Scheduling Multicast Traffic with Deadlines in Wireless Networks

Kyu Seob Kim, Chih-ping Li, and Eytan Modiano

Laboratory for Information and Decision Systems
Massachusetts Institute of Technology

***Abstract*—We consider the problem of transmitting multicast flows with hard deadlines over unreliable wireless channels. Every user in the network subscribes to several multicast flows, and requires a minimum throughput for each subscribed flow to meet the QoS constraints. The network controller schedules the transmissions of multicast traffic based on the instant feedback from the users. We characterize the multicast throughput region by analyzing its boundary points, each of which is the solution to a finite-horizon dynamic programming problem over an exponentially large state space. Using backward induction and interchange arguments, we show that the dynamic programming problems are solved by greedy policies that maximize the immediate weighted sum throughput in every slot. Furthermore, we develop a dynamic throughput-optimal policy that achieves any feasible throughput vector by tracking the running performance received by the users.**

I. Introduction

The problem of scheduling real-time transmissions in a wireless network has numerous applications, e.g., wireless users download popular video streams over cellular networks. For a video stream that is accessed by multiple users, it is efficient for the base station to deliver the stream as a multicast flow to a group of users. Transmitting multicast real-time flows in wireless networks faces several challenges. Packets of real-time traffic have hard delay constraints and are of no use after the deadlines. Wireless channels are unreliable so that packet loss is inevitable and retransmissions are necessary. Based on individual QoS constraints, the users may also have different throughput requirements for a multicast flow. An efficient wireless scheduling algorithm for multicast communications must take these issues into account.

In this paper, we study the problem of transmitting multicast flows with deadlines in wireless networks, with the assumption that there is instant ACK/NACK feedback from the users to the base station. We adopt the analytical framework in [1]. Consider a base station transmitting multicast flows to users over unreliable wireless channels. We assume a time-slotted system. Each multicast flow generates a packet periodically with a hard delay constraint; packets that violate the delay constraints are of no use and discarded immediately. In every slot, the base station selects a multicast flow to transmit, based on the information of the subset of subscribers having not received the packets for all multicast flows. Since wireless channels are unreliable, packets transmissions may fail, and the users notify the base station whether transmissions are successful at the end of every slot. The performance metric of interest is the proportion of packets a user successfully receives for each subscribed multicast flow, referred to as the throughput of the user for that flow. According to the QoS constraints, a user has a minimum throughput requirement for each multicast flow to which it subscribes. The goal of this paper is to study the set of achievable throughput vectors in this multicast setting, and develop adaptive control policies that support any feasible throughput vector.

This multicast scheduling problem is studied in [1], with the key assumption that users do not send any feedback to the base station. In a large wireless network, collecting feedback from the users may be infeasible. However, in a network of moderate size, feedback can be readily available and a scheduling policy can utilize such information to improve the network performance. The difficulty in making scheduling decisions based on the user feedback is that it is a dynamic programming problem with an exponentially large state space, where the base station decides which multicast flow to transmit according to the subset of users having not received the packets in every slot. Generally in such problems it may not be possible to characterize the throughput region, or develop practical control policies, due to the curse of dimensionality. In this paper, we show that the multicast scheduling problem is, in fact, solvable by a greedy algorithm. Our main contributions include:

- We characterize the multicast throughput region with user feedback by analyzing its boundary points, each of which corresponds to a weighted sum throughput maximization problem, which is formulated as a finite-horizon dynamic programming problem.
- Using backward induction and interchange arguments, we show that the weighted sum throughput maximization problem is solved by greedy policies that transmit the multicast flow maximizing the immediate weighted sum reward in every slot. For the ease of exposition, we first analyze the special case of transmitting unicast flows, and generalize it to the multicast case.

This work is supported by NSF grant CNS-1217048, ONR grant N00014-12-1-0064, ARO MURI grant W911NF-08-1-0238, and by a grant from the MIT/Masdar Institute Cooperative Program.

978-14799-3360-0/14/$31.00 ©2014 IEEE

- We develop adaptive throughput-optimal control policies that achieve any feasible throughput vector in the multicast throughput region. This policy greedily transmits a multicast flow in a slot by striking a balance between serving a user with a large delivery debt needed to satisfy its throughput requirement, and transmitting a multicast flow that yields large throughput gain from among the unserved subscribers.

Scheduling multicast flows with deadlines over unreliable wireless channels is considered in [1]–[3]. Work [1] assumes homogeneous packet deadlines and that the user feedback is unavailable, in which case the multicast throughput region is decided by the number of slots allocated to each multicast flow. Prior work [2] assumes heterogenous packet deadlines and instant user feedback, and the multicast scheduling problem is an instance of restless multi-armed bandits [4]. The problem of transmitting unicast traffic with deadlines in wireless networks is investigated with different modeling assumptions [5]–[11]. Scheduling packet transmissions with deadlines over multi-hop networks is considered in [12]–[14].

The outline of this paper is as follows. Section II describes the network model. The unicast and multicast throughput regions are analyzed in Sections III and IV, respectively. The throughput-optimal control policy is proposed in Section V, followed by simulation results in Section VI.

II. Network Model

We consider a wireless network consisting of a base station sending multicast flows to N users $\{1,\ldots,N\}$. Let F be the set of multicast flows. For each flow $f \in F$, let N_f be the subset of users that subscribe to multicast flow f; a user may subscribe to several flows simultaneously. We consider a time-slotted system, where consecutive T slots constitute a frame; the kth frame is $[kT,(k+1)T)$, $k \in \mathbb{Z}^+$. Each multicast flow generates one packet at the beginning of a frame.[1] Suppose all packets have a hard deadline of T slots, and packets that have not been received by all of its intended receivers at the end of a frame are of no use and discarded. In a slot, the base station broadcasts a packet of a selected multicast flow to its subscribers over unreliable and independent wireless channels. The transmission of a flow-f packet to a user $n \in N_f$ is successful with probability p_n, and fails otherwise. When a user receives a packet successfully, it sends an acknowledgement to the base station over a perfect control channel at the end of the slot. The base station uses the ACK/NACK feedback information to schedule future packet transmissions.

Without loss of generality, we can simplify the above network model by assuming that each user subscribes to only one multicast flow. As an example, consider a user subscribing to two multicast flows. By defining two "agents" for the user, where agent $i \in \{1,2\}$ is the only receiver for flow i, transmitting two multicast flows to the user is equivalent to scheduling the multicast transmissions to the two agents. See Fig. 1 for an example. Note that the two agents should have the same channel process as the user. Since the base station can only transmit one multicast flow in a slot, at most one agent receives data at any time. Therefore, it is adequate to assume that the agents have independent wireless channels that have the same packet erasure probability.

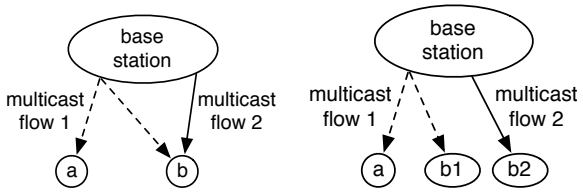

Fig. 1. The multicast system with a user subscribing to multiple flows on the left is equivalent to the one on the right in which each user subscribes to one flow.

Due to the unreliability of wireless channels and the hard deadline constraints, not all packets can be delivered to the users in a timely manner. Since real-time applications can usually tolerate some percentage of packet loss, the performance metric of interest in this paper is the long-term proportion of packets a user receives before deadlines, referred to as the user n throughput. Define the indicator random variable $d_n(k)=1$ if user n successfully receives a packet in the kth frame, and 0 otherwise. Define the user n throughput as

$$d_n = \liminf_{K\to\infty} \frac{1}{K}\sum_{k=0}^{K-1} \mathbb{E}[d_n(k)].$$

Since a user receives only one multicast flow, the index n implicitly specifies which multicast flow the user n subscribes to.

III. Achievable Throughput Region

We consider the class of scheduling policies that satisfy the following properties: (i) they do not use future information; (ii) they never idle the base station whenever there exists a packet not received by all its subscribers by the deadline; (iii) if all packets are delivered before the deadline, then the base station remains idle until the end of the frame. We refer to these scheduling policies as admissible policies. We define the throughput region Λ as the set of achievable throughput vectors $(d_1,\ldots,d_N)$ under the class Π of admissible policies.

To characterize the throughput region Λ, it suffices to focus on the set Λ_0 of achievable throughput vectors in the first frame $[0,T)$ under admissible policies. This is because the multicast system renews itself at the end of a frame, and therefore $\Lambda=\Lambda_0$. An admissible policy $\pi \in \Pi$ in the first frame transmits a multicast flow in the kth slot, $k \le T-1$, according to the outcome of scheduling decisions in the previous slots $0,1,\ldots,k-1$. Naturally, it is a finite-horizon dynamic programming problem, where the state space is the collection of subsets of users having not received packets in the first frame, and the action space is the set of multicast flows for

[1]The results in this paper can be easily generalized to a stochastic traffic model, or to a traffic model in which each flow generates multiple packets per frame. The latter case is equivalent to replacing a multicast flow by a group of subflows, each of which generates one packet per frame and has the same set of subscribers.

transmission. Therefore, it is difficult to obtain the multicast throughput region Λ_0 in closed form. We take an alternative approach that studies the throughput region Λ_0 by analyzing its boundary points. Each boundary point corresponds to a weighted sum throughput maximization problem over the course of the first frame. Specifically, let $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_N)$ be a nonnegative weight vector, and consider the optimization problem:

$$\text{maximize} \sum_{n=1}^{N} \alpha_n \, \mathbb{E}[d_n^{\pi}(0)], \quad \text{subject to } \pi \in \Pi, \qquad (1)$$

where $\mathbb{E}[d_n^{\pi}(0)]$ denotes the expected user n throughput in the first frame (i.e., frame zero) under policy π. The solution to the problem (1) specifies a boundary point at the intersection of the throughput region Λ_0 and the supporting hyperplane that has the normal vector $\boldsymbol{\alpha}$ and is tangent to Λ_0. See Fig. 2 for an example.

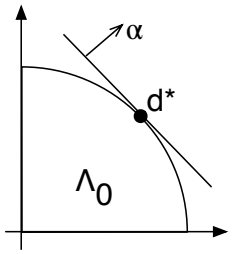

Fig. 2. A boundary point $\boldsymbol{d}^*$ at the intersection of the throughput region Λ and the maximizing supporting hyperplane with a normal vector $\boldsymbol{\alpha} \geq 0$. The vector $\boldsymbol{d}^*$ is the solution to the maximization problem (1) with weights $\boldsymbol{\alpha}$.

Next, we consider the throughput maximization problem (1) as a finite-horizon dynamic programming problem, and show that it is solved by a greedy policy that maximizes weighted immediate sum throughput in very slot. For the ease of exposition, we first analyze the unicast case, i.e., each multicast flow is subscribed by only one user. This approach is generalized to the analysis of the multicast throughput region in Section IV.

A. The Unicast throughput region

Let $\Lambda_0^{\text{unicast}}$ be the set of feasible throughput vectors in the first frame under admissible policies, assuming the base station serves unicast flows. The region $\Lambda_0^{\text{unicast}}$ is known to be the base of a polymatroid up to a linear scaling [15]. As a result, the weighted sum throughput maximization problem (1) in the special case of serving unicast flows becomes the following problem

$$\begin{aligned} \text{maximize} \quad & \sum_{n=1}^{N} \alpha_n \, \mathbb{E}[d_n^{\pi}(0)] \\ \text{subject to} \quad & (\mathbb{E}[d_n^{\pi}(0)])_{n=1}^{N} \in \Lambda_0^{\text{unicast}}. \end{aligned}$$

This maximization problem is an instance of linear polymatroid optimization [16], and is solved by the next greedy policy.

The Greedy Policy for a Unicast System (GreedyU)

1) In slot $k \in \{0, 1, \ldots, T-1\}$, transmit the unicast flow

$$j_k^* \in \operatorname{argmax}_{j \in s_k} \alpha_j \, p_j, \qquad (2)$$

where $s_k \neq \emptyset$ is the subset of users having not received the packets at the beginning of slot k. If $s_k = \emptyset$ then idle the system in slot k.

The GreedyU policy serves the user with the maximum expected weighted throughput $\alpha_j \, p_j$ in every slot. This policy is also known as the $c\mu$ rule in polymatroid optimization, and its optimality can be shown by using the complimentary slackness condition of linear programming [16]. This result, however, cannot be directly applied to the analysis of the multicast throughput region, which is not a polymatroid. Next, we present an alternative proof for the optimality of the GreedyU policy using dynamic programming methods. This new approach is needed to analyze the multicast throughput region as a generalization of the unicast throughput region.

B. Dynamic programming algorithm

Consider a finite-horizon dynamic programming problem with the state space $S = 2^{\{1,\ldots,N\}}$, which is the collection of all subsets of users. Each state $s_k \in S$ represents the subset of users having not received their packets at the beginning of the kth slot, $k = 0, \ldots, T-1$. Initially, we have $s_0 = \{1, \ldots, N\}$. Let u_k be the control action taken in slot k; note that u_k is possibly random and is a function of the system state s_k. An admissible policy $\pi \in \Pi$ in the first frame can be re-written as $\pi = (u_0, u_1, \ldots, u_{T-1})$. At state s_k, the set of feasible scheduling decisions is $U_k(s_k) = \{j \mid j \in s_k\}$, where $u_k(s_k) = j$ means transmitting multicast flow j in slot k. The feasible set $U_k(s_k)$ comes from the assumption that, under an admissible policy, a unicast flow can be transmitted only if its subscriber has not received the packet. Since wireless channels are unreliable, taking action $u_k(s_k) = j \in U_k(s_k)$ at state s_k leads to two possible outcomes in slot $(k+1)$: (i) we have the state $s_{k+1} = s_k \setminus \{j\}$ if the transmission succeeds, which occurs with probability p_j; (ii) we have $s_{k+1} = s_k$ if the transmission fails. User j obtains a reward $\alpha_j \geq 0$ if it receives a packet. Define the reward function g_k under the control $u_k(s_k) = j$ in the kth slot:

$$g_k(s_k, u_k) = g_k(s_k, j) = \alpha_j X_j, \quad j \in U_k(s_k),$$

where X_j is a Bernoulli random variable with mean p_j. When $s_k = \emptyset$, let $g_k(s_k, \cdot) = 0$ for all k. We seek to solve the reward maximization problem:

$$J^*(s_0) \triangleq \max_{\pi \in \Pi} \mathbb{E}\left\{ \sum_{k=0}^{T-1} g_k(s_k, u_k) \,\middle|\, s_0 \right\}, \qquad (3)$$

where the expectation is with respect to the randomness of wireless channels and scheduling decisions. We observe that the dynamic programming problem (3) is equivalent to the maximization problem (1) restricted to the unicast case.

The problem (3) is solved by a dynamic programming algorithm as follows [17]. Define the functions

$$J_T(s_T) = 0, \quad s_T \subseteq S, \tag{4}$$

$$J_k(s_k) = \max_{u_k \in U_k(s_k)} \mathbb{E}\{\, g_k(s_k, u_k) + J_{k+1}(s_{k+1}) \,\}, \tag{5}$$

for $k = 0, \ldots, T-1$, where J_k can be computed backwards after $J_{k+1}, \ldots, J_{T-1}$ are calculated. We have the following results [17, Prop. 1.3.1].

1) The maximization problem (3) is solved by $J^*(s_0) = J_0(s_0)$.
2) If the scheduling decision u_k^*, which is a function of s_k, is the maximizer of $J_k(s_k)$ in (5) for each s_k and k, then the policy $\pi^* = (u_0^*, u_1^*, \ldots, u_{T-1}^*)$ is optimal and solves (3).

C. *Optimality of the* GreedyU *policy*

Based on the above dynamic programming algorithm, the next theorem shows that the GreedyU policy solves the reward maximization problem (3).

Theorem 1. Given a fixed nonnegative weight vector $(\alpha_1, \ldots, \alpha_N)$, the GreedyU policy solves the reward maximization problem (3), which is a special case of weighted sum throughput maximization problem (1) in the special case of serving unicast flows.

Proof: See Appendix A. ∎

Using the GreedyU policy, we draw the unicast throughput region $\Lambda = \Lambda_0$ in a two-user wireless network. Consider a base station serving two users. The channel reliability probabilities are $p_1 = p_2 = 0.2$, and the frame size is $T = 5$. Fig. 3 shows the unicast throughput region by computing a collection of supporting hyperplanes, each of which intersects the region Λ at a maximum weighted sum throughput vector computed by the GreedyU policy.

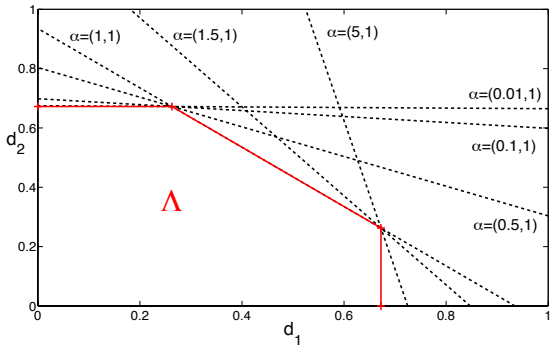

Fig. 3. The unicast throughput region in a two-user wireless network.

IV. The Multicast Throughput Region

Next we consider the scheduling problem of transmitting multicast flows in the wireless network. Similarly, we characterize the multicast throughput region by analyzing its boundary points, each of which is the solution to the maximization problem (1) with some weight vector $\boldsymbol{\alpha} \geq 0$.

To solve the maximization problem (1), we let $U_f(t)$ be the subset of users that subscribe to multicast flow f but have not received the packet from f at the beginning of slot t in the the first frame, $0 \leq t \leq T-1$. Since no users receive packets at the beginning of a frame, we have $U_f(0) = N_f$ for all flows $f \in F$. We consider the following greedy policy.

The Greedy Policy for a Multicast System (GreedyM)

1) Fix a nonnegative weight vector $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_N)$.
2) In slot $t \in \{0, \ldots, T-1\}$, compute

$$r_f(t) = \sum_{i \in U_f(t)} \alpha_i \, p_i \tag{6}$$

for each flow $f \in F$; let $r_f(t) = 0$ if $U_f(t) = \emptyset$.

3) In slot t, pick the multicast flow

$$f^* \in \operatorname{argmax}_{f \in F} r_f(t).$$

If $r_{f^*}(t) > 0$, then broadcast a flow-f^* packet. Otherwise, idle the system in slot t.

The quantity $r_f(t)$ is the expected weighted sum throughput in slot t by transmitting a flow-f packet to the subset $U_f(t)$ of subscribers having not received the packet. Therefore, the GreedyM policy transmits the multicast flow that yields the maximum throughput gain in every slot. By using similar backward induction and interchange arguments as those used to prove Theorem 1, the next theorem shows that the GreedyM policy solves the maximization problem (1) in the multicast system.

Theorem 2. Given a fixed nonnegative weight vector $(\alpha_1, \ldots, \alpha_N)$, the GreedyM policy solves the throughput maximization problem (1) in the multicast system.

Proof: See Appendix B. ∎

V. Throughput-Optimal Policy

Consider the problem that each user n has a minimum long-term throughput requirement $q_n \geq 0$ for its multicast flow. Assume $(q_1, \ldots, q_N) \in \Lambda$ is feasible. We develop a throughput-optimal policy that achieves any feasible throughput vector $(q_1, \ldots, q_N)$ in the multicast throughput region Λ.

Our policy is an adaptive control policy that keeps track of the average throughput of the users, and seeks to optimize the following tradeoff: whether to transmit packets to a user whose running throughput severely violates the delivery requirement, or transmit a multicast flow that has not been received by many subscribers? Specifically, define the indicator function $d_n(k) = 1$ if user n receives the packet from its subscribed flow in the kth frame, or 0 otherwise. At the beginning of the kth frame, define

$$D_n(k) = \sum_{j=0}^{k-1} \big(q_n - d_n(j)\big), \tag{7}$$

as the amount of accumulated throughput the base station owes to user n to achieve its throughput requirement q_n after the first

k frames. From (7) we have $D_n(k+1) = D_n(k)+q_n-d_n(k)$. Let $D_n(0) = 0$ for all n. Define $D_n^+(k) = \max\{D_n(k), 0\}$ as the delivery debt the base station owes to user n; the debt is zero if a user receives more than it should, i.e., $D_n(k) < 0$. We consider the following policy.

Frame-Based Max-Weight Policy (MW)

1) In the kth frame, apply the admissible policy $\pi \in \Pi$ that maximizes the weighted sum throughput $\sum_{n=1}^N D_n^+(k) \cdot \mathbb{E}[d_n(k)]$ in the frame.

Note that the expectation $\mathbb{E}[d_n(k)]$ is with respect to both unreliable channels and control decisions. We observe from the term $\sum_{n=1}^N D_n^+(k) \cdot \mathbb{E}[d_n(k)]$ that the MW policy seeks to improve the weighted sum of delivery debts owed to the users in every frame. Next, the maximization problem in the MW policy is equivalent to maximizing the expected weighted sum throughput in a frame with the weight $\alpha_n = D_n^+(k)$ for user n in the maximization problem (1). From the GreedyM policy in Section IV, the MW policy becomes the following policy.

Frame-Based Throughput-Optimal Policy (TO)

1) In slot t of the kth frame, observe the subsets $\{U_f(t)\}_{f\in F}$ of unserved users and compute

$$r_f(t) = \sum_{n\in U_f(t)} D_n^+(k) \cdot p_n$$

for each multicast flow $f \in F$.

2) Transmit the multicast flow $f^* \in \operatorname{argmax}_{f\in F} r_f(t)$ in slot t if $r_{f^*}(t) > 0$; otherwise, idle the system.

The sum $\sum_{n\in U_f(t)} p_n$ is the throughput gain of transmitting multicast flow f in slot t. Therefore, the GreedyM policy serves the multicast flow f^* that maximizes $r_f(t)$ in order to strike a balance between maximizing immediate throughput gain and reducing the delivery debts $D_n^+(k)$ owed to the users. The next theorem shows that the MW and the TO policies are throughput optimal.

Theorem 3. The MW policy (i.e., the TO policy) is throughput optimal in the multicast system. That is, the MW policy achieves any feasible throughput vector $(q_1, \ldots, q_N)$ in the multicast throughput region Λ.

Proof: See Appendix C. ∎

VI. Simulation Results

We study via simulations how the user feedback affects the throughput performance of multicast transmissions in unreliable wireless networks. This is compared to the prior work that studies the problem of scheduling multicast flows with deadlines, assuming the user feedback is unavailable [1].

Let $\mathsf{EWST}_{\text{nofb}}(\boldsymbol{\alpha})$ be the maximum expected weighted sum throughput of multicast transmissions without the user feedback, where $\boldsymbol{\alpha} \geq 0$ is a weight vector. In this case, the expected throughput of a user is decided by the number of slots the user is served. Let τ_f be the number of slots flow f is transmitted in a frame. The expected throughput of a user n subscribing to flow f in a frame is the probability that the user successfully receives the packet at least once; the probability is $(1-(1-p_n)^{\tau_f})$. As a result, we have

$$\mathsf{EWST}_{\text{nofb}}(\boldsymbol{\alpha}) = \max_{\sum_{f\in F}\tau_f = T} \sum_{f\in F}\sum_{n\in N_f} \alpha_n(1-(1-p_n)^{\tau_f}).$$

Notice that since packets are transmitted blindly, the weighted sum throughput is maximized by using every slot of the frame; hence the constraint $\sum_{f\in F}\tau_f = T$. An algorithm is provided to solve this maximization problem in [1]. Let $\mathsf{EWST}_{\text{fb}}(\boldsymbol{\alpha})$ be the maximum expected weighted sum throughput with the user feedback. The GreedyM policy in this paper achieves $\mathsf{EWST}_{\text{fb}}(\boldsymbol{\alpha})$ by Theorem 2. Fig. 4 shows the normalized values of $\mathsf{EWST}_{\text{fb}}(\boldsymbol{\alpha})$ and $\mathsf{EWST}_{\text{nofb}}(\boldsymbol{\alpha})$ under different weight vectors, confirming that the user feedback improves maximum system throughput.

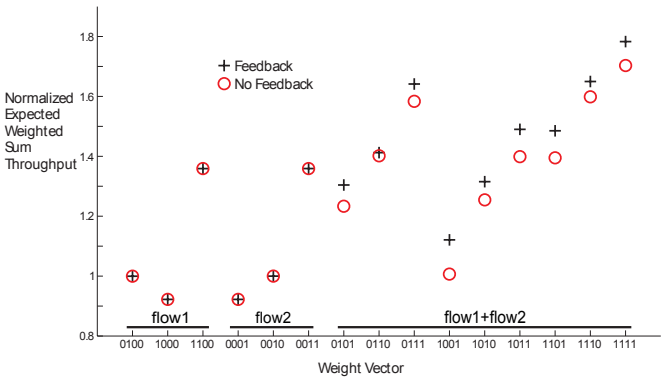

Fig. 4. The weighted sum throughput using instant feedback outperforms that without feedback. The setup is as follows. Frame size $T = 5$. There are two multicast flows. Users 1 and 2 subscribe to flow 1 with $(p_1, p_2) = (0.4, 0.8)$, and users 3 and 4 subscribe to flow 2 with $(p_3, p_4) = (0.9, 0.4)$. The x-axis represents different weight vectors $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \alpha_3, \alpha_4) \subseteq \{0, 1\}^4$. The y-axis represents the normalized expected weighed sum throughput $\mathsf{EWST}_{\text{fb}}(\boldsymbol{\alpha})/\|\boldsymbol{\alpha}\|$ and $\mathsf{EWST}_{\text{nofb}}(\boldsymbol{\alpha})/\|\boldsymbol{\alpha}\|$. There is no throughput improvement for the first six $\boldsymbol{\alpha}$ vectors because they correspond to the cases where only one flow has positive weights.

A. The popularity of a multicast flow

We examine how the number of subscribers to a multicast flow affects the throughput performance. Consider two flows having the same number of subscribers. All subscribers are assumed to have the same channel reliability probability p. We consider the following performance metric:

$$\frac{\mathsf{EWST}_{\text{fb}}(\mathbf{1}) - \mathsf{EWST}_{\text{nofb}}(\mathbf{1})}{\mathsf{EWST}_{\text{fb}}(\mathbf{1})} \times 100\%, \tag{8}$$

which is the normalized throughput gain from the feedback information. Fig. 5 shows that the metric (8) increases as the number of users per flow decreases for different values of channel reliability p. That is, the user feedback provides more throughput gain when a multicast flow becomes less popular. One cause of throughput loss in the no-feedback case is that

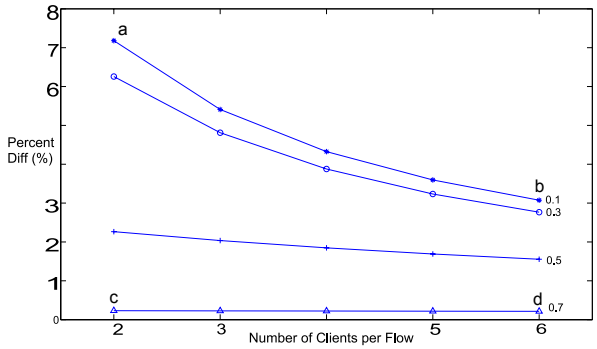

Fig. 5. The normalized throughput gain of feedback-aware transmissions as a function of channel reliability probability and the number of subscribers per flow. Frame size $T = 10$.

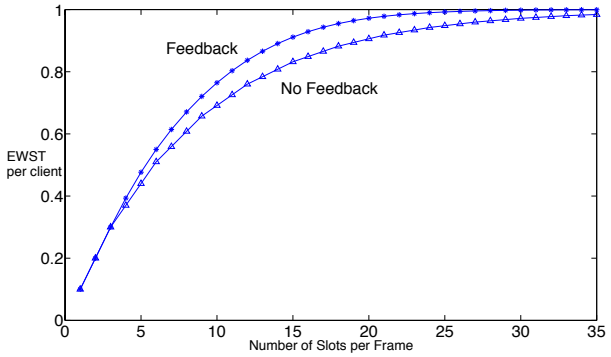

Fig. 6. The per-user throughput gain of feedback-aware multicast transmissions as a function of the frame size. A multicast network with three flows and two subscribers per flow is considered.

the base station retransmits a flow for which all subscribers have received the packet. This event occurs during the first retransmission of a flow f in a frame with probability $p^{|N_f|}$. This implies that, when the number $|N_f|$ of users per flow decreases, it is more likely that a blind retransmission results in no throughput gain, which contributes to the throughput gap between the two cases.

The points c and d in Fig. 5 show that, when channel conditions are good, there is little throughput loss of blind transmissions. This is because the frame size is large enough so that eventually most of the packets will be delivered. The point a is where feedback-aware multicast transmissions have the largest throughput gain. Here, the channel reliability probability p is low so that the feedback-aware transmissions need the entire frame for packet delivery, leaving few idle slots in the frame probabilistically. Since the only way for the throughput performance of blind transmissions to be close to that of feedback-aware transmissions is to benefit from retransmissions during the idle slots unused by feedback-aware transmissions, there is no room for blind transmissions to improve throughput in this case.

B. Frame size

We examine how the size of a frame affects the throughput performance. Consider three multicast flows, each of which has two subscribers. Every user has the same channel reliability probability $p = 0.3$. Fig. 6 shows the per-user throughput under both with-feedback and no-feedback cases for different frame sizes. When the frame size is small, the throughput is the same because we would expect that blind transmissions in a round robin fashion yield the maximum throughput in the symmetric case. When the frame size is large, the probability of a user not receiving the packet at the end of a frame is negligible, and thus blind transmissions incur no throughput loss. For other frame sizes, we observe that multicast transmissions with feedback outperform those without feedback, as expected.

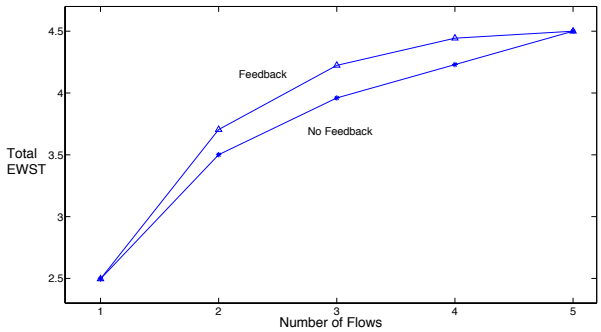

Fig. 7. The throughput gain of feedback-aware transmissions as a function of the number of multicast flows in the network.

C. The number of multicast flows

We study the throughput gain of feedback-aware transmissions as the number of multicast flows varies. We assume that every user has channel reliably probability $p = 0.3$, and that the frame size is $T = 5$. Each multicast flow has three subscribers. Fig. 7 shows the system throughput in both with-feedback and no-feedback cases. When there is only one flow, blindly serving the flow in every slot is throughput optimal. When the number of flows is sufficiently large, e.g., being equal to the frame size, then blindly serving a different flow in each slot is as good as feedback-aware transmissions under the symmetric scenario. In other cases, we expect that scheduling multicast flows using feedback improves throughput over that without using feedback.

VII. Conclusion

This paper studies the multicast scheduling problem for transmitting periodically generated traffic with hard deadlines over unreliable wireless channels. We study the set of achievable timely throughput vectors by analyzing the boundary of the multicast throughput region. Using backward induction and interchange arguments in dynamic programming, we show that the boundary points are achieved by greedy policies that seek to maximize immediate weighted sum throughput in every slot.

We utilize this greedy policy to design an adaptive throughput optimal policy over the multicast throughput region. This policy optimizes the tradeoff between serving a user that has not received enough packets to satisfy its QoS constraint, and transmitting a multicast flow that yields large instantaneous throughput gain from its subscribers. Through simulations, we identify conditions under which feedback-aware multicast transmissions have the largest, or the lowest, throughput gain over those that do not use feedback.

REFERENCES

[1] I. Hou and P. R. Kumar, "Broadcasting delay-constrained traffic over unreliable wireless links with network coding," in *ACM Int. Symp. Mobile Ad Hoc Networking and Computing (MobiHoc)*, 2011.

[2] V. Raghunathan, V. Borkar, M. Cao, and P. R. Kumar, "Index policies for real-time multicast scheduling for wireless broadcast systems," in *IEEE Proc. INFOCOM*, Apr. 2008, pp. 1570–1578.

[3] X. Li, C.-C. Wang, and X. Lin, "Throughput and delay analysis on uncoded and coded wireless broadcast with hard deadline constraints," in *IEEE Proc. INFOCOM*, 2010.

[4] P. Whittle, "Restless bandits: Activity allocation in a changing world," *J. Appl. Probab.*, vol. 25, pp. 287–298, 1988.

[5] I. Hou, V. Borkar, and P. Kumar, "A theory of QoS for wireless," in *IEEE Proc. INFOCOM*, Apr. 2009, pp. 486–494.

[6] I. Hou and P. R. Kumar, "Queueing systems with hard delay constraints: a framework for real-time communication over unreliable wireless channels," *Queueing Syst.*, vol. 71, pp. 151–177, 2012.

[7] ——, "Scheduling heterogeneous real-time traffic over fading wireless channels," in *IEEE Proc. INFOCOM*, 2010.

[8] ——, "Admission control and scheduling for QoS guarantees for variable-bit-rate applications on wireless channels," in *ACM Int. Symp. Mobile Ad Hoc Networking and Computing (MobiHoc)*, 2009.

[9] J. Jaramillo, R. Srikant, and L. Ying, "Scheduling for optimal rate allocation in ad hoc networks with heterogeneous delay constraints," *IEEE J. Sel. Areas Commun.*, vol. 29, no. 5, pp. 979–987, May 2011.

[10] A. Dua and N. Bambos, "Downlink wireless packet scheduling with deadlines," *IEEE Trans. Mobile Comput.*, vol. 6, no. 12, pp. 1410–1425, Dec. 2007.

[11] S. Shakkottai and R. Srikant, "Scheduling real-time traffic with deadlines over a wireless channel," *Wireless Networks*, vol. 8, no. 1, pp. 13–26, Jan. 2002.

[12] Z. Mao, C. E. Koksal, and N. B. Shroff, "Online packet scheduling with hard deadlines in multihop networks," in *IEEE Proc. INFOCOM*, 2013.

[13] R. Li and A. Eryilmaz, "Scheduling for end-to-end deadline-constrained traffic with reliability requirements in multi-hop networks," in *IEEE Proc. INFOCOM*, 2011.

[14] I. Hou, "Providing end-to-end delay guarantees for multi-hop wireless sensor networks over unreliable channels," arXiv report. [Online]. Available: http://arxiv.org/abs/1204.4465

[15] I. Hou, A. Truong, S. Chakraborty, and P. R. Kumar, "Optimality of periodwise static priority policies in real-time communications," in *IEEE Conf. Decision and Control (CDC)*, 2011.

[16] D. D. Yao, "Dynamic scheduling via polymatroid optimization," in *Performance Evaluation of Complex Systems: Techniques and Tools, Performance 2002, Tutorial Lectures*. London, UK: Springer-Verlag, 2002, pp. 89–113.

[17] D. P. Bertsekas, *Dynamic Programming and Optimal Control*, 3rd ed. Athena Scientific, 2005, vol. I.

APPENDIX A
PROOF OF THEOREM 1

Applying the dynamic programming algorithm in Section III-B to the unicast scheduling problem, (5) becomes

$$J_k(s_k) = \max_{j\in s_k} \Big\{ \alpha_j p_j + p_j J_{k+1}(s_k \setminus \{j\}) + (1-p_j) J_{k+1}(s_k) \Big\} \quad (9)$$

if $s_k \neq \emptyset$ and $J_k(s_k) = 0$ otherwise. The optimal policy $\pi^* = (u_0^*, u_1^*, \ldots, u_{T-1}^*)$ satisfies

$$u_k^*(s_k) \in \operatorname{argmax}_{j\in s_k} \Big\{ \alpha_j p_j + p_j J_{k+1}(s_k \setminus \{j\}) + (1-p_j) J_{k+1}(s_k) \Big\}. \quad (10)$$

To show that the greedy decisions j_k^* in (2) are optimal, it suffices to show that $j_k^* = u_k^*(s_k)$ for all s_k and k. We establish this result by backward induction and interchange arguments. It is useful to define the cost-to-go function under a policy $\pi = (u_0, u_1, \ldots, u_{T-1})$:

$$J^\pi(s,k) = \mathbb{E}\left\{ \sum_{i=k}^{T-1} g_i(s_i, u_i) \;\middle|\; s_k = s \right\}, \; s \in S.$$

Consider the following four cases.

(i) In any slot k, if $s_k = \emptyset$ then all users are served and the system stays idle. If $s_k = \{j\}$ for some j, then we have $u_k^*(s_k) = j_k^* = j$ and the base station must transmit the flow j because there is one unserved flow left. It remains to discuss the case $|s_k| \geq 2$ in each slot k.

(ii) Consider the tail subproblem that we seek to maximize the cost-to-go function in the last slot $[T-1, T)$ of the frame. In this slot, from (9) we have $J_{T-1}(s_{T-1}) = \max_{j\in s_{T-1}}\{\alpha_j\, p_j\}$ because $J_T(s_T) = 0$ for all s_T. Therefore, $u_{T-1}^*(s_{T-1}) = j_{T-1}^*$ for all s_{T-1}.

(iii) Consider the tail subproblem where the system is at state s_{T-2} and we maximize the cost-to-go function $J^{(\pi)}(s_{T-2}, T-2)$ over policies $\pi \in \Pi$ in the last two slots of the frame. Consider the two policies: $\pi_1 = (\ldots, u_{T-2}^{(1)}, u_{T-1}^*)$ and $\pi_2 = (\ldots, u_{T-2}^{(2)}, u_{T-1}^*)$, where

$$u_{T-2}^{(1)}(s_{T-2}) = k_1 \triangleq \operatorname{argmax}_{j\in s_{T-2}} \alpha_j\, p_j,$$
$$u_{T-2}^{(2)}(s_{T-2}) = l, \quad l \neq k_1.$$

In other words, both policies adopt the optimal action in slot $(T-1)$. But in slot $[T-2, T)$, policy π_1 uses the greedy decision while policy π_2 uses any other decision in slot $(T-2)$. Define $k_2 = \operatorname{argmax}_{j\in s_{T-2}\setminus\{k_1\}} \alpha_j\, p_j$. It follows that

$$J^{(\pi_1)}(s_{T-2}, T-2) = \alpha_{k_1} p_{k_1} + p_{k_1}\alpha_{k_2} p_{k_2} + (1-p_{k_1})\alpha_{k_1} p_{k_1},$$
$$J^{(\pi_2)}(s_{T-2}, T-2) = \alpha_l p_l + p_l \alpha_{k_1} p_{k_1} + (1-p_l)\alpha_{k_1} p_{k_1} = \alpha_l p_l + \alpha_{k_1} p_{k_1}.$$

By definition, we have $\alpha_{k_1} p_{k_1} \geq \alpha_{k_2} p_{k_2} \geq \alpha_l p_l$. We have

$$\begin{aligned} J^{(\pi_1)}(s_{T-2}, T-2) &= \alpha_{k_1} p_{k_1} + p_{k_1}\alpha_{k_2} p_{k_2} + (1-p_{k_1})\alpha_{k_1} p_{k_1} \\ &\geq \alpha_{k_1} p_{k_1} + p_{k_1}\alpha_l p_l + (1-p_{k_1})\alpha_l p_l \\ &= \alpha_{k_1} p_{k_1} + \alpha_l p_l \\ &= J^{(\pi_2)}(s_{T-2}, T-2). \end{aligned}$$

That is, $J^{(\pi_1)}(s_{T-2}, T-2) \geq J^{(\pi_2)}(s_{T-2}, T-2)$ for any $l \neq k_1$. Therefore, $u_k^*(s_{T-2}) = \operatorname{argmax}_{j\in s_{T-2}} \alpha_j\, p_j$.

(iv) To prove by backward induction, given $0 < t \leq T-1$, assume that the optimal policy at state s_k is $u_k^*(s_k) =$

$\text{argmax}_{j\in s_k}\,\alpha_j p_j$ for all $k \geq t$. We consider the tail subproblem where the system is at state s_{t-1} and we maximize the cost-to-go $J^{(\pi)}(s_{t-1}, t-1)$ during slots in $[t-1, T)$. Consider the two policies: $\pi_1 = (\ldots, u_{t-1}^{(1)}, u_t^*, \ldots, u_{T-1}^*)$ and $\pi_2 = (\ldots, u_{t-1}^{(2)}, u_t^*, \ldots, u_{T-1}^*)$, where

$$u_{t-1}^{(1)}(s_{t-1}) = k_1 \triangleq \text{argmax}_{j\in s_{t-1}}\,\alpha_j\, p_j,$$
$$u_{t-1}^{(2)}(s_{t-1}) = l, \quad l \neq k_1.$$

In other words, policy π_1 uses the greedy decision in slot $t-1$ and optimal decisions afterwards. Policy π_2 serves another user $l \neq k_1$ in slot $t-1$ and uses optimal decisions afterwards. Now, given the policies π_1 and π_2, we define a third policy $\pi_3 = (\ldots, u_{t-1}^{(3)}, u_t^{(3)}, u_{t+1}^*, \ldots, u_{T-1}^*)$ working as follows: (i) greedily serve user k_1 in slot $t-1$ according to π_1; (ii) always serve user l in slot t, where user l is chosen by policy π_2 in slot $t-1$; (iii) use the optimal actions after slot t.

We compare the two policies π_1 and π_3 starting at slot $t-1$. They are almost the same except that policy π_3 may serve a suboptimal user in slot t. From the Principle of Optimality [17] we have

$$J^{(\pi_1)}(s_{t-1}, t-1) \geq J^{(\pi_3)}(s_{t-1}, t-1). \tag{11}$$

Next, we show that

$$J^{(\pi_3)}(s_{t-1}, t-1) = J^{(\pi_2)}(s_{t-1}, t-1). \tag{12}$$

By the definition of policy π_3, we have

$$J^{(\pi_3)}(s_{t-1}, t-1) = \alpha_{k_1} p_{k_1} + p_{k_1}\, J^{(\pi_3)}(s_{t-1}\setminus\{k_1\}, t) + (1-p_{k_1})\, J^{(\pi_3)}(s_{t-1}, t). \tag{13}$$

In the second term on the right side of (13), we have

$$J^{(\pi_3)}(s_{t-1}\setminus\{k_1\}, t) = \alpha_l p_l + p_l\, J_{t+1}(s_{t-1}\setminus\{k_1, l\}, t+1) + (1-p_l)\, J_{t+1}(s_{t-1}\setminus\{k_1\}, t+1), \tag{14}$$

where J_{t+1} is the optimal cost-to-go function defined in (5). In the third term on the right side of (13), we have

$$J^{(\pi_3)}(s_{t-1}, t) = \alpha_l p_l + p_l\, J_{t+1}(s_{t-1}\setminus\{l\}, t+1) + (1-p_l)\, J_{t+1}(s_{t-1}, t+1). \tag{15}$$

Similarly, under policy π_2 we have

$$J^{(\pi_2)}(s_{t-1}, t-1) = \alpha_l\, p_l + p_l\, J_t(s_{t-1}\setminus\{l\}, t) + (1-p_l)\, J_t(s_{t-1}, t), \tag{16}$$

where

$$J_t(s_{t-1}\setminus\{l\}, t) = \alpha_{k_1}\, p_{k_1} + p_{k_1} J_{t+1}(s_{t-1}\setminus\{k_1, l\}, t+1) + (1-p_{k_1})\, J_{t+1}(s_{t-1}\setminus\{l\}, t+1), \tag{17}$$

$$J_t(s_{t-1}, t) = \alpha_{k_1}\, p_{k_1} + p_{k_1}\, J_{t+1}(s_{t-1}\setminus\{k_1\}, t+1) + (1-p_{k_1})\, J_{t+1}(s_{t-1}, t+1). \tag{18}$$

By plugging (14)-(15) into (13), and (17)-(18) into (16), we obtain the equality (12). Comparing (11) and (12) shows that

$$J^{(\pi_1)}(s_{t-1}, t-1) \geq J^{(\pi_2)}(s_{t-1}, t-1),$$

which holds for all users $l \neq k_1 = \text{argmax}_{j\in s_{t-1}}\,\alpha_j\, p_j$ served in slot $t-1$ by policy π_2. As a result, we have $u_{t-1}^*(s_{t-1}) = \text{argmax}_{j\in s_{t-1}}\,\alpha_j\, p_j$. This completes the proof.

APPENDIX B
PROOF OF THEOREM 2

The proof is similar to that of Theorem 1, and we provide a sketch of the proof here. Let s_t be the subset of users having not received the packets from all multicast flows at the beginning of slot t. A policy π in the first frame is $\pi = (u_0, \ldots, u_{T-1})$, where u_t maps the state s_t to a multicast flow f to transmit in slot t. For example, the greedy policy consists of these decisions

$$u_t^{(\text{G})}(s_t) \in \text{argmax}_{f\in F}\, r_f(t),\ t = 0, \ldots, T-1,$$

where $r_f(t)$ is defined in (6).

In the last slot $(T-1)$ of the first frame, the base station transmits the flow

$$u_{T-1}^{(G)}(s_{T-1}) = \text{argmax}_{f\in F}\, r_f(T-1)$$

to maximize the the weighted sum throughput in this slot.

Denote by $J^{(\pi)}(s_t, t)$ the cost-to-go function of policy π from slot t with the "initial" system state s_t at time t. Given a slot $t \leq T-1$, assume that applying the greedy decisions from slot t and onwards is optimal. That is, for any policy $\pi_1 = (u_0, \ldots, u_{T-1})$ we define a new policy $\pi_2 = (u_0, \ldots, u_{t-1}, u_t^{(G)}, \ldots, u_{T-1}^{(G)})$ and assume that

$$J^{(\pi_2)}(s_t, t) \geq J^{(\pi_1)}(s_t, t), \text{ for all } s_t.$$

Consider the policy $\pi_3 = (u_0, \ldots, u_{t-2}, u_{t-1}^{(G)}, \ldots, u_{T-1}^{(G)})$, and we seek to show that

$$J^{(\pi_3)}(s_{t-1}, t-1) \geq J^{(\pi_2)}(s_{t-1}, t-1) \tag{19}$$

for each state s_{t-1}. Fix a state s_{t-1} in slot $t-1$. We assume that $u_{t-1}(s_{t-1}) \neq u_{t-1}^{(G)}(s_{t-1})$; otherwise, equation (19) holds with equality.

Given policies π_2 and π_3, we define a policy π_2' as follows:

1) Acting the same as π_2 from slot 0 to $t-2$.
2) Choosing the greedy decision $u_{t-1}^{(G)}$ in slot $t-1$.
3) Let f be the multicast flow that policy π_2 would transmit in slot $t-1$, according to the fixed "initial state" s_{t-1}. Then transmit the flow f in slot t regardless of the current state s_t.

The only different between π_3 and π_2' is the decision in slot t. By the induction assumption and the Principle of Optimality, policy π_3 outperforms π_2' after slot $t-1$, that is,

$$J^{(\pi_3)}(s_{t-1}, t-1) \geq J^{(\pi_2')}(s_{t-1}, t-1). \tag{20}$$

Next we argue that

$$J^{(\pi_2')}(s_{t-1}, t-1) = J^{(\pi_2)}(s_{t-1}, t-1). \tag{21}$$

In slot $t-1$, let $f^* = u_{t-1}^{(G)}(s_{t-1})$ be the greedy decision and $f = u_{t-1}(s_{t-1})$ is the actual flow served by π_2. We have assumed $f^* \neq f$. Because f^* is the greedy decision that is not taken by π_2 in slot $t-1$, it remains to be the greedy decision in slot t for π_2. As a result, policy π_2 serves flow f in slot $t-1$ and f^* in slot t. On the other hand, policy π_2' serves flow f^* in slot $t-1$ and f in slot t. In other words, policies π_2 and π_2' behave the same over the two slots $t-1$ and t. Because channels are i.i.d. over slots, equation (21) holds. Combining (20) and (21) shows that greedy policy is optimal in slot $t-1$. This completes the proof.

APPENDIX C
PROOF OF THEOREM 3

Define the Lyapunov function in the kth frame:

$$L(k) = \frac{1}{2}\sum_{n=1}^{N}(D_n^+(k))^2. \tag{22}$$

From (7), we have

$$\begin{aligned} D_n^+(k+1) &= \max\{D_n(k) + q_n - d_n(k), 0\} \\ &\leq \max\{D_n^+(k) + q_n - d_n(k), 0\} \\ &= \max\{D_n^+(k) - d_n(k), -q_n\} + q_n \\ &\leq \max\{D_n^+(k) - d_n(k), 0\} + q_n. \end{aligned}$$

The first inequality follows $D_n(k) \leq D_n^+(k)$ and the second inequality uses $q_n \geq 0$. Squaring the above and using simple arithmetics, we obtain

$$L(k+1) - L(k) \leq N + \sum_{n=1}^{N} D_n^+(k)(q_n - d_n(k)). \tag{23}$$

Define the Lyapunov drift over the kth frame:

$$\Delta(k) = \mathbb{E}[L(k+1) - L(k) \mid H(k)],$$

where $H(k)$ is the system history prior to the kth frame, including the information of $D_n^+(k)$. From (23), we have

$$\Delta(k) \leq N + \sum_{n=1}^{N} D_n^+(k)q_n - \sum_{n=1}^{N} D_n^+(k)\mathbb{E}[d_n(k) \mid H(k)]. \tag{24}$$

The MW policy is designed to minimize the right-hand side of (24).

Let $(q_1, \ldots, q_N)$ be a feasible throughput vector within the multicast throughput region Λ. It follows that there exists a boundary point $\boldsymbol{d}^* = (d_1^*, \ldots, d_N^*)$ of Λ that dominates $(q_1, \ldots, q_N)$ entrywise, i.e., $d_n^* \geq q_n$ for all n. Let π_0 be an admissible policy that achieves $\boldsymbol{d}^*$ in every frame. One choice of the policy π_0 is to repeat the GreedyM policy that achieves the boundary point $\boldsymbol{d}^*$ under a suitable weight vector $\boldsymbol{\alpha} \geq 0$ in every frame. Under policy π_0 we have

$$\mathbb{E}[d_n(k) \mid H(k)] = d_n^* \geq q_n.$$

Since the max-weight policy minimizes the right-hand side of (24), comparing it with policy π_0 yields

$$\sum_{n=1}^{N} D_n^+(k)\mathbb{E}[d_n^{\mathsf{MW}}(k) \mid H(k)] \geq \sum_{n=1}^{N} D_n^+(k)d_n^*.$$

As a result, the inequality (24) evaluated under the max-weight policy satisfies

$$\begin{aligned} \Delta(k) &\leq N + \sum_{n=1}^{N} D_n^+(k)q_n - \sum_{n=1}^{N} D_n^+(k)\mathbb{E}[d_n^{\mathsf{MW}}(k) \mid H(k)] \\ &\leq N + \sum_{n=1}^{N} D_n^+(k)q_n - \sum_{n=1}^{N} D_n^+(k)d_n^* \leq N. \end{aligned}$$

Taking expectation, summing over $k = 0, \ldots, K-1$, and using $D_n^+(0) = 0$, we have

$$\mathbb{E}[L(K)] = \frac{1}{2}\sum_{n=1}^{N}\mathbb{E}[\big(D_n^+(K)\big)^2] \leq KN.$$

It shows that

$$\mathbb{E}[\big(D_n^+(K)\big)^2] \leq 2NK, \quad n = 1, \ldots, N.$$

It follows that, as $K \to \infty$,

$$0 \leq \frac{\mathbb{E}[D_n^+(K)]}{K} \leq \sqrt{\frac{\mathbb{E}[(D_n^+(K))^2]}{K^2}} \leq \sqrt{\frac{2N}{K}} \to 0. \tag{25}$$

Using $D_n(K) \leq D_n^+(K)$ and (25), we have

$$\limsup_{K\to\infty} \frac{\mathbb{E}[D_n(K)]}{K} \leq \lim_{K\to\infty} \frac{\mathbb{E}[D_n^+(K)]}{K} = 0.$$

Equivalently, we have

$$\liminf_{K\to\infty} \frac{\mathbb{E}[-D_n(K)]}{K} \geq 0. \tag{26}$$

Plugging $D_n(K) = \sum_{k=0}^{K-1}(q_n - d_n(k))$ into (26), we conclude that the user n throughput d_n satisfies

$$d_n = \liminf_{K\to\infty} \frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}[d_n(k)] \geq q_n,$$

and the max-weight policy achieves the throughput requirement vector $(q_1, \ldots, q_N)$. The proof is complete.