Fifty-third Annual Allerton Conference
Allerton House, UIUC, Illinois, USA
September 29 - October 2, 2015

Optimal Scheduling of Real-Time Traffic in Wireless Networks with Delayed Feedback

Kyu Seob Kim, Chih-Ping Li, Igor Kadota and Eytan Modiano
Laboratory for Information and Decision Systems
Massachusetts Institute of Technology

Abstract— **In this paper we consider a wireless network composed of a base station and a number of clients, with the goal of scheduling real-time traffic. Even though this problem has been extensively studied in the literature, the impact of delayed acknowledgment has not been assessed. Delayed feedback is of increasing importance in systems where the round trip delay is much greater than the packet transmission time, and it has a significant effect on the scheduling decisions and network performance.**

Previous work considered the problem of scheduling real-time traffic with instantaneous feedback and without feedback. In this work, we address the general case of delayed feedback and use Dynamic Programming to characterize the optimal scheduling policy. An optimal algorithm that fulfills any feasible minimum delivery ratio requirements is proposed. Moreover, we develop a low-complexity suboptimal heuristic algorithm which is suitable for platforms with low computational power. Both algorithms are evaluated through simulations.

I. INTRODUCTION

Wireless networks provide a flexible platform to support a variety of applications such as voice, multimedia, e-mail and messaging. The increasing number of clients using this technology leads to a growing demand for various Quality of Service requirements.

Two requirements associated with real-time traffic are the maximum time to deliver a packet and the minimum delivery ratio. The first imposes a deadline by which the packets must be delivered, while the second limits the fraction of packets that miss the deadline. Meeting those requirements over an unreliable wireless channel is challenging due to transmission errors that result in packet losses.

Feedback is often used to improve reliability. After receiving a packet, the receiver sends an acknowledgment to the sender if the data was received correctly. The feedback mechanism can be implemented in a number of different ways, depending on the network characteristics. Latency, the ratio of the round trip delay time to the data transmission time, plays an important role in the design and performance of systems employing feedback [1].

From the sender standpoint, the feedback associated with each transmission arrives after a round trip delay. In low latency systems, such as wireless LANs with low data-rate and short round trip delay, waiting for the acknowledgment before transmitting the next packet has minimal impact on performance. On the other hand, when the round trip delay takes much longer then the data transmission, it is necessary to allow multiple data packets transmissions in a row prior to receiving the feedback. However, to the best of our knowledge, the influence of the round trip delay has not been analyzed in the context of scheduling real-time traffic.

The problem of optimizing scheduling decisions in networks that support traffic with deadlines has been addressed in the literature in a variety of contexts and under diverse system conditions. Wired multi-hop networks with unicast traffic have been studied in [2], [3], [4]. In [2], a tree network with a single destination was considered. It was shown that forwarding the packet with shortest time to extinction (STE policy) maximizes the delivery ratio. In [3], a more general network with multiple source destination pairs was addressed. An online algorithm for joint admission and scheduling was developed using time slot reservations. A similar network setting, using a different analytical model, was considered in [4], where a tractable approach for scheduling in multi-hop networks based on adapting the service discipline to meet delivery ratio constraints was designed.

In other related works, researchers have studied wireless single-hop networks [5], [6], [7], [8], [9]. The authors in [5] proposed an analytical framework to model a network supporting unicast traffic with deadlines and instant feedback. Based on this framework, two debt-based policies that meet any feasible constraints were presented. Articles [6], [7], [8], [9] extend the model in [5] to a variety of scenarios: [6] considers variable-bit-rate traffic; [7] generalizes for heterogeneous deadlines and time-varying channel; [8] models multicast traffic with instant feedback; and [9] models broadcast traffic with no feedback.

Prior literature addressed the specific cases of instant feedback and no feedback. In the case of instant feedback, the optimal scheduling policy was found to be a Greedy policy [5], [8], where clients are prioritized before every transmission opportunity and the undelivered packet with highest priority is forwarded. In the case of no feedback, as the scheduler does not know whether a packet was delivered or not, slots are allocated to each client a priory according to the algorithm found in [9].

The goal of this paper is to address the general case, in which the feedback delay is a parameter of the network. Our main contributions include:

- extend the network model of [5] to account for arbitrary

This work was supported by NSF Grant CNS-1217048 and ONR Grant N00014-12-1-0064, and by a fellowship from CAPES/Brazil.

feedback delays;
- solve for the optimal scheduling policy and characterize the feasible throughput region of the network;
- develop an optimal dynamic algorithm that satisfies any feasible delivery ratio requirements;
- develop a low-complexity heuristic algorithm that is shown through simulation to achieve good performance.

The remainder of this paper is outlined as follows. In the next section the system model is presented. In Sec. III, the problem is formulated and solved using Dynamic Programing. In Sec. IV, we describe the Feasible Region and the optimal algorithm. In Sec. V, we propose a heuristic algorithm and present simulation results. We conclude the paper and comment on further work in Sec. VI.

II. SYSTEM MODEL

Consider a wireless system with a base station (BS) sending unicast data packets with deadlines to N clients. Time is divided into slots, and T successive slots form a frame. The BS generates one packet per client at the beginning of every frame and drops those packets that were not delivered by the end of the frame. Notice that the maximum time to deliver a packet is the frame length (T). The kth frame comprises slots $t \in \{kT, \cdots, (k+1)T-1\}$, where $t,k \in \mathbb{Z}^+$ are the slot index and the frame index, respectively.

In a slot, the BS transmits the packet of a selected client $i \in \{1,\cdots,N\}$ over the wireless channel. The packet is successfully delivered to that client with probability p_i and a transmission error occurs with probability $1-p_i$. In either case, the client sends a feedback signal through a delayed control channel. The ACK/NACK reaches the BS, without errors, d slots after the transmission. Therefore, when a packet is transmitted in slot t, the BS receives the feedback at the end of slot $t+d$, implying that the feedback information is only available for the scheduling decision of slot $t+d+1$ onwards. Fig. 1 illustrates this events.

This network model allows us to consider the problem over one frame only, as in [8], due to the periodicity of the system. Packets arrive at the beginning of each frame and undelivered packets are dropped at the end of the same frame. The system is renewed at the start of each frame, thus, henceforward, we regard the system over the first frame, i.e. $k=0$.

The scheduling policies which are considered in this paper are *work conserving* and *non-anticipatory*, i.e. policies which never idle when there are undelivered packets in the system, and do not use future information in making decisions. This class of policies is denoted by Π and the policies by $\eta \in \Pi$.

Recalling that packets are dropped from the system when they miss the deadline, a performance metric of interest is the long-term delivery ratio of each client, referred to as its throughput. Let $D_i^\eta(k)$ be an indicator random variable that is equal to 1 if, by following policy η, the packet is delivered to client i during the kth frame, and zero otherwise. Then,

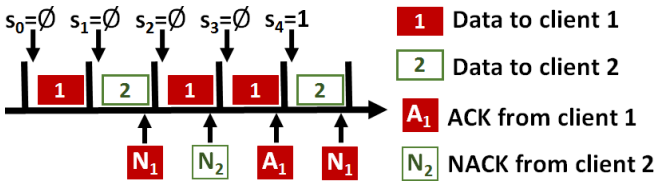

Fig. 1. Illustration of a frame in time for $N=2$, $T=5$ and $d=1$. The set s_t represents the clients that have received their packet and whose ACK signal have reached the BS by the beginning of the tth time slot.

the throughput of client i under policy η is given by

$$\hat{q}_i^\eta := \liminf_{K\to\infty} \frac{1}{K} \sum_{k=0}^{K-1} D_i^\eta(k). \quad (1)$$

Observe that if the same policy is used across frames, by the Law of Large Numbers $\hat{q}_i^\eta = E[D_i^\eta(0)]$, where the RHS is the expected throughput. We say that a given vector of required minimum delivery ratios, $(q_i)_{i=1}^N$, is fulfilled by policy η if and only if $\hat{q}_i^\eta \geq q_i$, for all i, with probability 1.

Another metric of interest is the expected weighted sum throughput (EWST). In particular, let $\vec{\alpha} = (\alpha_i)_{i=1}^N$ be a vector of client weights with $\alpha_i \geq 0$. Then, the EWST is expressed as $\sum_{i=1}^N \alpha_i E[D_i^\eta(0)]$. This metric has a central role in the dynamic program, for it characterizes a network-wide performance.

III. DYNAMIC PROGRAMMING FORMULATION

In this section, the discrete-time system is considered from the Dynamic Programming [10] standpoint, where we focus on the performance metric EWST. The three components of the cost-to-go function (state, transition and reward) are presented and the finite-horizon program is solved.

A. State Augmentation

Consider the set of clients $\mathcal{N} = \{1,...,N\}$ and the state space $S = 2^{\mathcal{N}}$, which is the collection of all subsets of clients. Let $s_t \in S$, $t \in \{0,\cdots,T-1\}$, represent the clients that have received their packet and whose ACK signal have reached the BS by the beginning of the tth time slot, as illustrated in Fig. 1. One implication of this definition is that $s_0 = \cdots = s_d = \emptyset$, because the first time a feedback is received in any frame is at the end of slot d.

From the point of view of the BS, clients can be divided into three groups. i) clients with confirmed delivered packets, ii) clients that have been served but the feedback signal has not arrived yet and iii) clients that were not served or have confirmed a transmission error (NACK). As each client belongs to one group, a complete representation of the system state can be achieved by characterizing two out of the three groups.

Let u_t be the scheduling decision (or control) in time slot t, i.e. the client selected by the BS for transmission in that slot. The set $(u_{t-1},\cdots,u_{t-d})$ depicts the transmissions which have not been acknowledged by the beginning of slot t. Hence, combining s_t with $(u_{t-1},\cdots,u_{t-d})$ gives the augmented state $\tilde{s}_t = (s_t; u_{t-1},\cdots,u_{t-d})$, which fully represents the system.

Recalling that the policies are work conserving, the set of allowed scheduling decisions can be defined as $U_t(s_t) = \mathcal{N} \backslash s_t$. Imposing $u_t \in U_t(s_t)$, $\forall t$, guarantees that the BS will only idle when all packets have been acknowledged, i.e. $U_t(s_t = \mathcal{N}) = \emptyset$.

B. State Transition

As mentioned, the first feedback arrives to the BS at the end of slot d. This means that the (augmented) state[1] for $t \in \{0, 1, \cdots, d-1\}$ changes according to the transition probability

$$P\{\tilde{s}_{t+1} = (s_{t+1} = \emptyset; u_t, u_{t-1}, \cdots, u_0) | \tilde{s}_t, u_t\} = 1. \quad (2)$$

On the other hand, when $t \in \{d, \cdots, T-1\}$, the transition depends on the feedback to be received at the end of slot t. There are three possible events: the feedback is a NACK, an ACK or it is associated to the transmission of a previously confirmed delivery. These events are illustrated in Fig. 1 at $t = 2$, 3 and 4, respectively. In the case of a NACK, the associated transition probability is

$$P\{\tilde{s}_{t+1} = (s_t; u_t, \cdots, u_{t-d+1}) | \tilde{s}_t, u_t\} = 1 - p_{u_{t-d}}, \quad (3)$$

and in the case of an ACK,

$$P\{\tilde{s}_{t+1} = (s_t \cup u_{t-d}; u_t, \cdots, u_{t-d+1}) | \tilde{s}_t, u_t\} = p_{u_{t-d}}. \quad (4)$$

Notice that these transitions can only take place at the tth slot if $u_{t-d} \in U_t(s_t)$. In contrast, $u_{t-d} \notin U_t(s_t)$ yields the deterministic transition

$$P\{\tilde{s}_{t+1} = (s_t; u_t, \cdots, u_{t-d+1}) | \tilde{s}_t, u_t\} = 1. \quad (5)$$

With all state transition probabilities defined, in the next section we analyze the reward function associated with each state transition.

C. Reward Function

The last concept to be introduced prior to the actual DP formulation is the reward function. The reward is directly related to the EWST. From the expression of this performance metric, a straightforward definition for the reward at time t is that the system gets a reward of $\alpha_{u_{t-d}}$ when an ACK is received from client u_{t-d} and zero otherwise.

For describing the reward function, three periods are distinguished. The first, $t \in \{0, \cdots, d-1\}$, in which there is no feedback:

$$g_t(\tilde{s}_{t+1}, \tilde{s}_t) = 0. \quad (6)$$

The second, $t \in \{d, \cdots, T-1\}$, with delayed feedback:

$$g_t(\tilde{s}_{t+1}, \tilde{s}_t) = \\ = \begin{cases} \alpha_{u_{t-d}} & \text{if } s_{t+1} = s_t \cup \{u_{t-d}\} \text{ and } u_{t-d} \in U_t(s_t), \\ 0 & \text{otherwise.} \end{cases} \quad (7)$$

[1]Henceforth we will use the terms augmented state and state interchangeably to denote $\tilde{s}_t$

The third, $t = T$, to account for the expected reward of packets that are successfully received by the end of the frame but whose ACK is received after the end of the frame:

$$g_T(\tilde{s}_T) = \sum_{i \in U_t(s_t)} \alpha_i (1 - (1 - p_i)^{n_i}), \quad (8)$$

where n_i is the cardinality of i in the set $\{u_{T-1}, \cdots, u_{T-d}\}$ and $U_t(s_t) = \mathcal{N} \backslash s_t$.

D. Dynamic Program

Within a frame, the system evolves in steps and yields a reward which is additive over time, making it adequate for a DP formulation. For a given $\vec{\alpha}$, the problem of optimizing the EWST is defined as

$$EWST(\vec{\alpha}) := \max_{\eta \in \Pi} \sum_{i=1}^{N} \alpha_i E\left[D_i^{\eta}(0)\right], \quad (9)$$

and solved by applying the cost-to-go function $J_t(\tilde{s}_t)$ recursively. Working backwards in time, we have

- for $t = T$:

$$J_T(\tilde{s}_T) = g_T(\tilde{s}_T = (s_T; u_{T-1}, \cdots, u_{T-d}));$$

- for $t \in \{d, \cdots, T-1\}$, the general equation:

$$J_t(\tilde{s}_t) = \max_{u_t \in U_t(s_t)} E_{\tilde{s}_{t+1}}[g_t(\tilde{s}_{t+1}, \tilde{s}_t) + J_{t+1}(\tilde{s}_{t+1})],$$

  which, in the case $u_{t-d} \notin U_t(s_t)$, takes the form:

$$J_t(\tilde{s}_t) = \max_{u_t \in U_t(s_t)} [J_{t+1}(s_t; u_t, \cdots, u_{t-d+1})],$$

  and with $u_{t-d} \in U_t(s_t)$, the form:

$$J_t(\tilde{s}_t) = \max_{u_t \in U_t(s_t)} [\alpha_{u_{t-d}} p_{u_{t-d}} + \\ + p_{u_{t-d}} J_{t+1}(s_t \cup u_{t-d}; u_t, \cdots, u_{t-d+1}) + \\ + (1 - p_{u_{t-d}}) J_{t+1}(s_t; u_t, \cdots, u_{t-d+1})];$$

- lastly, for $t \in \{0, \cdots, d-1\}$:

$$J_t(\tilde{s}_t) = \max_{u_t \in U_t(s_t)} [J_{t+1}(s_{t+1} = \emptyset; u_t, \cdots, u_0)].$$

At each step t and for every possible $\tilde{s}_t$, the value of $J_t(\tilde{s}_t)$ is attained by choosing the optimal u_t^*. By keeping track of those choices, the optimal policy η^* is obtained. The output of the recursion at $t = 0$ is the optimal performance $J_0(\emptyset) = EWST(\vec{\alpha})$ associated with η^*.

So far, the mechanism to obtain the optimal network performance for a given $\vec{\alpha}$ was described. Nonetheless, the actual constraint is not the weight vector, but a vector of minimum delivery ratios, $(q_i)_{i=1}^N$. In the next section, we propose an optimal algorithm which fulfills any feasible vector of q_i.

IV. FEASIBLE REGION AND OPTIMAL ALGORITHM

For defining the concept of feasibility, we first present the Feasible Throughput Region. Then, the optimal algorithm called Frame-based Max-weight Policy is described.

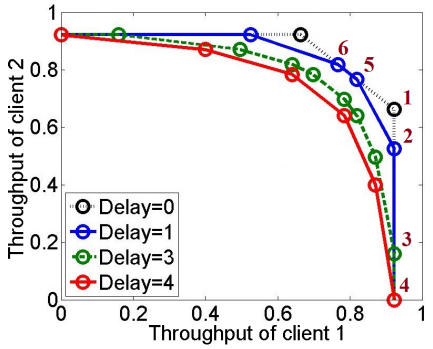

Fig. 2. Feasible Throughput Region of a two-user wireless network with $T = 5$, $p_1 = p_2 = 0.4$ and different values of delay. The circles represent the throughput vectors attained by the optimal policies.

A. Feasible Throughput Region

The Feasible Region is the set of throughput vectors $\left(E\left[D_i^{\eta}(0)\right]\right)_{i=1}^{N}$ that can be achieved by the policies in Π. To characterize this region, similarly to [8], we evaluate its boundaries. From the DP formulation, (9), it can be seen that, for a given weight vector, no policy attains a higher performance than η^*. Therefore, by sweeping $\vec{\alpha}$ and iteratively solving the Dynamic Program, the optimal throughput vectors, denoted by $(E[D_i^*(0)])_{i=1}^{N}$, are collected. It then follows that the Feasible Region is the convex hull of those vectors. The convex hull represents randomizations between different η^*.

In Fig. 2 the Feasible Region for a fixed (T,N,p_i) is illustrated. Delays range from instant feedback, $d = 0$, to no feedback, $d \geq T-1$. The convex hull is depicted by the lines and the optimal throughput vectors by the circles, some of which are numbered for future reference.

Any vector of minimum delivery ratios, $(q_i)_{i=1}^{N}$, that lies inside the convex hull is called feasible, for it can be fulfilled by a randomization over the optimal policies. Next, we propose a feasibility optimal algorithm, i.e. an algorithm which fulfills any feasible requirement.

B. Optimal Algorithm

The Optimal Algorithm is a dynamic schedule that drives the throughput vector of the network to a point above the feasible requirement. To achieve this goal, we use the delivery debt, defined as the difference between the number of required deliveries and the current number of deliveries, as the weight vector ($\vec{\alpha}$) in the Dynamic Program. The output is the optimal policy η_k^* to be applied in the current frame. This describes the outline of one iteration of the Optimal Algorithm. A precise description follows.

FRAME-BASED MAX-WEIGHT ALGORITHM:

(i) At the beginning of the frame k, calculate for each client:
- the total number of packets delivered $Q_i(k)$;
- the delivery debt $d_i(k) = kq_i - Q_i(k)$.

(ii) Solve the Dynamic Program with $\alpha_i = \max\{d_i(k),0\}$;

(iii) Employ η_k^* during frame k.

In each frame, the algorithm employs the optimal policy that gives higher weights to clients with higher debts. A proof similar to the one in [8], using Lyapunov Drift techniques [11], can be given to show that the Frame-based Max-weight Algorithm is feasibility optimal. Simulation results are provided in Sec. V.

A negative aspect of the Optimal Algorithm is that it can be computationally demanding, for it involves solving the Dynamic Program at every frame. One iteration of the DP entails maximizing over u_t, at each step t, for every possible $\tilde{s_t} = (s_t; u_{t-1}, \cdots, u_{t-d})$. Recalling that there are T slots in a frame, 2^N subsets s_t and N^d different sets $(u_{t-1}, \cdots, u_{t-d})$, this amounts to $\mathcal{O}(2^N N^d T)$ maximizations in a single iteration. A heuristic algorithm which is suited for platforms with low computational power and has a performance comparable to the optimal is presented next.

V. INSIGHT INTO THE POLICIES AND HEURISTIC ALGORITHM

By solving the DP, the Optimal Algorithm randomizes over all possible η^*, covering the entire Feasible Region. The Heuristic Algorithm uses a predefined subset of policies and a randomization based on vector projections to cover a considerable portion of the Feasible Region. The first part of this section discusses the choice of the subset of policies and the second part describes the Heuristic Algorithm.

A. Subset of Policies

A well-known scheduling strategy is the Greedy Policy. It prioritizes clients and, in each slot, schedules the client with highest priority. The Greedy Policy in the context of this paper is as follows: in slot t, the BS transmits $u_t = argmax_{u_t \in U_t(s_t)} \alpha_{u_t} p_{u_t}$. Notice that a given weight vector $\vec{\alpha}$ defines a priority ordering of clients in the Greedy Policy. It was shown in [8], that this is the optimal policy for instant feedback. The outcome of the DP extends this result, showing that in some ranges of $\vec{\alpha}$ the same Greedy Policy is also optimal for delayed feedback. In Fig. 2, points 1 through 4 and their symmetric along the diagonal correspond to Greedy Policies.

For a given network setting (T, N, p_i, d), consider the subset containing $N!$ Greedy Policies, one for each possible priority sequence. By randomizing over those policies, it is possible to obtain the convex hull of the corresponding limiting points. Fig. 3 shows the simulated throughput vectors of the Greedy Policies and the resulting Greedy Convex Hull. As can be seen, the Greedy Convex Hull is significantly reduced in comparison to the Feasible Region.

To expand the Greedy Convex Hull, we add to the subset a group of policies denoted here by Round Robin. Points 5 and 6 in Fig. 2 are examples of such policies. Round Robin policies choose the next client to be scheduled (from the pool of allowed u_t) in a cyclic order. Considering $T = 5$ and $N = 3$, a sample schedule would be $(1,2,3,1,2)$. Once the cyclic order is defined for the first frame, it is repeated in

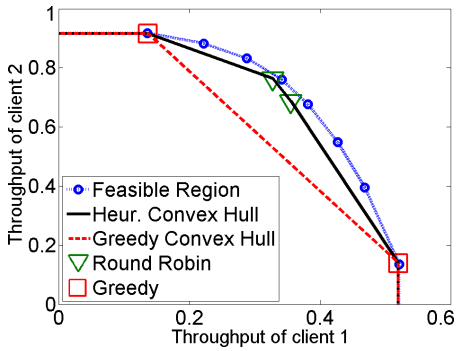

Fig. 3. Comparison of the different convex hulls for $T = 7$, $N = 2$, $d = 3$, $p_1 = 0.1$ and $p_2 = 0.3$. The throughput vectors of the Greedy and Round Robin Policies are obtained by simulating the network for 5×10^4 frames.

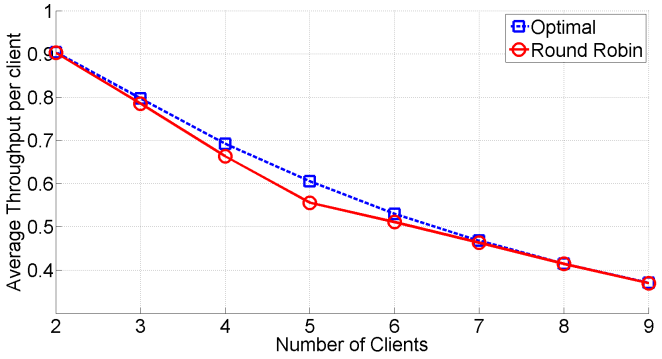

Fig. 4. Simulation results of the Average Throughput per client in networks with $T = 10$, $d = 5$ and $p_i = 1/3, \forall i$. For each value of N, simulations run for 10^6 frames.

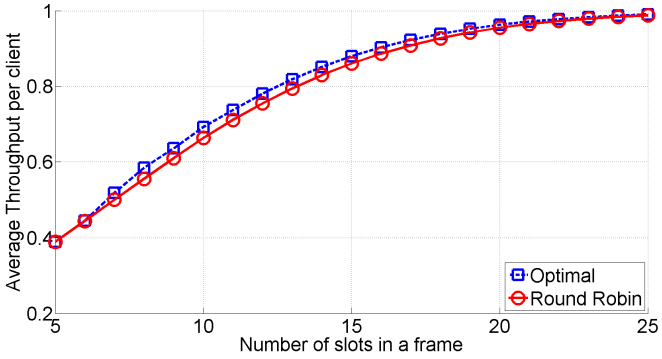

Fig. 5. Simulation results of the Average Throughput per client in networks with $N = 4$, $d = 5$ and $p_i = 1/3, \forall i$. For each value of T, simulations run for 10^6 frames.

the following frames. Intuitively, the objective is to postpone the retransmission of packets with pending feedback, in order to avoid the state transition represented by (5), which is associated to zero reward. As illustrated in Fig. 3, the simulated throughput vectors of the Round Robin policies expand the Greedy Convex Hull into the Heuristic Convex Hull. Notice that there are $N!$ Round Robin polices in the subset, one for each permutation of clients.

The Round Robin policy is not optimal for every network setting. However, as shown in Fig. 3, even when it is not optimal, this policy can be used to expand the Greedy Convex Hull. Naturally, the shorter the distance between the throughput vector of the Round Robin policy and the border of the Feasible Region, the larger is the Heuristic Convex Hull. To examine this distance in different network settings, we present next some simulation results.

Let the performance metric be $mean_i(\hat{q}_i^{\eta})$, which is the average throughput per client of policy η. Consider two policies: i) Round Robin with clients in order of increasing index; ii) Optimal Policy obtained from the DP for $\vec{\alpha} = (1)_{i=1}^{N}$. Figs. 4 and 5 compare both policies in a variety of settings. Simulations run for 10^6 frames. The results show that the average throughput per client of the Round Robin policy is comparable to the Optimal in every simulation, what suggests that the performance of the Round Robin policy is in fact close to the border of the Feasible Region, specifically of the point which represents the performance of the Optimal Policy for $\vec{\alpha} = (1)_{i=1}^{N}$.

B. Heuristic Algorithm and Simulation Results

The goal of the Heuristic Algorithm is to fulfill the minimum delivery ratio requirements by selecting the best policy, within the subset, to be employed in each frame. At the beginning of the kth frame, let the average debt of client i be defined as $a_i(k) := q_i - Q_i(k)/k$, where $(q_i)_{i=1}^{N}$ is the requirement vector and $Q_i(k)$ is the total number of delivered packets. The vector of average debts $(a_i(k))_{i=1}^{N}$ represents the best return a policy could provide. From the same perspective, the throughput vectors, $(\hat{q}_i^{\eta})_{i=1}^{N}$, of each Greedy and Round Robin policies represent their expected returns. By projecting each $(\hat{q}_i^{\eta})_{i=1}^{N}$ on $(a_i(k))_{i=1}^{N}$ and comparing the projection lengths, the algorithm can evaluate which policy is most suited for the kth frame. Fig. 6 illustrates the projection of one throughput vector. A complete description of the Heuristic Algorithm follows.

HEURISTIC ALGORITHM

(i) At the beginning of the frame k, compute for each client:
  - the total number of packets delivered $Q_i(k)$,
  - the average debt $a_i(k) = q_i - Q_i(k)/k$;

(ii) For each policy η in the subset, project the vector $(\hat{q}_i^{\eta})_{i=1}^{N}$ on $(a_i(k))_{i=1}^{N}$;

(iii) During the kth frame, employ the policy with longer projection length.

The Optimal and Heuristic Algorithms are studied via simulations. To examine their performances, the metrics of interest are the achieved throughput vector and the Deadlines Miss Ratio [5]. The DMR is defined as

$$\mathrm{DMR}(k) := \frac{1}{N}\sum_{i=1}^{N} \max\{d_i(k), 0\}, \qquad (10)$$

and represents the dynamic behavior of the algorithm regarding the delivery debt.

Simulation runs were provided for 5×10^4 frames in two distinct settings. Figs. 7 and 8 consider the network with

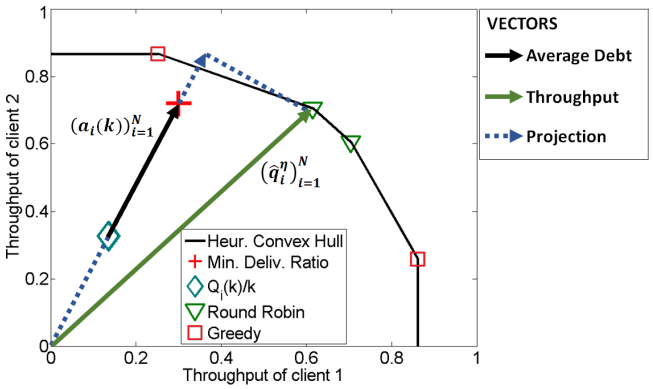

Fig. 6. Projection of the throughput vector of one of the Round Robin policies on the vector of average debts.

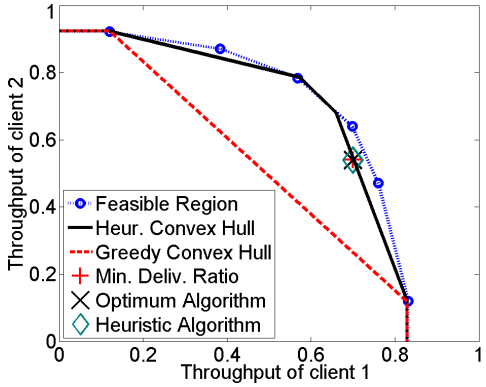

Fig. 7. Comparison of the throughput vectors attained by the Optimal and Heuristic Algorithms in a two-user network with $T = 5$, $d = 3$, $p_1 = 0.3$, $p_2 = 0.4$, $q_1 = 0.7$ and $q_2 = 0.54$. Both algorithms fulfill the minimum delivery ratio requirement.

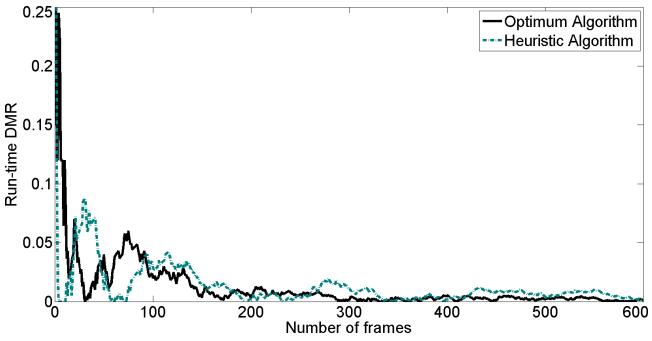

Fig. 8. Dynamic behavior of the simulation in Fig. 7 for the first 600 frames.

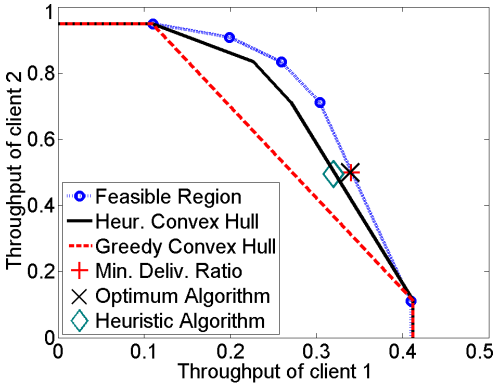

Fig. 9. Comparison of the throughput vectors attained by the Optimal and Heuristic Algorithms in a two-user network with $T = 5$, $d = 2$, $p_1 = 0.1$, $p_2 = 0.45$, $q_1 = 0.34$ and $q_2 = 0.5$. The Optimal Algorithm fulfills the minimum requirement, while the Heuristic Algorithm fails.

a requirement vector placed inside the Heuristic Convex Hull, while Figs. 9 and 10 consider a requirement vector placed between the Heuristic Convex Hull and the Feasible Region. The simulated throughput vectors in Fig. 7 shows that both algorithms can fulfill any constraints that lie inside the Heuristic Convex Hull. Moreover, the DMR in Fig. 8 shows that the Optimal and Heuristic Algorithms drive the throughput vector to the minimum delivery ratio within 200 frames even when this requirement is close to the edge of the Heuristic Convex Hull. In contrast, when the requirement vector is outside the Heuristic Convex Hull but inside the Feasible Region, we see from Figs. 9 and 10 that the Heuristic Algorithm fails to fulfill the delivery requirement but the Optimal Algorithm obtains the desired delivery ratio within a few hundred frames.

VI. CONCLUDING REMARKS

This paper studied the problem of scheduling packets with hard deadlines and delivery ratio constraints in a wireless network with delayed feedback. We extended the analytical model of [5] to account for delayed feedback and used Dynamic Programming to solve for the optimal policy. This solution was the groundwork for the Optimal and Heuristic Algorithms. The Optimal Algorithm was shown to be feasibility optimal and both were evaluated through simulations. Results suggested that the low-complexity Heuristic Algorithm can fulfill any delivery ratio requirements that lie inside the convex hull formed by its subset of policies.

Some interesting extensions of this work include consideration of multicast traffic and time-varying channels. Multicast traffic could be approached with the simplifications proposed in [8], in which the system model can be adapted, without loss of generality, to consider each client subscribing to a single flow. Time-varying channels could be considered using the framework proposed in [7].

REFERENCES

[1] D. Bertsekas and R. Gallager, *Data Networks*, 2nd ed. Prentice Hall, 1992.

[2] P. P. Bhattacharya, L. Tassiulas, and A. Ephremides, "Optimal scheduling with deadline constraints in tree networks," *IEEE Transactions on Automatic Control*, vol. 42, pp. 1703–1705, Dec. 1997.

[3] Z. Mao, C. E. Koksal, and N. B. Shroff, "Optimal online scheduling with arbitrary hard deadlines in multihop communication networks," *IEEE/ACM Transactions on Networking*, Nov. 2014.

[4] R. Li and A. Eryilmaz, "Scheduling for end-to-end deadline-constrained traffic with reliability requirements in multihop networks," *IEEE/ACM Transactions on Networking*, vol. 20, pp. 1649–1662, Feb. 2012.

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

[6] I.-H. Hou and P. R. Kumar, "Admission control and scheduling for qos guarantees for variable-bit-rate applications on wireless channels," in *Proceedings of ACM MOBIHOC*, May 2009, pp. 175–184.
[7] ——, "Scheduling heterogeneous real-time traffic over fading wireless channels," in *Proceedings of IEEE INFOCOM*, Mar. 2010, pp. 1–9.
[8] K. S. Kim, C.-P. Li, and E. Modiano, "Scheduling multicast traffic with deadlines in wireless networks," in *Proceedings of IEEE INFOCOM*, May 2014, pp. 2193–2201.
[9] I.-H. Hou, "Broadcasting delay-constrained traffic over unreliable wireless links with network coding," *IEEE/ACM Transactions on Networking*, vol. 23, pp. 728–740, Feb. 2014.
[10] D. Bertsekas, *Dynamic Programming and Optimal Control*, 3rd ed. Athena Scientific, 2005, vol. 1.
[11] M. J. Neely, *Stochastic Network Optimization with Application to Communication and Queueing Systems*. Morgan and Claypool Publishers, 2010.

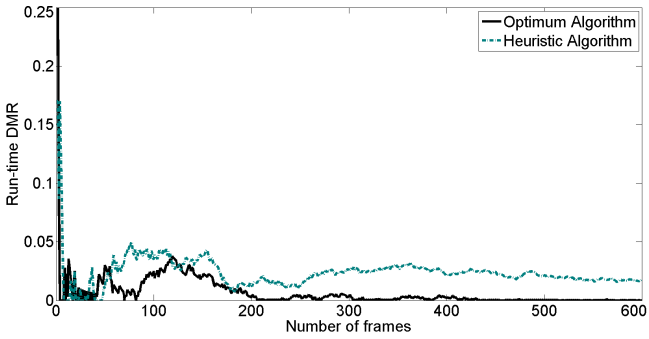

Fig. 10. Dynamic behavior of the simulation in Fig. 9 for the first 600 frames.