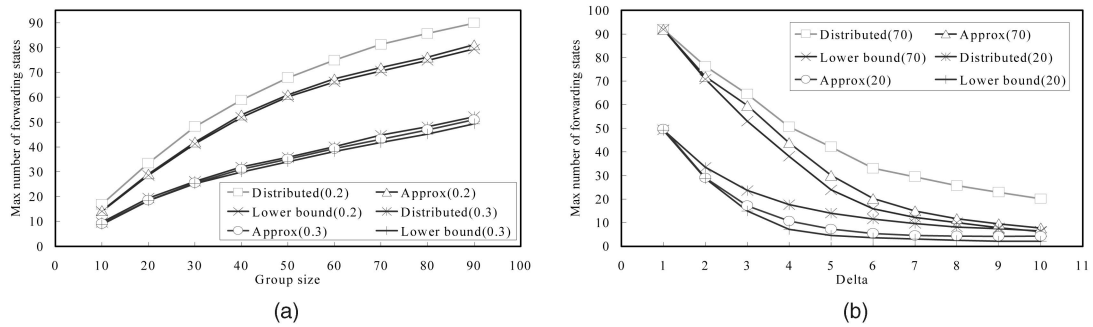


Fig. 9. Maximum number of forwarding states stored in a router with different δ values, different group sizes, and different α and β values in the Waxman distribution, where $\delta = 2$ in (a) and $\alpha = \beta = 0.2$ in (b), with $\text{affinity index} = 0$.

can deliver data with only a few forwarding states without incurring large forwarding delay. In Fig. 7b, a graph with larger α and β uses fewer state nodes in a tree. In this case, the depth of a multicast tree decreases, and each node in a tree has more child nodes. Therefore, each state node can serve more receivers and downstream state nodes, and fewer nodes have to store forwarding states. Moreover, the results of different α and β values converge as δ increases, because fewer nodes need to store forwarding states.

Fig. 8 compares the results of MINSTATE with different group sizes, different α and β values in the Waxman distribution, and different affinity indices. We measure the percentage of nodes with forwarding states in a multicast tree. In Fig. 8a, more nodes in a multicast tree store forwarding states as the group size increases. For a graph with larger α and β values, receivers tend to be connected to on-tree nodes with higher degrees. Therefore, a multicast tree requires only slightly more nodes to store forwarding states as the group size increases. In Fig. 8b, receivers tend to cluster together as the affinity index increases. Fewer nodes in a multicast tree store forwarding states, because a node outside a cluster may need to store only one state for the whole cluster. The results of different affinity indices converge as the group size approaches 100, because the distributions of receivers are similar in such cases.

Fig. 9 gives our algorithms of BALANCESTATE in the graphs with the Waxman distribution. There are 100 nodes and 100 multicast trees in the network. We find the lower bound on the optimal solution to BALANCESTATE with an algorithm based on Lagrangean relaxation. For a graph with larger α and β values, the size and height of a multicast tree decrease, since each node has more child nodes. Each router is used by fewer multicast groups. Therefore, both results in Fig. 9a become smaller. BALANCESTATE-APX performs slightly better than BALANCESTATE-DISTRIBUTED, since the algorithm knows the assignment of state nodes of all multicast trees. Therefore, the router with the most number of forwarding states can create forwarding states in nearby

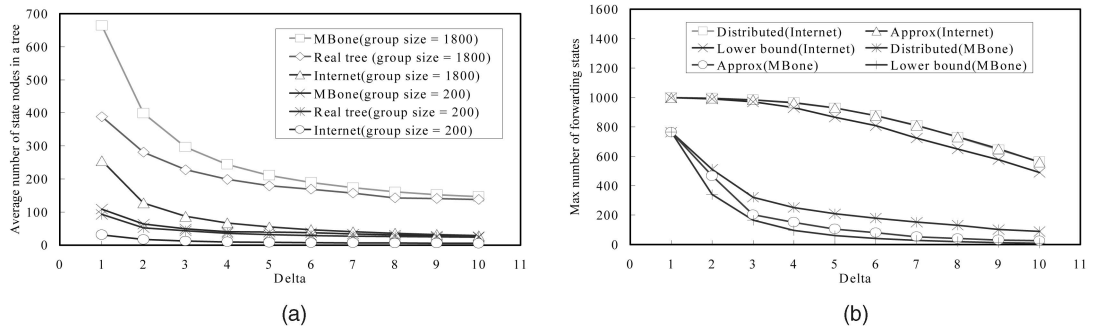


Fig. 10. Average number of state nodes in a multicast tree and maximum number of forwarding states stored in a router in different graphs with different δ values and different group sizes.

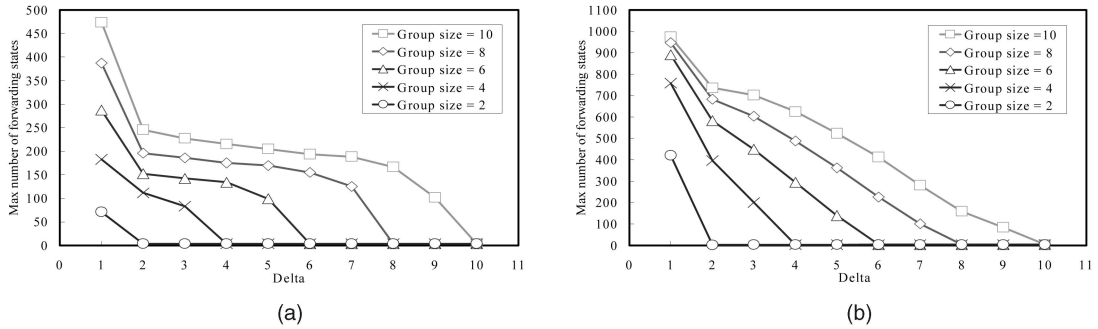


Fig. 11. Maximum number of forwarding states stored in a router in MBone and power-law graphs with different δ values and different group sizes. (a) MBone. (b) Internet graphs.

upstream and downstream routers and remove its forwarding states. Fig. 9b shows our BALANCESTATE algorithms with different δ values and different group sizes. When δ is small, the difference of the two algorithms decreases, because the assignment of state nodes becomes less flexible. There are fewer feasible assignments in this case. The results of the two algorithms approach the lower bound, because the algorithms have more chances of finding a good assignment.

5.2 Results in Graphs Representing Realistic Networks

Fig. 10 gives the results of MINSTATE in MBone, a real multicast tree traced by Mwalk, and large graphs with the power-law distribution generated by Inet. There are 1,000 trees in the network. To compare the results in MBone and the Internet graphs, we generate Internet graphs with 4,177 nodes, identical to MBone. In the two networks, each receiver in a multicast tree is randomly selected among all nodes in the network. The real multicast tree traced by Mwalk has 2,871 nodes, with 1,980 leaf nodes. In the real multicast tree, we randomly choose each receiver from the 1,980 receivers of the tree.[1] We prune some nodes and incident edges of the tree if the nodes and edges are not on the path from the root to any receiver.

We measure the average number of state nodes in a multicast tree as shown in Fig. 10a. The number of state nodes in a multicast tree decreases as δ increases. A small δ is sufficient to eliminate most of the forwarding states. A tree in the Internet graph uses much fewer state nodes than the others, because the Internet graph with the power-law distribution has few nodes with very large degrees. Therefore, the diameter of the Internet graph is smaller than that of MBone, and the height of a multicast tree in the Internet graph is smaller than the height of a tree in the other two graphs. Fig. 10b gives our BALANCESTATE algorithms in MBone and the Internet graphs[2] with different δ values. There are 1,000 multicast trees in the network, and each tree has 50 receivers. The results of MBone are smaller than those of the Internet graphs, and the results of MBone decrease much faster than those of Internet graphs as δ increases. The reason is that a few nodes with very large degrees in the Internet graphs participate in most of the multicast trees. On the contrary, the multicast trees in MBone are more evenly distributed in the network. The distributed algorithm BALANCESTATE-DISTRIBUTED, which is more suitable for protocol designs, performs as well as other algorithms and is close to the lower bound on the optimal solution in the Internet graphs. It is because the assignments of forwarding states in power-law graphs are less flexible than the assignment in MBone.

Fig. 11 measures the maximum number of forwarding states stored in a router obtained when each group has only a few receivers. This is suitable for the scenario that the network serves many multicast groups with the small-group applications such as videoconferences and multiplayer strategy games. For the assignment with only the branching routers storing the states in Internet graphs, a

1. Here, we do not use the affinity model to decide for the receivers of a multicast tree, because the references of MBone and the real multicast tree traced by Mwalk have not yet provided the physical location of each node.

2. Here, we do not compare our BALANCESTATE algorithms in the real multicast tree, because the reference only provides the routing of a multicast tree. The authors have not yet provided the routing of multiple multicast trees and the corresponding physical network.

few routers with very large degrees tend to act as the branching routers of most multicast trees, even when the group size is very small. On the contrary, our mechanism performs much better. A multicast tree maintains no forwarding state if the tree does not have more than δ receivers. Therefore, we can regard δ as a filter that makes a router store the forwarding states of the groups with more than δ receivers. In other words, a multicast tree does not store a forwarding state when the group size is small. As the group size increases, our algorithms select a few on-tree routers to store the forwarding states. The distribution of forwarding states among routers is more balanced, and the forwarding delay of each data packet is limited.

In summary, for MINSTATE, a multicast tree can use a small δ and a few forwarding states to deliver data, even when the group size is very large. In other words, our mechanism can deliver data with only a few forwarding states without incurring large forwarding delay. For BALANCESTATE, the assignment with only branching routers storing the states is not scalable in terms of the number of multicast trees, even when each tree has only a few receivers. The problem becomes more acute for Internet graphs, because a fewer routers with very large degrees tend to act as the branching routers of most multicast trees. On the contrary, our mechanism solves the problem by balancing forwarding states among routers. Our BALANCESTATE algorithms approach the lower bounds on optimal solutions. Therefore, our mechanism can improve the scalability of SSM and IP multicast.

6 Conclusion

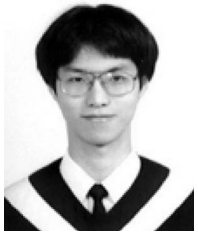
In this paper, we have proposed a scalable and adaptive multicast forwarding mechanism for SSM and IP multicast. Our mechanism is more scalable in terms of both the number of members in a multicast group and the number of multicast groups in the network. Multicast packets are sent between these routers via Xcast. We formulate the assignment of forwarding states among routers as two optimization problems. For each multicast tree, the first problem MINSTATE minimizes the number of routers that store forwarding states of the multicast tree. For the whole network, the second problem BALANCESTATE minimizes the maximum number of forwarding states stored in a router. For MINSTATE, we design a distributed algorithm that finds the optimal solution. For BALANCESTATE, we prove that the problem is NP-hard, and we design an approximation algorithm and a distributed algorithm to solve the problem. In addition, we prove that the previous work that assigns all branching routers as the only state nodes is a special case of our mechanism. Our simulation results show that our mechanism uses fewer forwarding states, and the distribution of forwarding states among routers is more balanced as compared with previous approaches in both artificial and realistic networks. Therefore, our mechanism can improve the scalability of SSM and IP multicast.

Acknowledgments

This work was supported by the National Science Council (NSC), Taiwan, under a Center Excellence Grant NSC95-2752-E-002-006-PAE, and under Grant Number NSC95-2221-E-002-066.

References

[1] P.V. Mieghem, G. Hooghiemstra, and R. Hofstad, "On the Efficiency of Multicast," *IEEE/ACM Trans. Networking,* vol. 9, no. 6, pp. 719-732, Dec. 2001.
[2] W. Fenner, "Internet Group Management Protocol Version 2," *IETF RFC 2236,* Nov. 1997.
[3] D. Waitzman, C. Partridge, and S. Deering, "Distance Vector Multicast Routing Protocol," *IETF RFC 1075,* 1988.
[4] J. Moy, "Multicast Routing Extensions for OSPF," *Comm. ACM,* vol. 37, no. 8, pp. 61-66, 1994.
[5] T. Ballardie, P. Francis, and J. Crowcroft, "Core-Based Trees (CBT)," *Proc. ACM SIGCOMM '93,* pp. 85-95, 1993.
[6] S. Deering et al., "The PIM Architecture for Wide-Area Multicast Routing," *IEEE/ACM Trans. Networking,* vol. 4, no. 2, pp. 153-162, Apr. 1996.
[7] D. Thaler, M. Handley, and D. Estrin, *The Internet Multicast Address Allocation Architecture,* IETF RFC 2908, 2000.
[8] S. Bhattacharyya, *An Overview of Source-Specific Multicast (SSM),* IETF RFC 3569, July 2003.
[9] P.I. Radoslavov, E. Estrin, and R. Govindan, "Exploiting the Bandwidth-Memory Tradeoff in Multicast State Aggregation," technical report, Dept. of Computer Science, Univ. of Southern California, pp. 99-697, 1999.
[10] T. Wong, R. Katz, and S. McCanne, "An Evaluation of Preference Clustering in Large-Scale Multicast Applications," *Proc. IEEE INFOCOM '00,* pp. 451-460, 2000.
[11] S. Song, Z. Zhang, B. Choi, and D.H.C. Du, "Protocol Independent Multicast Group Aggregation Scheme for the Global Area Multicast," *Proc. IEEE GLOBECOM '00,* pp. 370-375, 2000.
[12] A. Fei, J. Cui, M. Gerla, and M. Faloutsos, "Aggregated Multicast: An Approach to Reduce Multicast State," *Proc. IEEE GLOBECOM '01,* pp. 1595-1599, 2001.
[13] J. Tian and G. Neufeld, "Forwarding State Reduction for Sparse Mode Multicast Communication," *Proc. IEEE INFOCOM '98,* pp. 711-719, 1998.
[14] I. Stoica, T.S.E. Ng, and H. Zhang, "REUNITE: A Recursive Unicast Approach to Multicast," *Proc. IEEE INFOCOM '00,* pp. 1644-1653, 2000.
[15] T. Wong and R. Katz, "An Analysis of Multicast Forwarding State Scalability," *Proc. Eighth IEEE Int'l Conf. Network Protocols (ICNP '00),* pp. 105-115, 2000.
[16] V. Visoottiviseth, H. Kido, and Y. Takahashi, "Sender Initiated Multicast (SIM)," IETF Internet Draft, work in progress, Mar. 2003.
[17] A. Boudani and B. Cousin, "Simple Explicit Multicast (SEM)," IETF Internet Draft, work in progress, June 2003.
[18] A. Boudani, B. Cousin, and J. Bonnin, "An Effective Solution for Multicast Scalability: The MPLS Multicast Tree (MMT)," IETF Internet Draft, work in progress, June 2003.
[19] R. Boivie, N. Feldman, Y. Imai, W. Livens, and D. Ooms, *Explicit Multicast (Xcast) Concepts and Options,* IETF RFC 5058, Nov. 2007.
[20] R.M. Karp, "Reducibility among Combinatorial Problems," *Complexity of Computer Computations,* Plenum Press, pp. 85-104, 1972.
[21] B.M. Waxman, "Routing of Multipoint Connections," *IEEE J. Selected Areas in Comm.,* vol. 6, no. 9, pp. 1617-1622, Dec. 1988.
[22] *USC/ISI SCAN Project,* http://www.isi.edu/scan/mbone.html, 2007.
[23] R. Chalmers and K. Almeroth, "On the Topology of Multicast Trees," *IEEE/ACM Trans. Networking,* vol. 11, no. 1, pp. 153-165, Feb. 2003.
[24] *Mwalk Project,* http://www.nmsl.cs.ucsb.edu/mwalk/, 2007.
[25] H. Tangmunarunkit, R. Govindan, and J. Jamin, "Network Topology Generators: Degree-Based vs. Structurel," *Proc. ACM SIGCOMM '02,* pp. 147-159, 2002.
[26] *Inet Topology Generator,* http://topology.eecs.umich.edu/inet/, 2007.
[27] E.W. Zegura, K.L. Calvert, and M.J. Donahoo, "A Quantitative Comparison of Graph-Based Models for Internet Topology," *IEEE/ACM Trans. Networking,* vol. 5, no. 6, pp. 770-783, Dec. 1997.
[28] G. Phillips, S. Shenker, and H. Tangmunarunkit, "Scaling of Multicast Trees: Comments on the Chuang-Sirbu Scaling Law," *Proc. ACM SIGCOMM '99,* pp. 41-51, 1999.
[29] D.-N. Yang, "Scalability in Xcast-Based Multicast," PhD dissertation, Nat'l Taiwan Univ., 2004.



De-Nian Yang received the BS and PhD degrees from the National Taiwan University, Taipei, in 1999 and 2004, respectively. He is currently a postdoctoral researcher for the military services in the Department of Electrical Engineering, National Taiwan University. His research interests include network planning, multicasting, and quality of service (QoS) in wireless networks. He is a member of the IEEE. He received the Best Student Paper Award in the First IEEE International Conferences on Multimedia and Expo (ICME 2000).

Wanjiun Liao received the BS and MS degrees in computer science from the National Chiao-Tung University, Taiwan, in 1990 and 1992, respectively, and the PhD degree in electrical engineering from the University of Southern California (USC), Los Angeles, in 1997. She joined the Department of Electrical Engineering, National Taiwan University (NTU), Taipei, as an assistant professor in 1997 and has been a full professor since August 2005. She was a visiting Professor at the University of Hong Kong, Hong Kong, in the summer of 2006. She is currently on the editorial boards of the *IEEE Transactions on Wireless Communications* and the *IEEE Transactions on Multimedia*. She was the Technical Program Committee (TPC) chair, a cochair, or a member of many international conferences. She was a tutorial cochair of the IEEE INFOCOM 2004, the technical program area chair of the IEEE International Conferences on Multimedia and Expo (ICME) 2004, the technical program vice chair of the IEEE GLOBECOM Symposium on Autonomous Networks 2005, a technical program cochair of the Multimedia Communications and Signal Processing Track of the IEEE International Conference on Communications, Circuits and Systems (ICCCAS) 2006, and a technical program cochair of the IEEE GLOBECOM 2007 General Symposium. Her research interests include wireless networks, multimedia networks, and broadband access networks. She is a coauthor, with her students, of papers that received the Best Student Paper Awards in the IEEE International Conference on Multimedia and Expo (ICME) in 2000 and the IEEE ICCCAS 2002. She is a senior member of the IEEE. She received many research awards. She was a recipient of the Outstanding Research Paper Award from USC in 1997, the Distinguished Teaching Award from NTU in 2000, the K.T. Li Young Researcher Award honored by the ACM in 2003, the Fu Ssu-Nien Award from NTU in 2005 for her research achievements, the Distinguished EE Professor Award from the Chinese IEE in 2006, and the Distinguished Research Award from the National Science Council (NSC) in 2006. She was elected as one of the Outstanding Young EEs by the *EE Times* in 1997 and one of the ROC Ten Distinguished Women in 2000.