Scheduling multicast traffic with deadlines in wireless networks

The MIT Faculty has made this article openly available. Please share how this access benefits you. Your story matters.
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 perceived by the users.

I. INTRODUCTION

The problem of scheduling real-time transmissions in a wireless network has relevant 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 shall 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 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 it subscribes. The goal of this paper is to study the set of achievable throughput vectors in this multicast scheduling problem, 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, instant 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. As a result, there seems to be no closed-form expression for the throughput region, and it is unclear whether a simple scheduling policy exists to satisfy feasible throughput requirements of the users. Our contributions in this paper are as follows.

- 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 inuction and interchange arguments, we show that the weighted sum throughput maximization problem is solved by greedy policies that transmits the multicast flow maximizing the immediate weighted sum reward in every slot. For the sake of exposition, we first analyze the special case of transmitting unicast flows, and generalize it to the multicast case.
- We develop adaptive throughput-optimal control policies

This work is supported by NSF grant CNS-0915988, ONR grant N00014-12-1-0064, ARO MURI grant W911NF-08-1-0238, and DTRA grants HDTRA1-07-1-0004 and HDTRA-09-1-005.
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 owed by the base station to satisfy its throughput requirement, and transmitting a multicast flow that yields large throughput gain from 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 heterogeneous 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 has been investigated with different modeling assumptions [5]–[11]. Scheduling packet transmissions with deadlines over multi-hop networks are considered in [12]–[14].

An 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 \( k \)th 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.

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 \( k \)th 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 \( \Lambda \) as the set of achievable throughput vectors \((d_1, \ldots, d_N)\) under the class II of admissible policies.

To characterize the throughput region \( \Lambda \), it suffices to focus on the set \( \Lambda_0 \) 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 \( k \)th slot, \( k \leq T - 1 \), 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 transmission. Therefore, it is difficult to obtain the multicast

\(^1\)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.
throughput region $\Lambda_0$ in closed form. We take an alternative approach that studies the throughput region $\Lambda_0$ 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 $\alpha = (\alpha_1, \ldots, \alpha_N)$ be a nonnegative weight vector, and consider the optimization problem:

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

(1)

where $\mathbb{E}[d_n^*(0)]$ denotes the expected user $n$ throughput in the first frame (i.e., frame zero) under policy $\pi$. The solution to the problem (1) specifies a boundary point at the intersection of the throughput region $\Lambda_0$ and the supporting hyperplane that has the normal vector $\alpha$ and is tangent to $\Lambda_0$. See Fig. 2 for an example.

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 every 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

$$\maximize \sum_{n=1}^{N} \alpha_n \mathbb{E}[d_n^*(0)]$$

subject to $$(\mathbb{E}[d_n^*(0)])_{n=1}^{N} \in \Lambda_0^{\text{unicast}},$$

and is solved by the following policy.

The Greedy Policy for a Unicast System (GreedyU)

1) In slot $k \in \{0, 1, \ldots, T - 1\}$, transmit the unicast flow $j_k^* \in \arg\max_{j \in s_k} \alpha_j p_j$, 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 the context of polymatroid optimization, and its optimality can be proved by using the complimentary slackness condition in 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 a new 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 $k$th 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 rewritten 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 $k$th 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:

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

(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.

According to [17, Prop. 1.3.1], the problem (3) is solved by a dynamic programming algorithm as follows. Define the functions

$$J_T(s_T) = 0, \quad s_T \subseteq S,$$

(4)

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 the context of polymatroid optimization, and its optimality can be proved by using the complimentary slackness condition in 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 a new 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 $k$th 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 rewritten 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 $k$th 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:

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

(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.

According to [17, Prop. 1.3.1], the problem (3) is solved by a dynamic programming algorithm as follows. Define the functions

$$J_T(s_T) = 0, \quad s_T \subseteq S,$$
\[ J_k(s_k) = \max_{u_k \in U_k(s_k)} \mathbb{E} \{ g_k(s_k, u_k) + J_{k+1}(s_{k+1}) \}, \quad (5) \]

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 \( \Lambda \) at a maximum weighted sum throughput vector computed by the GreedyU policy.

![Fig. 3. The unicast throughput region in a two-user wireless network.](image)

**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 \( \alpha \geq 0 \).

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 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 \( \alpha = (\alpha_1, \ldots, \alpha_N) \).
2) In slot \( t \in \{0, \ldots, T - 1\} \), compute
   \[ r_f(t) = \sum_{i \in U_f(t)} \alpha_i p_i \]
   for each flow \( f \in F \); let \( r_f(t) = 0 \) if \( U_f(t) = \emptyset \).
3) In slot \( t \), pick the multicast flow
   \[ f^* = \arg\max_{f \in F} r_f(t). \]
   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 \( q_n \) 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 \( \Lambda \).

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 \( k \)th frame, or 0 otherwise. At the beginning of the \( k \)th frame, define

\[ D_n(k) = \sum_{j=0}^{k-1} (q_n - d_n(j)), \quad (6) \]

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. 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 $k$th frame, apply the admissible policy $π ∈ Π$ that maximizes the weighted sum throughput $\sum_{n=1}^{N} D_n^+(k) \cdot Ε[d_n(k)]$ in the frame.

Note that the expectation $Ε[d_n(k)]$ is with respect to both unreliable channels and control decisions. We observe from the term $\sum_{n=1}^{N} D_n^+(k) \cdot Ε[d_n(k)]$ 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 $α_n = D_n^+(k)$ 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 $k$th frame, observe the subsets $\{U_f(t)\}_{f ∈ F}$ of unserved users and compute

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

for each multicast flow $f ∈ F$.

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

The sum $\sum_{n ∈ U_f(t)} p_n$ 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 $ΕWST_{fb}(α)$ be the maximum expected weighted sum throughput of multicast transmissions without the user feedback, where $α ≥ 0$ 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 $(1 - (1 - p_n)^{τ_f})$. As a result, we have

$$ΕWST_{fb}(α) = \max_{\sum_{f ∈ F} τ_f = T} \sum_{f ∈ F} \sum_{n ∈ N_f} α_n (1 - (1 - p_n)^{τ_f}).$$

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

**A. The popularity of a multicast flow**

We examine how the number of subscribers in 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:

$$ΕWST_{fb}(1) - ΕWST_{nofb}(1) \times \frac{100}{ΕWST_{fb}(1)},$$

which is the normalized throughput gain from the feedback information. Fig. 5 shows that the metric (7) 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 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 $p^{τ_f}$. 

![Figure 4](image_url)
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.

\section*{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 outperforms those without feedback as expected.

\section*{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.

\section*{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 surprisingly achieved by greedy policies that seek to maximize immediate weighted sum throughput in every slot. We utilize this greedy policy to design adaptive throughput optimal policies over the multicast throughput region. This policy optimizes the tradeoff between serving a user not receiving enough packets to satisfy its QoS constraint, and transmits a multicast flow that yields large
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 the user feedback.

REFERENCES


APPENDIX A

PROOF OF THEOREM 1

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

\[
J_k(s_k) = \max_{j \in S_k} \left\{ \alpha_j p_j + p_j J_{k+1}(s_k \setminus \{j\}) \right\} + (1 - p_j)J_{k+1}(s_k),
\]

(8)

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

\[
u_k^*(s_k) = \arg\max_{j \in S_k} \left\{ \alpha_j p_j + p_j J_{k+1}(s_k \setminus \{j\}) \right\}
\]

(9)

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 \( \pi = (u_0, u_1, \ldots, u_{T-1}) \):

\[
J^\pi(s, k) = E \left\{ \sum_{t=k}^{T-1} g_t(s_t, u_t) \mid s_k = s, s \in S. \right\}
\]

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 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 (8) 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: \( \pi_1 = (\ldots, u_{T-2}^*, u_{T-1}^*) \) and \( \pi_2 = (\ldots, u_{T-2}^*(2), u_{T-1}^*) \), where

\[
J_{T-2}^\pi(s_{T-2}) = k_1 \triangleq \arg\max_{j \in S_{T-2}} \alpha_j p_j,
\]

\[
u_{T-2}^\pi(s_{T-2}) = l, \ l \neq k_1.
\]

In other words, both policies adopt the optimal action in slot \( T-1 \). But in slot \( T-2 \), policy \( \pi_1 \) uses the greedy decision while policy \( \pi_2 \) uses any other decision in slot \( T-2 \).

Define \( k_2 = \arg\max_{j \in S_{T-2}} \{j^* \} \alpha_j p_j \). It follows that \( J^\pi(s_{T-2}, T-2) = \alpha_{k_1} p_{k_1} + \alpha_{k_2} p_{k_2} + (1 - p_{k_1}) \alpha_{k_1} p_{k_1} \),

\[
J^\pi_{T-2}(s_{T-2}) = \alpha_{k_1} p_{k_1} + \alpha_{k_2} p_{k_2} + (1 - p_{k_1}) \alpha_{k_1} p_{k_1} + (1 - p_{k_2}) \alpha_{k_2} p_{k_2} = \alpha_{k_1} + \alpha_{k_2} p_{k_2}.
\]

By definition, we have \( \alpha_{k_1} p_{k_1} \geq \alpha_{k_2} p_{k_2} \geq \alpha_{k_1} p_{k_1} \). It follows that \( J^\pi(s_{T-2}, T-2) \geq J^\pi_{T-2}(s_{T-2}, T-2) \) for any \( l \neq k_1 \). Therefore, \( u_{T-2}^*(s_{T-2}) = \arg\max_{j \in S_{T-2}} \alpha_j p_j \).

(iv) To prove by backward induction, given \( 0 < t \leq T-1 \), assume that the optimal policy at state \( s_k \) is \( u_k^*(s_k) = \arg\max_{j \in S_k} \alpha_j p_j \) 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 function \( J^\pi(s_{t-1}, t-1) \) during slots in \([t-1, T])\). Consider the two policies: \( \pi_1 = (\ldots, u_{t-1}^*(1), u_t^*, \ldots, u_{T-1}^*) \) and \( \pi_2 = (\ldots, u_{t-2}^*(2), u_{t-1}^*, \ldots, u_{T-1}^*) \), where

\[
u_{t-1}^\pi(s_{t-1}) = k_1 \triangleq \arg\max_{j \in S_{t-1}} \alpha_j p_j,
\]

\[
u_{t-1}^\pi(s_{t-1}) = l, \ l \neq k_1.
\]

In other words, policy \( \pi_1 \) uses the greedy decision in slot \( t-1 \) and optimal decisions afterwards. Policy \( \pi_2 \) serves another
user $l \neq k_1$ in slot $t - 1$ and uses optimal decisions afterwards. Now, given the policies $\pi_1$ and $\pi_2$, we define a third policy $\pi_3 = (\ldots, u_{t-1}^{(3)}, u_t^{(3)}, u_{t+1}^{(3)}, \ldots, u_{T-1}^{(3)})$ working as follows: (i) greedily serve user $k_1$ in slot $t - 1$ according to $\pi_1$; (ii) always serve user $l$ in slot $t$, where user $l$ is chosen by policy $\pi_2$ in slot $t - 1$. Policy $\pi_3$ uses the optimal actions after slot $t$.

We compare the two policies $\pi_1$ and $\pi_3$ starting at slot $t - 1$. They are almost the same except that policy $\pi_3$ may serve a suboptimal user in slot $t$. From the Principle of Optimality [17] we have

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

(10)

Next, we have

$$J(\pi_3)(s_{t-1}, t - 1) = \alpha_k p_k + p_k J(\pi_3)(s_{t-1} \setminus \{k_1\}, t) + (1 - p_k) J(\pi_3)(s_{t-1}, t),$$

$$J(\pi_3)(s_{t-1} \setminus \{k_1\}, t) = \alpha_k p_1 + p_1 J(\pi_3)(s_{t-1} \setminus \{k_1, l\}, t + 1) + (1 - p_1) J(t + 1)(s_{t-1} \setminus \{k_1\}, t + 1),$$

$$J(\pi_3)(s_{t-1}, t) = \alpha_l p_1 + p_1 J(t + 1)(s_{t-1} \setminus \{l\}, t + 1) + (1 - p_1) J(t + 1)(s_{t-1}, t + 1).$$

Similarly, we have

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

$$J(t(s_{t-1} \setminus \{l\}, t) = \alpha_k p_k + p_k J(t+1)(s_{t-1} \setminus \{k, l\}, t + 1) + (1 - p_k) J(t+1)(s_{t-1} \setminus \{l\}, t + 1),$$

$$J(t(s_{t-1}, t) = \alpha_l p_k + p_k J(t+1)(s_{t-1} \setminus \{k, l\}, t + 1) + (1 - p_k) J(t+1)(s_{t-1}, t + 1).$$

From the above, we can observe that

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

(11)

Comparing (10) and (11) shows that

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

which holds for all users $l \neq k_1 = \arg\max_{j \in s_{t-1}} \alpha_j p_j$ served in slot $t - 1$ by policy $\pi_2$. As a result, we have $u_{t-1}^*(s_{t-1}) = \arg\max_{j \in s_{t-1}} \alpha_j p_j$. 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. Let $s_k$ be the subset of users having not received the packets at the beginning of slot $k$. A policy $\pi$ in the first frame is $\pi = (u_0, \ldots, u_{T-1})$, where $u_k$ maps the state $s_k$ to a multicast flow $f$ to transmit in slot $k$. For example, the greedy policy consists of these decisions

$$u_k^{(G)}(s_k) = \arg\max_{f \in F} r_f(t), \quad k = 0, \ldots, T - 1.$$

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

$$u_{T-1}^{(G)}(s_{T-1}) = \arg\max_{f \in F} r_f(T - 1)$$

to maximize the weighted throughput in this slot.

Denote by $J(\pi)(s_t, t)$ the cost-to-go function of policy $\pi$ from slot $t$ with the “initial” system state $s_t$ at time $t$. Given a slot $t \leq T - 1$, we 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

$$J(\pi_2)(s_t, t) \geq J(\pi_1)(s_t, t),$$

for each $s_t$.

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

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

for each $s_{t-1}$. (12)

Fix a given state $s_{t-1}$ in slot $t - 1$. We assume that $u_{t-1}(s_{t-1}) \neq u_t^{(G)}(s_{t-1})$; otherwise, equation (12) holds with equality.

Given policies $\pi_2$ and $\pi_3$, we define a policy $\pi_1^*$ as follows:

1. Acting the same as $\pi_2$ from slot 0 to $t - 2$.
2. Choosing the greedy decision $u_t^{(G)}$ in slot $t - 1$.
3. Let $f$ be the multicast flow that policy $\pi_2$ 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 difference between $\pi_3$ and $\pi_1^*$ is the decision in slot $t$. By the induction assumption and the Principle of Optimality, policy $\pi_3$ outperforms $\pi_1^*$ after slot $t - 1$, that is,

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

(13)

Next we argue that

$$J(\pi_1^*)(s_{t-1}, t - 1) = J(\pi_2)(s_{t-1}, t - 1).$$

(14)

In slot $t - 1$, let $f^*$ be the greedy decision and $f$ is the actual flow served by $\pi_2$. We have assumed $f^* \neq f$. Because $f^*$ is the greedy decision that is not taken by $\pi_2$ in slot $t - 1$, it remains to be the greedy decision in slot $t$ for $\pi_2$. As a result, policy $\pi_2$ serves flow $f$ in slot $t - 1$ and $f^*$ in slot $t$. On the other hand, policy $\pi_1^*$ serves flow $f^*$ in slot $t - 1$ and $f$ in slot $t$. Because channels are i.i.d. over slots, the statistical behavior of policies $\pi_2$ and $\pi_1^*$ are the same over the two slots $t - 1$ and $t$. Therefore, equation (14) holds. Combining (13) and (14) 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 $k$th frame:

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

(15)

From (6), we have

$$D_n^+(k + 1) = \max\{D_n(k) + q_n - d_n(k), 0\} \leq \max\{D_n^+(k) + q_n - d_n(k), 0\} \leq \max\{D_n^+(k) - d_n(k), 0\} + q_n.$$
Taking expectation and summing over $k$ have

$$L(k + 1) - L(k) \leq N + \sum_{n=1}^{N} D_n^+(k)(q_n - d_n(k)).$$  \hspace{1cm} (16)

Define the Lyapunov drift over the $k$th frame:

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

where $H(k)$ is the system history prior to the $k$th frame. From (16), we have

$$\Delta(k) \leq N + \sum_{n=1}^{N} D_n^+(k)q_n - \sum_{n=1}^{N} D_n^+(k)\mathbb{E}[d_n(k) \mid H(t)].$$ \hspace{1cm} (17)

The MW policy is designed to minimize the right-hand side of (17).

Let $(q_1, \ldots, q_N)$ be a feasible throughput vector within the multicast throughput region $\Lambda$. It follows that there exists a boundary point $d^* = (d^*_1, \ldots, d^*_N)$ of $\Lambda$ that dominates $(q_1, \ldots, q_N)$ entrywise, i.e., $d^*_n \geq q_n$ for all $n$. Let $\pi_0$ be the stationary policy that achieves $d^*$, and under this policy we have

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

Since the max-weight policy minimizes the right-hand side of (17), comparing it with policy $\pi_0$ yields

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

As a result, the inequality (17) evaluated under the max-weight policy satisfies

$$\Delta(k) \leq N + \sum_{n=1}^{N} D_n^+(k)q_n - \sum_{n=1}^{N} D_n^+(k)\mathbb{E}[d^*_n(k) \mid H(t)]$$

$$\leq N + \sum_{n=1}^{N} D_n^+(k)q_n - \sum_{n=1}^{N} D_n^+(k)d^*_n \leq N.$$

Taking expectation and summing over $k = 0, \ldots, K - 1$, we have

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

Since $D_n^+(K) \geq 0$, we have

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

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

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

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

$$\limsup_{K \to \infty} \frac{\mathbb{E}[D_n(K)]}{K} \leq 0 \Rightarrow \liminf_{K \to \infty} \frac{\mathbb{E}[\text{max}(D_n(K))]}{K} \geq 0.$$