Multi-Tree Multicast with a Backpressure Algorithm

Chunglae Cho and Ye Xia

***Abstract*— This paper addresses the problem of utility maximization of multicast sessions with multiple trees over static infrastructure networks. The utility functions are general concave functions. Inspired by the derivation of the unicast backpressure algorithms, we introduce a problem formulation with *tree-flow conservation* constraints and derive a multicast backpressure algorithm. The algorithm is much more distributed and local than previous algorithms. We provide a rigorous analysis of the performance including the primal optimality and the bound on real queue sizes using the Lyapunov optimization technique and the convex optimization techniques together.**

I. Introduction

Massive content distribution has become one of the most important applications on the Internet. One important class of content distribution technique is *swarming*. In a swarming session, a file is broken into many chunks at the source node, which are then distributed to the receivers through various paths. Despite its origin in the end-system-based peer-to-peer community, swarming is also attractive for content distribution services over infrastructure networks provided by network or service providers. For those services, infrastructure nodes are strategically placed by the providers and are well managed and relatively static. In this setting, it has been shown that it is beneficial to view swarming as distribution over *multiple* multicast trees [1], [2]. This view allows us to focus on a formal approach based on optimization theory to develop optimal solutions systematically.

In this paper, we study the problem of utility maximization of multicast sessions with multiple trees. We suppose that each session has an infinite data backlog in its source and is given a small number of trees over which its data is divided and then distributed to its receivers. The objective is to maximize the sum of session utilities which are functions of their admitted rates. We assume that the utility functions are general concave functions, which can reflect various fairness criteria among sessions. The major constraints are link capacity constraints. Then, our problem is to find the optimal tree rate allocation while the queues in the network remain finite.

Our contributions are as follows. First, we present a backpressure algorithm for the utility maximization problem of multi-tree multicast. Backpressure algorithms are desirable since they are distributed and local: There is no global exchange of control messages. Various forms of backpressure algorithms for the unicast network flow problems have been introduced [3]–[9]. However, applying the backpressure approach to multicast problems is not straightforward. Most backpressure algorithms for unicast are derived by relaxing the flow conservation constraints in the static optimization formulation(see [6], [9], [10] for representative examples), whereas it is not obvious what constraints represent such flow conservation in multicast problems. To circumvent the difficulty, we introduce a problem formulation with *tree-flow conservation* constraints, by which we mean that the amount of flow on a link on a tree should be no less than the amount of flow on its parent link with respect to the tree. Then, we present a subgradient algorithm by relaxing the tree-flow conservation constraints. The algorithm is more local than earlier multicast algorithms in [1], [2], [11], [12]. The control is distributed over the network components and the communication overhead is low since each node only needs to know the queue sizes at itself and at its neighboring nodes.

Second, we show how to analyze the optimality and the network stability of the subgradient algorithm under a general concave objective function. It is hard to simply apply the convex optimization techniques for subgradient algorithms to show the primal optimality of our algorithm. This is mainly due to the assumption of general concavity rather than strict concavity on the utility functions. The result is the lack of a continuous map from the dual to the primal variables, which is crucial to prove primal convergence as described in [13], [14]. Even though the dual variables converge, the primal variables oscillate. Instead, we show the primal optimality of the algorithm in the long-time average sense using the Lyapunov optimization technique in [7], [9]. However, we still need to use the convex optimization techniques in [14], [15] to prove the dual optimality. Since the subgradient algorithm uses virtual queues, we can only show the boundedness of long-time average virtual queue backlogs with the Lyapunov technique, which is not enough to show network stability. We use the result on the dual optimality to show the real queue backlogs are bounded. The two sets of analytical techniques complement each other in our analysis.

The use of the optimization approach on multicast problems has been reported in [1], [2], [11], [12], [16]–[18]. Most of them, except [16], [18], require global exchange of control messages. In the algorithms, source nodes and links belonging to the same session need to exchange their session rates and link prices. In [11], [16], [18], the authors consider multi-rate multicast problems with a single fixed multicast tree for each session. In [1], [17], special objectives are considered such as maximizing throughput or minimizing network congestion. The authors of [12] assume that the network bottlenecks are at the nodes' uplink capacities

C. Cho and Y. Xia are with the Department of Computer and Information Science and Engineering, University of Florida, Gainesville, FL 32611. {ccho, yx1}@cise.ufl.edu.

whereas we assume that the bottlenecks can be anywhere in the network. In [1], [2], although an optimal set of trees with an optimal rate allocation are found among all possible trees, the algorithms are less local and the computation requirements at each iteration are much heavier than the proposed algorithm. In [18], a similar problem formulation to ours is introduced; but the algorithms are different. The algorithm in [18] uses real queue updates whereas ours uses virtual queue updates. However, our algorithm has stronger results on the real queue boundedness.

The remaining paper is organized as follows. Section II describes the network model and problem formulation. In Section III, we present the backpressure algorithm. In Section IV, we show that our algorithm achieves an optimal solution and the virtual queues are bounded in the long-time average sense. We provide stronger results on queue boundedness in Section V. The concluding remarks are in Section VI.

II. Problem Description

A. Network Model

Consider a network which is represented by a directed, edge-capacitated graph $G = (V, E)$, where V is the set of nodes and E is the set of directed links. Each link e in E has a finite capacity $c_e > 0$. Let $c_{max} \triangleq \max_{e\in E} c_e$ be the maximum link capacity over all links. Let S be the set of all multicast sessions in the network and $|S|$ be the number of sessions. Each session is associated with a source and a set of destinations, which is a subset of V.

Each session s is given a set of trees T_s that it uses for data transmission. Let $|T_s|$ be the number of trees in T_s. Let T be the union of T_s. Note that if $t_1 \in T_{s_1}$ and $t_2 \in T_{s_2}$ have the same topology, we regard them as different trees. Let r^t_e be the transmission rate assigned for tree t on link e. Let E_t be the set of links on tree t. Denote $b(e)$ and $d(e)$ to be the transmitting (tail) node and the receiving (head) node of link e, respectively. Let V_t be the set of transmitting nodes $b(e)$ for all e in E_t. V_t represents the set of nodes on tree t which are not leaves. Let $o(t)$ be the root node of tree t. Let $p(t, e)$ be the parent link of link e on tree t. Let $\Omega(t, n)$ be the set of child links at node n on tree t. Let x_s be the admitted rate for session s, and y^t_n be the admitted rate at node n on tree t. We assume that each source has an infinite backlog of data. Let Q^t_e be the real queue size at link e for tree t.

B. Problem Formulation

Denote (a_i) to be the vector with entries a_i. Let x, y and r be the vectors $(x_s)_{s\in S}$, $(y^t_n)_{t\in T,n\in V_t}$, and $(r^t_e)_{t\in T,e\in E_t}$, respectively. The notation such as $(a)_{t\in T}$ means repeating the value a in all entries of the vector. Denote $\mathbf{1}$ to be the vector whose entries are all 1's.

Let $\bar{x}_s(k)$ and $\bar{x}_s$ be the time average of the admitted traffic rate for session s up to time k and its limit [1], respectively, i.e.,

$$\bar{x}_s(k) \triangleq \frac{1}{k}\sum_{\kappa=0}^{k-1} x_s(\kappa),\ \bar{x}_s \triangleq \lim_{k\to\infty} \bar{x}_s(k).$$

We define $\bar{y}^t_n(k)$, $\bar{y}^t_n$, $\bar{r}^t_e(k)$ and $\bar{r}^t_e$ similarly.

Our problem $\mathbf{P}$ is as follows.

$$\begin{aligned} \mathbf{P}:\ \max\ & \sum_{s\in S} U_s(\bar{x}_s) \\ \text{s.t.}\ & \bar{r}^t_e - \bar{r}^t_{p(t,e)} - \bar{y}^t_{b(e)} \ge 0, \quad \forall t \in T, \forall e \in E_t, && (1) \\ & \sum_{t\in T_s} \bar{y}^t_{o(t)} = \bar{x}_s, \quad \forall s \in S, && (2) \\ & \sum_{t:e\in E_t} \bar{r}^t_e \le c_e, \quad \forall e \in E, && (3) \\ & 0 \le \bar{x}_s \le X_{max}, \quad \forall s \in S, && (4) \\ & \bar{y}^t_n \ge 0, \quad \forall t \in T, \forall n \in V_t, && (5) \\ & \bar{r}^t_e \ge 0, \quad \forall t \in T, \forall e \in E_t. && (6) \end{aligned}$$

The constraints in (1) imply that for every tree t, the allocated link rate on a link e on the tree should be no less than the sum of the link rate of its parent link $p(t, e)$ and the exogenous arrival to the tail node $b(e)$. It can be considered as a relaxed form of the flow conservation constraints for trees. We assume that $\bar{r}^t_{p(t,e)} \triangleq 0$ if $p(t, e)$ is null. Note that the tree rate variables $\bar{y}^t_n$ for tree t are defined for all non-leaf nodes $n \in V_t$. The variables $\bar{y}^t_n$ for $n \neq o(t)$ appear unnecessary because the only other constraints on them are non-negativity constraints in (5). In fact, our algorithm always assigns 0 to those variables. However, these variables are needed for performance analysis, which will become clear in Section IV.

The constraints in (2) mean that the aggregate rate transmitted by the trees for a session must be equal to the session rate. The constraints in (3) are the link capacity constraints that the sum of the transmission rates on a link cannot exceed the link capacity. In (4), we assume the session rates must be bounded by X_{max} from above, where X_{max} is a constant.

Let Λ be

$$\Lambda \triangleq \{(x, y, r)|(x, y, r) \text{ is feasible to problem } \mathbf{P}.\}.$$

We have the following assumption on Λ.

Assumption 1: There exists a feasible solution of $\mathbf{P}$ such that the constraints in (1) hold with strict inequality.

It is highly probable that the above assumption is valid in real networks and it is used for showing the dual optimality of our algorithm.

We also define the sets $\Lambda_{(x,y)}$ and Λ_r as follows, respectively.

$$\begin{aligned} \Lambda_{(x,y)} \triangleq\ & \{(x, y)|(x, y) \text{ satisfies the constraints in (2), (4)} \\ & \text{and (5).}\}, \\ \Lambda_r \triangleq\ & \{r|r \text{ satisfies the constraints in (3) and (6).}\}. \end{aligned}$$

These notations will be used to simplify later expressions.

We have the following assumptions on the utility function $U_s(x_s)$.

[1]We temporarily assume that the limit exists. We shall replace $\lim$ with $\liminf$ or $\limsup$ in case that the limit does not exist.

Assumption 2: U_s is a concave function. U_s is continuous and differentiable. The derivative of U_s is bounded on $[0, X_{max}]$.

Note that U_s is a somewhat general concave function. It could be linear or non-strictly concave. We do not even assume that it is monotonically increasing. Under this assumption, which is more general than that in [6], [14], we do not have a continuous map from the dual variables to the primal variables in the subgradient algorithm. Such continuity is available in [6], [14] and is used to prove the primal optimality. The assumption also implies that U_s is bounded on $[0, X_{max}]$.

III. Backpressure Algorithm

Let $[\cdot]^+$ and $[\cdot]_a^b$ denote the projection onto the non-negative domain and the interval of $[a, b]$, respectively. Let γ_e^t be the non-negative Lagrange multipliers associated with the constraints in (1). Let γ be the vector $(\gamma_e^t)_{t\in T, e\in E_t}$. From problem $\mathbf{P}$, we relax the constraints in (1) and write the Lagrangian as follows.

$$\begin{aligned}
&L(x, y, r; \gamma) \\
=& \sum_{s\in S} U_s(x_s) + \sum_{t\in T}\sum_{e\in E_t} \gamma_e^t (r_e^t - r_{p(t,e)}^t - y_{b(e)}^t) \\
=& \sum_{s\in S} \Big(U_s(x_s) - \sum_{t\in T_s} y_{o(t)}^t \sum_{e\in\Omega(t,o(t))} \gamma_e^t \Big) \\
&- \sum_{s\in S}\sum_{t\in T_s}\sum_{e\in E_t : e\notin\Omega(t,o(t))} y_{b(e)}^t \gamma_e^t \\
&+ \sum_{e\in E}\sum_{s\in S}\sum_{t\in T_s : e\in E_t} r_e^t \Big(\gamma_e^t - \sum_{e'\in\Omega(t,d(e))} \gamma_{e'}^t\Big).
\end{aligned}$$

The last equality holds because, for all t,

$$\sum_{e\in E_t} y_{b(e)}^t \gamma_e^t = \sum_{e\in\Omega(t,o(t))} y_{b(e)}^t \gamma_e^t + \sum_{e\in E_t : e\notin\Omega(t,o(t))} y_{b(e)}^t \gamma_e^t,$$

where $b(e) = o(t)$ for $e \in \Omega(t, o(t))$.

Then, the dual function is given by

$$\begin{aligned}
&D(\gamma) \\
=& \max_{(x,y)\in\Lambda_{(x,y)}, r\in\Lambda_r} L(x, y, r; \gamma) \\
=& \max_{(x,y)\in\Lambda_{(x,y)}} \sum_{s\in S} \Big(U_s(x_s) - \sum_{t\in T_s} y_{o(t)}^t \sum_{e\in\Omega(t,o(t))} \gamma_e^t \Big) \\
&+ \max_{(x,y)\in\Lambda_{(x,y)}} \sum_{s\in S} \Big(-\sum_{t\in T_s}\sum_{e\in E_t : e\notin\Omega(t,o(t))} y_{b(e)}^t \gamma_e^t \Big) \\
&+ \max_{r\in\Lambda_r} \sum_{e\in E}\sum_{s\in S}\sum_{t\in T_s : e\in E_t} r_e^t \Big(\gamma_e^t - \sum_{e'\in\Omega(t,d(e))} \gamma_{e'}^t\Big) \qquad (7) \\
=& \max_{(x,y)\in\Lambda_{(x,y)}} \sum_{s\in S} \Big(U_s(x_s) - \sum_{t\in T_s} y_{o(t)}^t \sum_{e\in\Omega(t,o(t))} \gamma_e^t \Big) \\
&+ \max_{r\in\Lambda_r} \sum_{e\in E}\sum_{s\in S}\sum_{t\in T_s : e\in E_t} r_e^t \Big(\gamma_e^t - \sum_{e'\in\Omega(t,d(e))} \gamma_{e'}^t\Big),
\end{aligned}$$

where the second equality holds because the constraints of the maximization can be separated over the three terms, and the last equality holds because the maximum of the second term in (7) is always zero. Then, the dual problem is as follows.

$$\begin{aligned}
&\min\ D(\gamma) \\
&\text{s.t.}\ \ \gamma_e^t \geq 0, \quad \forall t \in T, \forall e \in E_t.
\end{aligned}$$

Following the standard subgradient method [15], we have a subgradient algorithm as follows.

$$(x(k), y(k), r(k)) = \arg \max_{(x,y)\in\Lambda_{(x,y)}, r\in\Lambda_r} L(x, y, r; \gamma(k)), \qquad (8)$$

$$\begin{aligned}
&\gamma_e^t(k+1) = [\gamma_e^t(k) - \delta(r_e^t(k) - r_{p(t,e)}^t(k) - y_{b(e)}^t(k))]^+, \\
&\quad \forall t \in T, \forall e \in E_t, \qquad (9)
\end{aligned}$$

where $\delta > 0$ is a step size.

Let $q_e^t(k) = (1/\delta)\gamma_e^t(k)$, which represents a virtual queue size at link e for tree t at time slot k. Let $q(k)$ be the vector $(q_e^t(k))_{t\in T, e\in E_t}$. Then, we can rewrite the algorithm as follows.

- Session and tree rate control: At each time slot k, session s solves the following optimization problem and assigns an optimal solution to $x_s(k)$ and $y_{o(t)}^t(k)$ for $t \in T_s$.

$$\begin{aligned}
\max\ & \frac{1}{\delta} U_s(x_s) - \sum_{t\in T_s} y_{o(t)}^t \sum_{e\in\Omega(t,o(t))} q_e^t(k) \qquad (10) \\
\text{s.t.}\ & \sum_{t\in T_s} y_{o(t)}^t = x_s, \\
& 0 \leq x_s \leq X_{max}, \\
& y_{o(t)}^t \geq 0, \quad \forall t \in T_s.
\end{aligned}$$

- Link scheduling: At each time slot k, each link e solves the following optimization problem and assigns an optimal solution to $r_e^t(k)$ for trees t such that $e \in E_t$.

$$\begin{aligned}
\max\ & \sum_{t: e\in E_t} r_e^t \Big(q_e^t(k) - \sum_{e'\in\Omega(t,d(e))} q_{e'}^t(k)\Big) \qquad (11) \\
\text{s.t.}\ & \sum_{t: e\in E_t} r_e^t \leq c_e, \\
& r_e^t \geq 0,\ \forall t : e \in E_t.
\end{aligned}$$

- Virtual queue update: At each time slot k, each link e updates the virtual queues for trees t such that $e \in E_t$ as follows:

$$q_e^t(k+1) = [q_e^t(k) - r_e^t(k) + r_{p(t,e)}^t(k) + y_{b(e)}^t(k)]^+. \qquad (12)$$

The subproblems (10) and (11) can be easily solved. First, we can solve subproblem (10) as follows. Let $t_s^*(k)$ be a tree with the minimum total queue backlog at the source for the outgoing links on the tree:

$$t_s^*(k) \in \arg\min_{t\in T_s} \sum_{e\in\Omega(t,o(t))} q_e^t(k).$$

If there are more than one such trees, we pick one of them arbitrarily but deterministically. Then, we consider the following expression:

$$\zeta_s(x_s) \triangleq \frac{1}{\delta} U_s'(x_s) - \sum_{e \in \Omega(t_s^*(k), o(t_s^*(k)))} q_e^{t_s^*(k)}(k).$$

If there exists x_s on $[0, X_{max}]$ satisfying the equality $\zeta_s(x_s) = 0$, let $x_s(k)$ denote it. Otherwise, we set $x_s(k)$ as follows:

$$x_s(k) = \begin{cases} X_{max}, & \text{if } \zeta_s(x_s) > 0 \text{ for all } x_s \in [0, X_{max}], \\ 0, & \text{otherwise.} \end{cases}$$

Then, for all $t \in T_s$, we set $y_{o(t)}^t$ to be

$$y_{o(t)}^t(k) = \begin{cases} x_s(k), & \text{if } t = t_s^*(k), \\ 0, & \text{otherwise.} \end{cases}$$

Note that since U_s' may not be one-to-one, $x_s(k)$ can oscillate over time even if $q_e^t(k)$ stabilizes. Once source s determines its session rate $x_s(k)$ and tree $t_s^*(k)$, it pushes $x_s(k)$ amount of data from its reservoir to the outgoing links at the root of the selected tree. To summarize, source s selects only one tree that has the minimum total queue backlog at the source for the outgoing links on the tree. Then, it sends x_s amount of data onto the selected tree.

Subproblem (11) can be solved as follows. First, each link e finds the tree with the maximum differential backlog,

$$\beta_e^*(k) \triangleq \max_{t: e \in E_t} \{q_e^t(k) - \sum_{e' \in \Omega(t, d(e))} q_{e'}^t(k)\}. \quad (13)$$

If $\beta_e^*(k) \geq 0$, then let $\tau_e^*(k)$ be the tree that solves (13) with tie broken deterministically. If $\beta_e^*(k) < 0$, then we set $\tau_e^*(k)$ to be null. Next, link e assigns $r_e^t(k)$ for each tree t on link e as follows:

$$r_e^t(k) = \begin{cases} c_e, & \text{if } t = \tau_e^*(k), \\ 0, & \text{otherwise.} \end{cases}$$

If $\tau_e^*(k)$ is null, all $r_e^t(k)$ at link e are assigned zero.

After link e determines the tree $\tau_e^*(k)$ which uses the link capacity exclusively, the tail node of the link transmits c_e amount of data if it has sufficient data in the (real) queue. Otherwise, it transmits only the data in the queue and does not use the remaining link capacity.[2] Then, the receiving node duplicates the received data into the queues of the child links of link e on the tree.

It is not difficult to check that at each time slot k, the above algorithm finds a solution that maximizes the Lagrangian L given the dual variables $\delta q(k)$. This property is important in the later performance analysis.

The parameter δ can be used to adjust the performance bound. Rewriting the algorithm from (8)-(9) into (10)-(12) has the benefit that only the sources need to adjust δ if the performance bound needs to be changed.

In a real implementation, each source node performs the session and tree rate allocation and each node performs the link scheduling and virtual queue updates for its outgoing links. They exchange the virtual queue length of their outgoing links with their neighbors at each time slot. Furthermore, since the total number of trees is reasonably small, each node can keep the tree topology information. Therefore, each packet only needs to carry its tree identifier.

[2]In the analysis of real queue boundedness, we need to assume that the remaining link capacity is not used.

IV. Long-time Average Performance Analysis

We show in this section that with the algorithm (10)-(12), the achieved utility can be arbitrarily close to the optimum and the virtual queue sizes are bounded in the long-time average sense. The analysis follows the Lyapunov optimization technique in [7], [9].

Define $\mathcal{Y}$ to be

$$\mathcal{Y} \triangleq \{y | \exists (x, r) \text{ such that } (x, y, r) \text{ is feasible to } \mathbf{P}.\}.$$

Let Y be the largest admitted rate on y_n^t such that the vector $y_{sym} \triangleq (Y)_{t \in T, n \in V_t}$, whose entries are all Y's, is in $\mathcal{Y}$. That is, y_{sym} is obtained by pushing the same rates as much as possible not only into the root nodes of the trees but also into all other non-leaf nodes of the trees.

We consider the following ϵ-tightened problem $\mathbf{P}(\epsilon)$.

$$\begin{aligned} \mathbf{P}(\epsilon): \ \max \ & \sum_{s \in S} U_s(\bar{x}_s) \\ \text{s.t.} \ & \bar{r}_e^t - \bar{r}_{p(t,e)}^t - \bar{y}_{b(e)}^t \geq \epsilon, \quad \forall t \in T, \forall e \in E_t, \\ & \text{and the constraints (2) - (6),} \end{aligned}$$

where $0 < \epsilon < Y$. Problem $\mathbf{P}(\epsilon)$ is well defined for all ϵ such that $0 < \epsilon < Y$. Since y_{sym} is in $\mathcal{Y}$, $(Y - \epsilon)_{t \in T, n \in V_t}$ is a part of a feasible solution to problem $\mathbf{P}(\epsilon)$. Therefore, we can always find a feasible solution for all ϵ such that $0 < \epsilon < Y$.

The above problem formulation is obtained by tightening the constraints in (1) of problem $\mathbf{P}$. Such tightening can be considered as, for every tree, pushing additional ϵ flow into every non-leaf node on the tree.

Define $\Lambda(\epsilon)$ and $\mathcal{Y}(\epsilon)$ to be

$$\Lambda(\epsilon) \triangleq \{(x, y, r) | (x, y, r) \text{ is feasible to } \mathbf{P}(\epsilon).\},$$
$$\mathcal{Y}(\epsilon) \triangleq \{y | \exists (x, r) \text{ such that } (x, y, r) \text{ is feasible to } \mathbf{P}(\epsilon).\},$$

respectively, where $0 < \epsilon < Y$. It is easy to see that $\Lambda(\epsilon)$ and $\mathcal{Y}(\epsilon)$ are subsets of Λ and $\mathcal{Y}$, respectively. That is, any feasible solution of problem $\mathbf{P}(\epsilon)$ is feasible to problem $\mathbf{P}$.

Let (x^*, y^*, r^*) and $(x^*(\epsilon), y^*(\epsilon), r^*(\epsilon))$ be some optimal solutions of problem $\mathbf{P}$ and $\mathbf{P}(\epsilon)$, respectively. Note that $(x^*(\epsilon), y^*(\epsilon), r^*(\epsilon))$ is feasible to problem $\mathbf{P}$.

Remark: The optimal solution of the ϵ-tightened problem is used in the analysis. To get a proper performance bound, it is important to decide which constraints are tightened. Here, we tighten the constraints that were relaxed when we derived the algorithm. It allows us to easily associate ϵ with each of the virtual queues, which is crucial to get the virtual queue size bound.

Define U_{max} to be $\sum_{s \in S} \max_{0 \leq x_s \leq X_{max}} U_s(x_s)$. Note that U_{max} is well defined. Let f^* and $f^*(\epsilon)$ be the optimal objective values of problem $\mathbf{P}$ and $\mathbf{P}(\epsilon)$, respectively. Then, we have the following lemma.

Lemma 1:

$$f^*(\epsilon) \to f^*, \text{ as } \epsilon \to 0.$$

Proof: The proof is omitted for brevity. ■

Let $\sum_{s,t,e}(\cdot)$ be the abbreviated notation of $\sum_{s\in S}\sum_{t\in T_s}\sum_{e\in E_t}(\cdot)$. Define the Lyapunov function $V(k)$ and the Lyapunov drift $\Delta(k)$ as follows:

$$V(k) \triangleq \sum_{s,t,e}(q_e^t(k))^2,\ \Delta(k) \triangleq V(k+1) - V(k).$$

Then, we can get a bound for the Lyapunov drift as follows.

Lemma 2: For any finite $\delta > 0$ and $0 < \epsilon < Y$, there exists a constant B such that for all time slots t and all virtual queue sizes $q(k)$, the Lyapunov drift satisfies

$$\Delta(k) - \frac{2}{\delta}\sum_{s\in S} U_s(x_s(k)) \le B - \frac{2}{\delta}f^*(\epsilon) - 2\epsilon\sum_{s,t,e} q_e^t(k). \tag{14}$$

Proof: For brevity, we only give a sketch of the proof. By squaring (12) and followed by simple manipulation, we have

$$\begin{aligned}&(q_e^t)^2(k+1) - (q_e^t)^2(k)\\ &\le (r_e^t(k) - r_{p(t,e)}^t(k) - y_{b(e)}^t(k))^2\\ &\quad - 2q_e^t(k)(r_e^t(k) - r_{p(t,e)}^t(k) - y_{b(e)}^t(k)).\end{aligned}$$

Summing the inequality over all $t \in T$ and all $e \in E_t$ and adding the term $-(2/\delta)\sum_{s\in S} U_s(x_s(k))$ to the both sides of the inequality, we have

$$\begin{aligned}&\Delta(k) - \frac{2}{\delta}\sum_{s\in S} U_s(x_s(k))\\ &\le B - 2\sum_{s\in S}\Big(\frac{1}{\delta}U_s(x_s(k)) - \sum_{t\in T_s}\sum_{e\in E_t} y_{b(e)}^t(k)q_e^t(k)\Big)\\ &\quad - 2\sum_{s,t,e} q_e^t(k)\Big(r_e^t(k) - r_{p(t,e)}^t(k)\Big),\end{aligned} \tag{15}$$

where $B \triangleq \sum_{s,t,e}(c_{max} + X_{max})^2$.

Since the algorithm (10)-(11) greedily minimizes the right hand side of (15), and $(x^*(\epsilon), y^*(\epsilon), r^*(\epsilon))$ is feasible to problem $\mathbf{P}$ and $\mathbf{P}(\epsilon)$, after a few simple manipulations, we get (14). ■

Using the above lemma, we can derive the following theorem.

Theorem 3: For any parameter $\delta > 0$, the algorithm (10)-(12) satisfies the following performance bounds.

$$\liminf_{k\to\infty}\sum_{s\in S} U_s(\bar{x}_s(k)) \ge f^* - \frac{B\delta}{2}, \tag{16}$$

$$\limsup_{k\to\infty}\frac{1}{k}\sum_{\kappa=0}^{k-1}\sum_{s,t,e} q_e^t(\kappa) \le \frac{B + 2U_{max}/\delta}{2Y}. \tag{17}$$

Proof: For brevity, we only give a sketch of the proof. From (14), summing the inequality over $k \in \{0, 1, ..., K-1\}$, we get

$$\begin{aligned}&V(K) - V(0) - \frac{2}{\delta}\sum_{\kappa=0}^{K-1}\sum_{s\in S} U_s(x_s(\kappa))\\ &\le BK - \frac{2K}{\delta}f^*(\epsilon) - 2\epsilon\sum_{\kappa=0}^{K-1}\sum_{s,t,e} q_e^t(\kappa).\end{aligned} \tag{18}$$

Using the fact that removing the terms $V(K)$ and $-2\epsilon\sum_{\kappa=0}^{K-1}\sum_{s,t,e} q_e^t(\kappa)$ preserves the inequality, rearranging (18), and taking the $\liminf$ as $K \to \infty$, we get

$$\liminf_{K\to\infty}\frac{1}{K}\sum_{\kappa=0}^{K-1}\sum_{s\in S} U_s(x_s(\kappa)) \ge f^*(\epsilon) - \frac{\delta B}{2}.$$

Using Jensen's inequality and letting $\epsilon \to 0$, we get (16).

On the other hand, from (18), using the definition of U_{max} and the fact that removing the terms $V(K)$ and $-2Kf^*(\epsilon)/\delta$ preserves the inequality, rearranging (18), and taking the $\limsup$ as $K \to \infty$, we get

$$\limsup_{k\to\infty}\frac{1}{k}\sum_{\kappa=0}^{k-1}\sum_{s,t,e} q_e^t(\kappa) \le \frac{B + 2U_{max}/\delta}{2\epsilon}.$$

Letting $\epsilon \to Y$, we get (17). ■

Theorem 3 implies that the long-time average of the solution given by the algorithm can be arbitrarily close to the optimal solution of problem $\mathbf{P}$ by choosing δ to be arbitrarily small. However, the bound on the long-time average of the virtual queue sizes increases as δ decreases. Therefore, the parameter δ can be used as a knob to tradeoff the optimization performance and virtual queue size bounds.

Note that the boundedness of the virtual queue sizes does not imply that the algorithm stabilizes the network. To claim that, we have to show that the real queue sizes are bounded, which we will do in Section V.

V. Further Results on Queue Boundedness

In this section, we show stronger results on the boundedness of the queues. We show that the virtual and real queue sizes at every time slot are bounded above, respectively.

A. Dual Optimality and Boundedness of Virtual Queues

The boundedness of the virtual queues can be proven from the dual optimality of our algorithm using the convex optimization techniques in [14], [15].

Let $J_e^t(y, r) \triangleq r_e^t - r_{p(t,e)}^t - y_{b(e)}^t$ and $J(y, r)$ be the vector $(J_e^t(y, r))_{t\in T, e\in E_t}$. Since x, y and r are bounded, there exists a constant $M_J < \infty$ such that

$$M_J \ge \max_{(x,y)\in\Lambda_{(x,y)}, r\in\Lambda_r}\|J(y, r)\|^2.$$

Let $\Gamma^* \triangleq \{\gamma^* \ge 0 | D(\gamma^*) = \min_{\gamma\ge 0} D(\gamma)\}$ be the set of optimal dual solutions of problem $\mathbf{P}$.

Lemma 4: Γ^* is non-empty, closed and bounded.

Proof: The proof is omitted for brevity. ■

For finite $\eta > 0$ and $\gamma^* \in \Gamma^*$, let $\Gamma(\eta) \triangleq \{\gamma \ge 0 | D(\gamma) \le D(\gamma^*) + \eta\}$. Note that $\Gamma(\eta)$ is bounded.

Now, we are ready to show that the scaled virtual queue vectors $\delta q(k)$ can be arbitrarily close to Γ^* by choosing small enough $\delta > 0$. Let $d(\gamma, \Gamma^*) \triangleq \min_{\gamma^*\in\Gamma^*}\|\gamma - \gamma^*\|$ be the distance between γ and the nearest optimal solution of the dual. For finite $\eta > 0$, let $\xi(\eta) \triangleq \max_{\gamma\in\Gamma(\eta)} d(\gamma, \Gamma^*) + \eta$.

Lemma 5: For any $\epsilon > 0$, there exist $\delta > 0$ and a sufficiently large $K_0 < \infty$ such that, with any finite initial feasible $q(0)$, for all $k \ge K_0$, $d(\delta q(k), \Gamma^*) < \epsilon$.

Proof: For brevity, we only give a sketch of the proof. It is similar to the proof of Proposition 5 in [14]. The vector $J(y(k), r(k))$ is a subgradient of D at $\delta q(k)$ (see [15], page 731). Fix $\eta > 0$ and pick δ such that $\delta \leq \min\{\eta/M_J, \eta/\sqrt{M_J}\}$. Then, there exists a time K_0 such that $d(\delta q(k), \Gamma^*) \leq \xi(\eta)$ for all $k \geq K_0$. Since $\xi(\eta) \to 0$ as $\eta \to 0$, by picking $\eta > 0$ sufficiently small, we have $\xi(\eta) < \epsilon$. ■

In the next lemma, we show that the virtual queue backlogs at every time slot are bounded above in our algorithm.

Lemma 6: For any finite $\delta > 0$, there exists a finite $M_q > 0$ such that, for every link e and tree t, the virtual queue size

$$q_e^t(k) < M_q,$$

for all time slots k.

Proof: For brevity, we only give a sketch of the proof. Pick any $\gamma^* \in \Gamma^*$. Given a finite $\delta > 0$, we choose η such that $\eta = \max\{\delta M_J, \delta\sqrt{M_J}\}$. Since $\delta \leq \min\{\eta/M_J, \eta/\sqrt{M_J}\}$, by the same argument used in Lemma 5, there exists K_0 such that $d(\delta q(k), \gamma^*) \leq \xi(\eta)$. Then, by the boundedness of Γ^*, $\sup_k \|\delta q(k)\| < \infty$. Since $\delta > 0$, $\sup_k \|q(k)\| < \infty$. ■

B. Boundedness of Real Queues

To argue that the network is stabilized with the algorithm, we need to show that the real queue sizes are bounded. In [9], [18], the authors show the stability of their algorithms by claiming that the long-time average of the real queue sizes are bounded. In this section, we show that with our algorithm, the real queue sizes at every time slot are bounded, which is a stronger result.

Let $Q_e^t(k)$ be the real queue size at link e for tree t at time slot k and $Q(k)$ be the vector $(Q_e^t(k))_{t\in T, e\in E_t}$. The algorithm (10)-(12) satisfies the following real queue bound.

Theorem 7: There exists a finite $M_q > 0$ such that, for every link e and tree t, the real queue size

$$Q_e^t(k) < M_q,$$

for all time slots k.

Proof: Let $R_e^t(k)$ be the amount of real traffic transmitted at link e on tree t at time slot k and $R(k)$ be the vector $(R_e^t(k))_{t\in T, e\in E_t}$.

Assume that all the real and virtual queues are empty at time $k = 0$. Applying Loynes' formula, we have

$$\begin{aligned} Q_e^t(k) &= \max_{0\leq k'\leq k} \sum_{u=k'}^{k} \left(R_{p(t,e)}^t(u) + y_{b(e)}^t(u) - r_e^t(u)\right) \\ &\leq \max_{0\leq k'\leq k} \sum_{u=k'}^{k} \left(r_{p(t,e)}^t(u) + y_{b(e)}^t(u) - r_e^t(u)\right) \\ &= q_e^t(k+1), \end{aligned}$$

where the inequality holds because $R_e^t(k) \leq r_e^t(k)$ for all time slots k.

Since by Lemma 6, $q_e^t(k) < M_q$ for all k, $Q_e^t(k)$ is also bounded above by M_q for every k. ■

VI. Conclusion

In this paper, we have presented a backpressure algorithm for the utility maximization problem for multi-tree multicast. Compared with previous algorithms, our algorithm is not only distributed but local. It is not straightforward to show that our algorithm leads to both primal optimality and network stability due to the assumption of general concave utility functions and the fact that the algorithm relies on virtual queue updates. We have shown that two sets of analytical tools, the Lyapunov optimization technique and the convex optimization techniques, can complement each other and circumvent the difficulty.

References

[1] X. Zheng, C. Cho, and Y. Xia, "Optimal peer-to-peer technique for massive content distribution," in *Proc. IEEE INFOCOM*, vol. 8, 2008, pp. 151–155.

[2] ——, "Content distribution by multiple multicast trees and intersession cooperation: optimal algorithms and approximations," in *Proc. IEEE Conference on Decision and Control*, 2009, pp. 5857–5862.

[3] L. Tassiulas and A. Ephremides, "Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks," *IEEE Transactions on Automatic Control*, vol. 37, no. 12, pp. 1936–1948, 1992.

[4] B. Awerbuch and T. Leighton, "A simple local-control approximation algorithm for multicommodity flow," in *Annual Symposium on Foundations of Computer Science*, vol. 34, 1993, pp. 459–468.

[5] ——, "Improved approximation algorithms for the multi-commodity flow problem and local competitive routing in dynamic networks," in *Proc. ACM Symposium on Theory of Computing*, 1994, pp. 487–496.

[6] X. Lin and N. Shroff, "Joint rate control and scheduling in multihop wireless networks," in *Proc. IEEE Conference on Decision and Control*, 2004, pp. 1484–1489.

[7] M. Neely, E. Modiano, and C. Rohrs, "Dynamic power allocation and routing for time-varying wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 1, pp. 89–103, 2005.

[8] A. Eryilmaz and R. Srikant, "Fair resource allocation in wireless networks using queue-length-based scheduling and congestion control," *IEEE/ACM Transactions on Networking*, vol. 15, no. 6, pp. 1333–1344, 2007.

[9] M. Neely, E. Modiano, and C. Li, "Fairness and optimal stochastic control for heterogeneous networks," *IEEE/ACM Transactions on Networking*, vol. 16, no. 2, pp. 396–409, 2008.

[10] L. Georgiadis, M. Neely, and L. Tassiulas, *Resource allocation and cross layer control in wireless networks*. Now Publishers Inc., 2006.

[11] Y. Cui, Y. Xue, and K. Nahrstedt, "Optimal resource allocation in overlay multicast," *IEEE Transactions on Parallel and Distributed Systems*, vol. 17, no. 8, pp. 808–823, 2006.

[12] M. Chen, M. Ponec, S. Sengupta, J. Li, and P. Chou, "Utility maximization in peer-to-peer systems," in *Proc. ACM SIGMETRICS*, 2008, pp. 169–180.

[13] X. Lin and N. Shroff, "Utility maximization for communication networks with multipath routing," *IEEE Transactions on Automatic Control*, vol. 51, no. 5, pp. 766–781, 2006.

[14] ——, "The impact of imperfect scheduling on cross-layer congestion control in wireless networks," *IEEE/ACM Transactions on Networking*, vol. 14, no. 2, pp. 302–315, 2006.

[15] D. Bertsekas, *Nonlinear programming*, 2nd ed. Athena Scientific, 1999.

[16] K. Kar, S. Sarkar, and L. Tassiulas, "Optimization based rate control for multirate multicast sessions," in *Proc. IEEE INFOCOM*, 2001, pp. 123–132.

[17] Y. Cui, B. Li, and K. Nahrstedt, "On achieving optimized capacity utilization in application overlay networks with multiple competing sessions," in *Proc. ACM Symposium on Parallelism in Algorithms and Architectures*, 2004, pp. 160–169.

[18] L. Bui, R. Srikant, and A. Stolyar, "Optimal resource allocation for multicast sessions in multi-hop wireless networks," *Philosophical Ttransactions of the Royal Society A*, vol. 366, no. 1872, pp. 2059–2074, 2008.