A Simple Rate Control Algorithm for Maximizing Total User Utility

Koushik Kar† Saswati Sarkar* Leandros Tassiulas†

† ECE Department
University of Maryland
College Park, MD 20742, USA
{koushik,leandros}@eng.umd.edu

* EE Department
University of Pennsylvania
Philadelphia, PA 19104, USA
swati@ee.upenn.edu

Abstract—**In this paper, we consider the rate control problem with the objective of maximizing the total user utility. It takes into account the possible differences in user requirements, and also provides a framework for achieving a wide range of fairness objectives. We propose a simple algorithm for achieving the optimal rates for this problem. The algorithm can be implemented in a distributed way and does not require the network to know the user utility functions. In our algorithm, the network communicates to the user the number of congested links on the user's path, and the user (end-host) adjusts its rate accordingly, taking into account its utility function and the network congestion feedback. We show through analysis and experimentation that our algorithm converges to the optimum rates.**

I. INTRODUCTION

Effective rate control of *elastic* traffic sources [6] is required in order to control congestion in a communication network. Elastic traffic sources are those which do not require a fixed rate of service and can adjust their transmission rates based on the congestion level of the network. Examples of elastic traffic sources include internet traffic sources using TCP, and sources using ABR service in ATM networks. A rate control strategy should ensure that the network is used efficiently, while guaranteeing that the traffic offered to the network is such that the congestion at the network resources remain within an acceptable level. Besides these, it is also desirable that the rate control algorithm would ensure that the available network resources are shared by the competing streams of traffic in some fair manner.

There can be many different measures of fairness, one of the most well-known being max-min fairness [1]. Most of the notions of fairness explored in the literature treat all users equally. The differences in rate allocations are only due to the different path bandwidths and processing capability limitations. However, users in general have widely varying bandwidth requirements, and therefore it is desirable that any fair rate allocation scheme would take into account this heterogeneity in user requirements.

Fairness definitions can be generalized in a nice way by using utilities. Utility of an user is a function connecting the bandwidth given to the user with the "value" associated with the bandwidth (note that throughout the paper, the terms "user", "session" and "source" are used synonymously). The utility could be some measure of say, the perceived quality of audio/video, the user satisfaction, or even the amount paid by the user for the bandwidth allotted to it, and could be different for different users. Thus it provides a framework to differentiate among users on the basis of their requirements and/or revenues. One possible fairness objective, as advocated recently by Kelly in [7], is to allocate bandwidths such that they maximize the sum of the user utilities (assumed to be concave functions), subject to the link capacity constraints. This is also the problem that we address in this paper. The rate control algorithm that we propose achieves the optimal rates for this total user utility maximization problem. The algorithm is distributed and does not require the network to know the user utility functions. Note that this algorithm can be used for achieving a wide variety of fairness objectives by choosing the user utility functions appropriately (for example, if all the utility functions are logarithmic and same for all users, the achieved rates are proportionally fair [7]).

In the algorithm proposed in this paper, the network communicates to the user *the number of congested links on the user's path*. On congestion, the user decreases its rate based on this network feedback; otherwise it increases its rate based on the derivative of its utility function. An attractive feature of this algorithm, particularly from a practical perspective, is the simplicity of both the user and network (link) algorithms. Moreover, note that the congestion information that the network needs to provide the user (i.e., the number of congested links on its path) can be conveyed in only $\lfloor \log_2 \bar{L} \rfloor + 1$ bits, where $\bar{L}$ is the maximum number of links on a user's path. This implies that in most real networks, including the internet, just one byte in the data packet/ACK packet header should be sufficient to carry the network congestion feedback (note that one byte would allow 255 links on a user's path).

Several alternative approaches to this system utility maximization problem have been proposed in recent literature [9],[8],[10],[11]. These approaches are surveyed in Section VII, where we also compare them with our approach. It is also worth noting here that whereas all these existing approaches require *strict* concavity of utility functions for guaranteed convergence, we require only concavity. Thus linear utility functions are allowed in our case, but not in the earlier ones. Moreover, our approach also guarantees convergence for a wide class of non-differentiable functions, which are outside the framework of the previous approaches.

The paper is organized as follows. In the next section we define our problem formally. In Section III we present the basic algorithm and describe a distributed implementation of it. Section IV presents the convergence analysis for our algorithm, while Section V describes some experimental results. In Section VI, we discuss a few issues related to our approach. We compare our approach with the existing approaches in Section VII, and conclude in Section VIII.

II. Problem Statement

Consider a network consisting of a set L of unidirectional links, where a link $l \in L$ has capacity c_l. The network is shared by a set S of unicast sessions (users). Let $L_s \subseteq L$ denote the set of links used by session $s \in S$. Also let $S_l \subseteq S$ denote the set of sessions that use link $l \in L$. Each session has a minimum required transmission rate $b_s \geq 0$, and a maximum required transmission rate $B_s < \infty$. Moreover, each session s is associated with a utility function $U_s : \Re_+ \to \Re$, which is assumed to be concave, continuous, bounded and increasing in the interval $X_s = [b_s, B_s]$. Thus session s has a utility $U_s(x_s)$ when it is transmitting at a rate x_s, where $x_s \in X_s$. Our objective is to maximize the "social welfare", i.e., sum of the utilities over all the sessions, subject to the link capacity constraints. The problem can be posed as:

P:

$$\max \sum_{s\in S} U_s(x_s) \tag{1}$$

subject to

$$\sum_{s\in S_l} x_s \leq c_l \qquad \forall l \in L \tag{2}$$

$$x_s \in X_s \qquad \forall s \in S \tag{3}$$

The constraints (2) indicate that the total rate of the sessions using a link cannot exceed the capacity of the link.

Let $x = (x_s, s \in S)$ denote the vector of the session rates. Also let X_S denote the entire region in the $|S|$-dimensional space where x is constrained to lie due to (3), i.e., $X_S = \{(x_1, ..., x_{|S|}) : x_s \in X_s \;\; \forall s \in S\}$. Thus the set of constraints in (3) can be equivalently written as $x \in X_S$. Let X_L denote the region in the $|S|$-dimensional space defined by the $|L|$ constraints in (2). Thus $X_L = \{(x_1, ..., x_{|S|}) : \sum_{s\in S_l} x_s \leq c_l \;\forall l \in L\}$. Let $X = X_S \cap X_L$. Thus the problem **P** can be equivalently written as the maximization of the sum of the utility functions, as stated in (1), subject to $x \in X$. Note that X_S is compact (i.e., closed and bounded) and convex, and X_L is convex. Thus X is compact and convex. We will assume that the problem **P** is feasible, i.e., X is nonempty. Thus an optimal solution exists, although it may not be unique.

III. A Distributed Algorithm

Now we present a distributed algorithm that solves the optimization problem formulated in Section II, and describe how it can be implemented in a real network. The convergence analysis of the algorithm is presented in the next section. The basic idea behind the algorithm is derived from [5], where an iterative *subgradient*[1] based optimization method has been proposed for a very general class of convex optimization problems. However, the optimization procedure in [5], if implemented in our case, would require centralized coordination, and is therefore not practical for large networks (we describe more details on this algorithm in Section VI). The algorithm presented below is a modified version of the algorithm in [5] which is amenable to distributed implementation and yet retains the convergence properties of the original algorithm.

A. An iterative optimization algorithm

Before we describe the algorithm, consider two positive sequences $\{\alpha_n\}$ and $\{\beta_n\}$, with the following properties:

$$\lim_{n\to\infty} \alpha_n = 0 \qquad \sum_{n=1}^{\infty} \alpha_n = \infty \tag{4}$$

$$\lim_{n\to\infty} \beta_n = 0 \qquad \sum_{n=1}^{\infty} \beta_n = \infty \tag{5}$$

$$\lim_{n\to\infty} \frac{\alpha_n}{\beta_n} = 0 \tag{6}$$

For example, $\alpha_n = (1/n)$ and $\beta_n = (1/\sqrt{n})$ satisfy (4)-(6).

Now consider an iterative procedure to solve **P**, where $x_s^{(n)}$, the rate of session $s \in S$ at the nth step, is updated as follows

$$x_s^{(n+1)} = \begin{cases} [\, x_s^{(n)} + \alpha_n U'(x_s^{(n)}) \,]_{X_s} & \text{if } \tilde{e}_s^{(n)} = 0 \\ [\, x_s^{(n)} - \beta_n \tilde{e}_s^{(n)} \,]_{X_s} & \text{if } \tilde{e}_s^{(n)} > 0 \end{cases} \tag{7}$$

where $[\,\cdot\,]_{X_s}$ denotes a projection[2] on the set X_s, and

$$\tilde{e}_s^{(n)} = \sum_{l\in L_s} e_l^{(n)} \tag{8}$$

and

$$e_l^{(n)} = \mathbf{1}\Big(\sum_{s\in S_l} x_s^{(n)} > c_l\Big) \tag{9}$$

$$= \begin{cases} 0 & \text{if } \sum_{s\in S_l} x_s^{(n)} \leq c_l \\ 1 & \text{if } \sum_{s\in S_l} x_s^{(n)} > c_l \end{cases} \tag{10}$$

The function $\mathbf{1}(\cdot)$ in (9) is an indicator function. Thus the variable $e_l^{(n)}$ can be interpreted as the "link congestion indicator" for link l. Note that $\tilde{e}_s^{(n)}$ is the number of "congested links" on the path of the session, i.e., the links on its path for which the capacity constraints are violated. Therefore, as (8) states, $\tilde{e}_s^{(n)}$ is the sum of the link congestion indicators on the path of session s. Also note that the session rate update procedure described in (7) inherently assumes that the function U_s is differentiable in X_s. This, in general, is not necessary. If $U_s'(x_s)$ does not exist as some point $x_s \in X_s$, it can be replaced by a subgradient of U_s at x_s.

The update procedure of (7) basically states that when any of the links on the path of a session is congested, it backs off by decreasing its rate, whereas when none of the links on its path is congested, it increases its rate according to the derivative of its utility function at that point. Here, α_n and β_n denote the step-sizes for increment and decrement, respectively. Note that

[1] A subgradient, defined in the context of convex/concave functions, can be viewed as a generalized gradient, and may exist even if the gradient does not. See the appendix for the formal definition.

[2] Since $X_s = [b_s, B_s]$, thus for any scalar y, $[y]_{X_s} = \min(B_s, \max(b_s, y))$.

when any of the links on the path of session s is congested, the reduction in the rate of the session is proportional to the number of congested links on the session's path.

As we will show in the next section, the step-sizes α_n and β_n need to satisfy (4)-(6) for the algorithm to have guaranteed convergence. Note that (6) roughly implies that the increment of the rate of a session (when there is no congestion), needs to be (asymptotically) much smaller as compared to the decrement (when there is congestion).

B. Distributed implementation

Now let us see how the iterative procedure described above can be implemented in a distributed way, using the links and the session sources/receivers like processors in a distributed computation system. Let link l be responsible for keeping track of e_l. Also assume that the rate computation for a session (according to (7)) is carried out at the source of the session.

The algorithm described in the last subsection is a synchronous algorithm. In a large network, due to practical considerations, we would like to implement an asynchronous version of this algorithm. In the asynchronous version, the algorithm remains the same except that all updates occur asynchronously, and are triggered when rates/congestion indicators change, or after some fixed time intervals, or a combination of both.

Note that a session needs to know only the total number of congested links on its path and not the exact set of congested links. To see an example of a distributed asynchronous implementation of our algorithm, consider an ACK-based protocol where the ACK packets (going from the receiver to the source of a session) use the same path as the data packets but in the backward direction. Let each ACK packet have a congestion notification field E. When it goes through a link l, the link adds the congestion indication bit to the entry in the E-field of the ACK packet. Thus when the ACK packet reaches the source node, the field E of the packets contains the number of congested links on the session's path, which is used in the computation of the new rate at the source. Let each data packet contain a field R indicating the current rate of packet transmission for the session. The links on the path of the session read the field R in order to know the current rate for that session. These rate values are used to update the link congestion indicator. The link and session algorithms are described in Figure 1.

Conditions (4)-(6) are required for the iterative process to converge to the optimum. In reality, however, it may not be feasible to decrease the step-sizes or the step-size ratios beyond a particular point. For the case when step-sizes α_n and β_n are kept fixed, a slightly weaker convergence result holds, as we state in the next section. A similar result also holds when the step-sizes may not be constant but converge to some positive values.

One drawback of the algorithm described in Figure 1 is that the actual rates need to be communicated from the users to the links. In practice, the total rate of traffic on a link can be estimated, and this estimated rate can be used to update the link congestion indicator. In all the experiments that we have carried out, our algorithm converged to the optimum rates even with this modification (see Section V).

Link l 's algorithm

1. Read the R field of all data packets going through the link to know the current session rates.

2. Periodically update e_l as

$$e_l^{(n+1)} = \begin{cases} 0 & \text{if } \sum_{s \in S_l} x_s^{(n)} \le c_l \\ 1 & \text{if } \sum_{s \in S_l} x_s^{(n)} > c_l \end{cases}$$

3. Add e_l to the E field of all ACK packets going through the link.

Session s 's algorithm

1. Read the E field of all ACK packets to know the current number of congested links on the path, and accordingly update $\tilde{e}_s$.

2. Periodically compute the new rate as

$$x_s^{(n+1)} = \begin{cases} [\, x_s^{(n)} + \alpha_n U'(x_s^{(n)}) \,]_{X_s} & \text{if } \tilde{e}_s^{(n)} = 0 \\ [\, x_s^{(n)} - \beta_n \tilde{e}_s^{(n)} \,]_{X_s} & \text{if } \tilde{e}_s^{(n)} > 0 \end{cases}$$

3. Send traffic at the current rate x_s setting the field R to x_s.

Fig. 1. Link and session algorithms

IV. Convergence Analysis

Now we investigate the convergence properties of the algorithm outlined in the last section.

A. Assumptions

We prove the convergence of our algorithm for the synchronous case only. In addition to the assumptions already mentioned in the last section, we will make a few additional assumptions for the convergence analysis.

Assumption 1: (Bounded slope) For every $s \in S$, $a_s \le U_s'(x_s) \le A_s \quad \forall x_s \in X_s$ where $a_s > 0$ and $A_s < \infty$.

If U_s is non-differentiable in X_s (i.e., U_s' does not exist at all points in X_s), we will assume that Assumption 1 holds for all subgradients of U_s in X_s.

Assumption 2: (Interior point) Let $\text{int}(X_L)$ denote the interior of the set X_L. Then $\text{int}(X_L) \cap X_S \neq \phi$.

It is easy to see that the interior point assumption holds if $c_l > 0 \quad \forall l \in L$, $b_s = 0 \quad \forall s \in S$ and $B_s > 0 \quad \forall s \in S$. Also note that feasibility of the problem **P** is also implied by the above assumption.

For the sake of simplicity of the analysis, we make an additional assumption in this paper, as stated below. However, this assumption is not necessary for the convergence results to hold. Refer to [13] for the proof of convergence for the more general case.

Assumption 3: (Unique optimum) The problem **P** has a unique optimal solution, x^*.

Note that Assumption 3 holds when the all the user utility functions are strictly concave. In the following, we also assume that the initial rate vector $x^{(0)}$ is feasible, i.e., $x^{(0)} \in X$.

B. Convergence with diminishing step-sizes

Let $U(x) = \sum_{s \in S} U_s(x_s)$ be the overall user utility, and $U^* = U(x^*)$ be the corresponding optimal value. Now we proceed to state the convergence result for our algorithm, when the step-size sequences $\{\alpha_n\}$ and $\{\beta_n\}$ satisfy (4)-(6).

Theorem 1: For the iterative procedure (7)-(10), and with the step-sizes satisfying (4)-(6), the sequence of rate vectors $x^{(n)}$ converges to x^*, the unique optimal solution of **P**.

We prove the above theorem in the Appendix. Note that from the continuity of U it follows that $\lim_{n\to\infty} U(x^{(n)}) = U^*$.

To see what the convergence result would be for the case where the optimum solution is non-unique, let $\rho(x, Y) = \min_{y \in Y} ||x - y||$ denote the Euclidean distance of a point x from any set Y. If X^* is the set of optimal solutions of **P**, then the convergence result takes the following form [13]: $\lim_{n\to\infty} \rho(x^{(n)}, X^*) = 0$. From the continuity of U, it follows that $\lim_{n\to\infty} U(x^{(n)}) = U^*$, as before.

C. Convergence with constant step-sizes

If the step-sizes are constant, we can not guarantee convergence to the optimum in the sense stated in Theorem 1. However, it is possible to show a slightly weaker result, as stated below. Let $C_\delta(x^*)$ be a sphere of radius δ around x^*, i.e., $C_\delta(x^*) = \{x : ||x - x^*|| \leq \delta\}$.

Theorem 2: Consider the iterative procedure (7)-(10) with $\beta_n = \beta$ and $\alpha_n = \alpha = \eta\beta$ for all n. Then given any $\delta > 0$, there exists $\tilde{\beta}_\delta > 0$ and $\tilde{\eta}_\delta > 0$, such that for any α,β satisfying $0 < \beta < \tilde{\beta}_\delta$ and $0 < (\alpha/\beta) = \eta < \tilde{\eta}_\delta$,

$$\lim_{n\to\infty} \rho(x^{(n)}, C_\delta(x^*)) = 0 \tag{11}$$

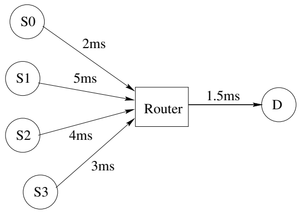
The proof of the above theorem is along the same lines as that for Theorem 1, and can be found in [13]. The above theorem implies that given any neighborhood around the optimum, we can choose the step-size β and the step-size ratio η small enough so that our algorithm (with constant step-sizes) converges to that neighborhood[3].

V. Experimental Results

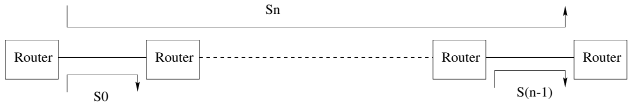
Next we study the convergence properties of our algorithm through simulation experiments carried out on some simple networks. Our simulations are carried out in an asynchronous time-varying environment. In all the experimental results presented here, the session rates are not explicitly conveyed to the links, and the links set the link congestion indicators based on the estimated rates.

Figure 2 shows the two networks that we consider, both adopted from [12]. Figure 2 (a) shows 4 sources sending traffic to a single destination. The sources send traffic to a common router through individual access links (each of capacity 20 packets/ms) which then sends the traffic on the shared link (of capacity 12 packets/ms). The numbers shown across the links in the figure are the link propagation delays (we neglect any processing delays in our simulations). The multilink network consists of one long session and several short sessions. In the multilink network, the capacities and the propagation delays for the network

[3]For a given δ, the values of $\tilde{\beta}_\delta$, $\tilde{\eta}_\delta$ depend on the parameters of the problem **P**, and can be calculated given a particular instance of the problem (see [13]).

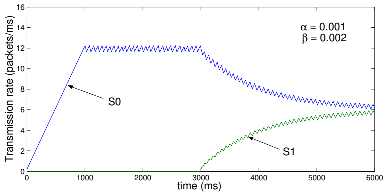


(a) A single-link network

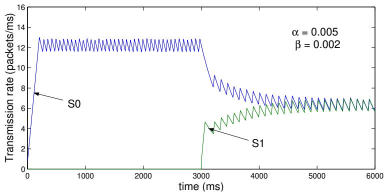


(b) A multilink network

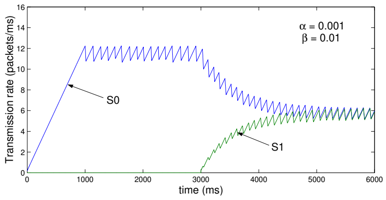
Fig. 2. Two example networks



(a)



(b)



(c)

Fig. 3. Effect of α, β

links are 12 packets/ms and 1.5 ms respectively, while the same for the access links (not shown in figure) are 20 packets/ms and 2 ms respectively. The user utility functions are $w_s \ln x_s$ (thus the optimal allocation is weighted proportionally fair), with the minimum and maximum rates being 0.1 packets/ms and 20 packets/ms, respectively (for the values of the weights w_s, see later). In this section, we assume that the scheduling policy is FIFO. All the simulation results shown in this paper are for constant step-sizes α, β.

First we investigate the effect of α, β on the convergence properties of the algorithm. Consider the single-link network with only the sources S1 and S2, with $w_1 = w_2 = 12$. S1 starts sending traffic at time t=0 while S2 starts at t=3 secs. Figure 3 shows the plots of the transmission rates for three different sets of values of α, β. Note that the plotted curves show some

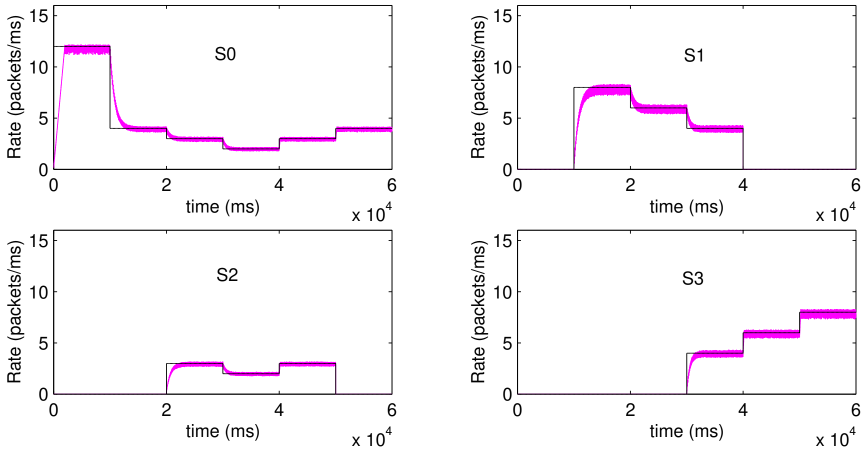


Fig. 4. Convergence for the single-link network. (The straight lines are the optimal (theoretical) rates)

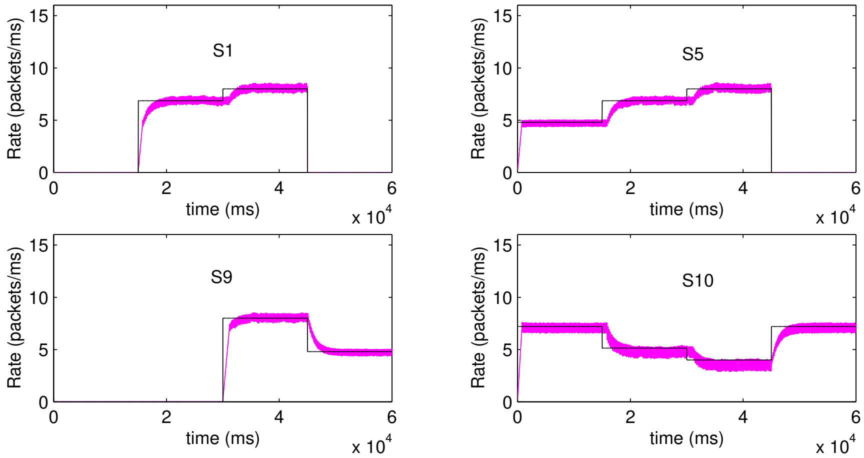


Fig. 5. Convergence for the multilink network. (The straight lines are the optimal (theoretical) rates)

small and regular fluctuations. This can be explained as follows. When the total traffic is close to the link capacity, the link congestion indicator fluctuates rapidly between 0 and 1, as can be expected from intuition. Since we are using constant step-sizes (instead of step-sizes satisfying (4)-(6) required for exact convergence), the fluctuations of the link congestion indicators translate to fluctuations in the session rates. These fluctuations (around the optimal values) for the case of constant step-sizes can also be expected from Theorem 2. In the figure, comparing (a) with (b) and (a) with (c), we observe that as expected, larger values of α, β lead to faster convergence (however, note that the time taken for a session arriving at an unloaded link to reach the full link capacity depends only on α, as we see for session S1). However, there is a tradeoff involved here, since making α, β large also makes the rate fluctuations (around the optimal values) larger. In practice, we would like to have large step-sizes initially (to ensure fast convergence) and small step-sizes later (to reduce fluctuations when the rates are close to the optimal values). Note that we would also like to have a small $\frac{\alpha}{\beta}$ to ensure that the algorithm converges close to the optimal solution (Theorem 2). However, in practice, setting $\frac{\alpha}{\beta}$ to a very small value could reduce the average throughput, as one would intuitively expect. Moreover, that might also amount to making α too small (thus slowing down convergence time) or making β too large (thus increasing fluctuations). In the rest of the simulations presented in this section, $\frac{\alpha}{\beta}$ is set to 0.2.

Consider again the single-link network but with all the 4 sources, as shown in Figure 2 (a), the weights w_s of the sources being (6,12,6,12). S0 is active during 0s-60s, S1 during 10s-40s, S2 during 20s-50s, and S3 during 30s-60s. Figure 4 shows the transmitted rates for the 4 sources during the interval 0s-60s, along with the optimal (theoretical) rates, shown by straight lines (in the figure, α = 0.001 and β = 0.005). The thickening of the plotted curves are due to small, regular but rapid fluctuations of the transmitted rates, the reasons for which have already been discussed in the previous paragraph. The plots show that the rates reach the optimal values and fluctuate close to it. As already mentioned, in practice, these fluctuations around the optimal values could be reduced by reducing the step-sizes once the session detects that rates are fluctuating around the same mean value. The link utilization observed in this case was close to 98%.

Next consider the multilink network with 10 links. Session S1 is active during 15s-45s, S5 during 0s-45s, S9 during 30s-60s and S10 (the long session) during 0s-60s. All the other sources are inactive. Figure 5 shows the transmitted rates for the sources

S1, S5, S9, S10 during the interval 0s-60s, along with the optimal rates (α = 0.0005 and β = 0.0025). The weights w_s for S1, S5, S9 are all 12, while that for S10 is 18. These plots too demonstrate the fact that our algorithm achieves rates that are close to the optimal rates in an asynchronous slowly time-varying environment.

VI. Discussion

In this section, we will first describe the optimization method proposed by Poljak in [5] (which motivated the development of our algorithm) and point out its differences with our algorithm. Then we investigate the question whether we could reduce the congestion feedback from the network (to the session) in our scheme even further, say to just one bit.

A. *Poljak's algorithm*

In [5], the author proposes an iterative primal algorithm for convex constrained minimization problems. As shown in [5], the algorithm converges to the optimal set of solutions under some fairly general assumptions. According to the algorithm presented in [5], the update procedure in our case would be

$$x^{(n+1)} = \begin{cases} [\, x^{(n)} + \gamma_n \frac{\nabla U(x^{(n)})}{||\nabla U(x^{(n)})||} \,]_{X_S} & \text{if } x^{(n)} \in X_L \\ [\, x^{(n)} - \gamma_n \frac{\tilde{e}^{(n)}}{||\tilde{e}^{(n)}||} \,]_{X_S} & \text{if } x^{(n)} \notin X_L \end{cases}$$

or equivalently, stated in terms of session updates (compare with (7)),

$$x_s^{(n+1)} = \begin{cases} [\, x_s^{(n)} + \gamma_n \frac{U_s'(x^{(n)})}{||\nabla U(x^{(n)})||} \,]_{X_s} & \text{if } x^{(n)} \in X_L \\ [\, x_s^{(n)} - \gamma_n \frac{\tilde{e}_s^{(n)}}{||\tilde{e}^{(n)}||} \,]_{X_s} & \text{if } x^{(n)} \notin X_L \end{cases}$$

where the step-sizes γ_n satisfy the following conditions

$$\lim_{n\to\infty} \gamma_n = 0 \qquad \sum_{n=1}^{\infty} \gamma_n = \infty \tag{12}$$

However, this update procedure can not be implemented in a distributed and scalable way. Firstly, due to the presence of the terms $||\nabla U(x^{(n)})||$ and $||\tilde{e}^{(n)}||$ in the session rate-update procedure, each session needs to have some up-to-date information from every other user and from the links. Secondly, note that the session also needs to know if $x^{(n)} \in X_L$ or not in order to decide whether to increase its rate or to back-off. Thus the user would need to know if there is some link in the network that is congested, and back-off if there is (even if the session is not using the link). This is possible in a centralized system with a central server keeping track of all link states and communicating them to the sessions. However such a solution does not seem feasible for implementation in a large network. Our algorithm, on the other hand, is amenable to a completely decentralized implementation. We, however, require two different step-sizes α and β instead of the single step-size γ.

B. *One-bit congestion feedback*

Although using one byte of the data packet/ACK packet header (which is sufficient for our algorithm) for network congestion notification does not introduce a significant overhead, it is still interesting to investigate if it could be reduced even further, say to just a single bit. In that case the algorithm could be implemented just with the proposed Explicit Congestion Notification (ECN) bit. The obvious modification to our algorithm in that case would be to do an OR of all the congestion indicator bits of the links on a session's path (rather than the SUM). We show by a simple example that with this modification, our algorithm may not converge to the optimal solution. Consider a simple network with two links and three sessions, as shown in Figure 6. Sessions s_1 and s_2 use one link each, while session s_3 uses both the links (the links have the same capacity). The sessions have identical utility functions, minimum and maximum rates (the minimum rate is zero while the maximum rate is much larger than the link capacities). Let the step-sizes α_n and β_n be the same for all the sessions. Then if the sessions start at time 0 and always update their rates synchronously, then it is not too difficult to see that all the sessions always get the same rates, irrespective of what their common utility function is. Thus the longer session s_3 eventually gets half of the capacity of each link (the same as the shorter sessions), whereas it should have received lesser. Thus although the modified algorithm converges in this case, it converges to a value different from the optimum. In general, even if the congestion indication field is more than one bit but less than $\lfloor \log_2 \bar{L} \rfloor + 1$ bits (the minimum required in our case), it seems unlikely that some simple modification of this algorithm would be able to provide deterministic convergence guarantees in such cases. Whether there is some alternative approach that achieves this remains an interesting open question.

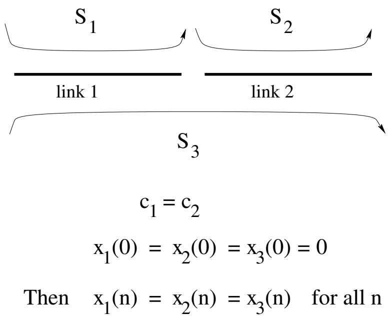


Fig. 6. An example with one-bit congestion feedback

In practice the number of bits allocated to the congestion indication field would be fixed by the protocol. If this field is less than the minimum required in our case, one possibility is to go for heuristic approaches. Thus instead of communicating the required congestion information directly, it is possible to convey it some implicit way using fewer number of bits (by some probabilistic marking policy, for instance). There could be several such heuristic approaches, although their convergence properties, both theoretical and experimental, require further investigation.

VII. Related Work

Next we describe some existing algorithms for the same problem that we are trying to solve, and compare these with our algorithm on several aspects.

In [8], Low et al. propose an algorithm based on the dual approach (note that our algorithm is a primal algorithm) for the same problem. Here each link in a network maintains a congestion price ("link price"), which it calculates based on the aggregate rate on that link. The network conveys to the user the sum of the link prices on its path, which then chooses a rate so as to maximize its profit based on this "session price". A problem with the practical implementation of this algorithm is that the congestion prices (which are basically the dual variables) are real numbers and could vary over a wide range. This poses a difficulty in communicating the price to the end-host using a small number of bits. However this is not a problem in our approach, as we have discussed before. Also note that in certain cases, computing a maximizer (as required by the user in the algorithm in [8]) may be significantly more difficult than just computing a derivative (as required in our algorithm).

In [12], the authors suggest a randomized marking based implementation of the algorithm in [8], that uses only one bit for the network congestion feedback. Here the single congestion indication bit is marked probabilistically (and independently) at each link on the user's path. The marking probability is such that the user can estimate the session price by seeing the proportion of marked packets. However, the authors do not provide any proof of convergence, (note that even if the algorithm converges, the convergence would be in some probabilistic sense). On the other hand, our algorithm has guaranteed deterministic convergence. Our algorithm needs to have a larger congestion indication field as compared to the algorithm in [8], but as we have argued before, just one byte in the packet header does not seem to be a significant overhead. Moreover, the randomized marking policy of [12] can be applied to our algorithm too, and thus the number of congested links can be implicitly conveyed through that single congestion bit.

In [9], the authors propose both primal and dual algorithms for this system utility maximization problem. These algorithms allow a wide variety of congestion price functions. However, the algorithms in [9] solve only an approximate version of the original problem rather than the actual problem. Moreover, the price functions are required to be continuous, and this poses a difficulty in conveying the price using a small number of bits without sacrificing precision, as we have already discussed.

Another related, but different, approach is proposed in [10]. In this approach, the user adjusts its rate based on the proportion of marked packets or end-to-end (measurable) losses. However, the convergence has been proved under certain simplifying assumptions, some of which are not likely to hold in practice.

In [11], the authors present a window-based flow control approach for the same problem. Here the users choose some weights and the window-based flow control scheme, on convergence, allocates rates that are proportionally fair with respect to those weights. The user chooses the next set of weights based on the earlier weights and the allocated rates. The information on the state of congestion in the network is conveyed to the user implicitly through the packet round-trip times. A drawback of this algorithm is that it is a two-level optimization algorithm, i.e., after choosing a set of weights, the users have to wait till the rates have converged or are close to it (which may not always be easy to detect, since the convergence may be asymptotic), before choosing a new set of weights. Moreover, the authors in [11] provide theoretical convergence guarantees only for the simple case where all the sessions share a single bottleneck link.

Also, as mentioned in Section I, unlike all these previous approaches, our algorithm guarantees convergence even for linear and non-differentiable utility functions.

VIII. Conclusions and Future Work

In this paper, we considered an optimization based approach to flow control. We assumed that each user is associated with a (possibly different) utility function, and presented a simple distributed algorithm that achieves the socially optimal rates. We proved the convergence of our algorithm for the case of synchronous updates, and experimentally demonstrated its convergence in an asynchronous environment.

There are several interesting and challenging questions that merit further investigation. Important theoretical questions include that of investigating the convergence of the algorithm for the asynchronous case, and for the case where the session rates are estimated rather than being explicitly communicated from the users to the links. On the practical side, convergence and other associated properties of the algorithm need to be studied extensively for larger and more complex networks.

References

[1] D. P. Bertsekas, R. G. Gallagher, *Data Networks*, Prentice Hall, 1992.
[2] D. P. Bertsekas, *Nonlinear Programming*, Athena Scientific, 1995.
[3] D. P. Bertsekas, J. N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*, Athena Scientific, 1989.
[4] R. T. Rockafellar, *Convex Analysis*, Princeton Univ. Press, 1970.
[5] B. T. Poljak, "A General Method of Solving Extremum Problems", *Soviet Math Doklady*, vol. 8, no. 3, 1967, pp. 593-597.
[6] S. Shenker, "Fundamental Design Issues for the Future Internet", *IEEE Journal on Selected Areas in Communications*, vol. 13, no. 7, 1995, pp. 1176-1188.
[7] F. P. Kelly,"Charging and Rate Control for Elastic Traffic", *European Transactions on Telecommunications*, vol. 8, no. 1, 1997, pp. 33-37.
[8] S. Low, D. E. Lapsley, "Optimization Flow Control, I: Basic Algorithm and Convergence", *IEEE/ACM Transactions on Networking*, vol. 7, no. 6, December 1999.
[9] F. Kelly, A. Maulloo, D. Tan,"Rate Control for Communication Networks: Shadow Prices, Proportional Fairness and Stability ", *Journal of Operations Research Society*, vol. 49, no. 3, 1998, pp. 237-252.
[10] S. Kunniyur, R. Srikant,"End-to-End Congestion Control Schemes: Utility Functions, Random Losses and ECN Marks", *Proceedings of Infocom 2000*, March 2000.
[11] R. La, V. Anantharam,"Charge-Sensitive TCP and Rate Control in the Internet", *Proceedings of Infocom 2000*, March 2000.
[12] S. Athuraliya, S. Low, D. Lapsley,"Random Early Marking", Submitted for publication, www.ee.mu.oz.au/staff/slow/research/
[13] K. Kar, S. Sarkar, L. Tassiulas, "A Primal Algorithm for Optimization Based Rate Control for Unicast Sessions", Technical Report TR 2000-22, Institute for Systems Research and University of Maryland, 2000.

Appendix

Since our analysis is based on subgradient methods, let us formally define a subgradient [2](pp. 490) before proceeding further.

Definition 1: Consider a convex and continuous function $h(x)$ defined on a convex set $C \subseteq \Re^k$. Then a vector $w_0 \in \Re^k$ is called a *subgradient* of h at a point $x_0 \in C$ if it satisfies

$$h(x) - h(x_0) \geq (w_0, x - x_0) \qquad \forall x \in C \qquad (13)$$

In general subgradient at a point may be non-unique. The set of subgradients at a point is called the *subdifferential*. If the gradient of $h(x)$ exists at a point, then the subdifferential at that point consists of only one subgradient, which is the gradient at the point.

For each $l \in L$, define $g_l : \Re_+^{|S|} \to \Re$ as $g_l(x) = \sum_{s \in S_l} x_s - c_l$. Now define $g : \Re_+^{|S|} \to \Re$ as $g(x) = \sum_{l \in L} \max(0, g_l(x))$. Then the function g is convex, and $g(x) > 0$ iff $x \notin X_L$. Thus the entire set of constraints in (2) can be replaced by a single constraint $g(x) \leq 0$. Let $v_s^{(n)}$ denote a subgradient of U_s at $x_s^{(n)}$. Thus if U_s is differentiable, then $v_s^{(n)} = U_s'(x_s^{(n)})$. Let $v^{(n)} = (v_1^{(n)}, v_2^{(n)}, ..., v_{|S|}^{(n)})$ be the vector of the subgradients. Thus $v^{(n)}$ is a subgradient of U at $x^{(n)}$. Now consider the vectors $\tilde{e}^{(n)} = (\tilde{e}_1^{(n)}, \tilde{e}_2^{(n)}, ..., \tilde{e}_{|S|}^{(n)})$. Note that $\tilde{e}^{(n)}$ is a subgradient of g at $x^{(n)}$.

Now let us restate our algorithm in terms of the notation just introduced. Note that the iterative update procedure of (7)-(10) can be stated as

$$x^{(n+1)} = \begin{cases} [\, x^{(n)} + \alpha_n v^{(n)} \,]_{X_S} & \text{if } g(x^{(n)}) \leq 0 \\ [\, x^{(n)} + (\alpha_n \tilde{v}^{(n)} - \beta_n \tilde{e}^{(n)}) \,]_{X_S} & \text{if } g(x^{(n)}) > 0 \end{cases} \tag{14}$$

where

$$\tilde{v}^{(n)} = (\tilde{v}_1^{(n)}, \tilde{v}_2^{(n)}, ..., \tilde{v}_{|S|}^{(n)}) \tag{15}$$

and

$$\tilde{v}_s^{(n)} = \begin{cases} v_s^{(n)} & \text{if } \tilde{e}_s^{(n)} = 0 \\ 0 & \text{if } \tilde{e}_s^{(n)} > 0 \end{cases} \tag{16}$$

Now we state a few simple facts that will be useful to us in our analysis. Note that from the definition of a subgradient, it follows that if $h(x)$ is a convex function, then for any point $x_1 \in \{x : h(x) \leq h(x_0)\}$, $(w_0, x_1) \leq (w_0, x_0)$. Also note that from Assumption 1, it follows that there exists some $a > 0$ and $A < \infty$ such that $a \leq ||v^{(n)}|| \leq A$ for all n. Moreover, it is easy to see that for any n such that $g(x^{(n)}) > 0$ (i.e., $x^{(n)} \notin X_L$), $||\tilde{e}^{(n)}||$ is bounded as $1 \leq ||\tilde{e}^{(n)}|| \leq |S||L|$.

Now let $D_\epsilon = \{x : x \in X_L, U(x) \geq U^* - \epsilon\}$. We prove two lemmas before we proceed with the proof of Theorem 1.

Lemma 1: Given any $\epsilon > 0$, there exists $r_\epsilon > 0$ and $\tilde{x}_\epsilon \in \mathrm{int}(D_\epsilon) \cap X_S$ such that the relation

$$||x^{(n+1)} - \tilde{x}_\epsilon||^2 \leq ||x^{(n)} - \tilde{x}_\epsilon||^2 - r_\epsilon \alpha_n$$

holds for all sufficiently large n for which $x^{(n)} \notin D_\epsilon$.

Proof:

Since $\mathrm{int}(X_L) \cap X_S \neq \phi$ (Assumption 2), it follows that there exists $\tilde{x}_\epsilon \in \mathrm{int}(D_\epsilon) \cap X_S$. Choose $\tilde{r}_\epsilon > 0$ such that $x \in D_\epsilon$ for all x satisfying $||x - \tilde{x}_\epsilon|| \leq \tilde{r}_\epsilon$. Now suppose that $x^{(n)} \notin D_\epsilon$ for some n. Consider the two cases:

Case 1: $x^{(n)} \in X_L$:

It follows that $D_\epsilon \subset \{x : U(x) \geq U(x^{(n)})\}$. Thus $(v^{(n)}, x) \geq (v^{(n)}, x^{(n)})$ for all $x \in D_\epsilon$. In particular, choose $x = \tilde{x}_\epsilon - \tilde{r}_\epsilon \frac{v^{(n)}}{||v^{(n)}||} \in D_\epsilon$ (note $||v^{(n)}|| \geq a > 0$). Thus

$$\begin{aligned} (v^{(n)}, x^{(n)}) &\leq (v^{(n)}, \tilde{x}_\epsilon - \tilde{r}_\epsilon \frac{v^{(n)}}{||v^{(n)}||}) \\ &= (v^{(n)}, \tilde{x}_\epsilon) - \tilde{r}_\epsilon ||v^{(n)}|| \end{aligned} \tag{17}$$

Therefore

$$\begin{aligned} &||x^{(n+1)} - \tilde{x}_\epsilon||^2 \\ &= ||[\, x^{(n)} + \alpha_n v^{(n)} \,]_{X_S} - \tilde{x}_\epsilon||^2 \\ &\leq ||x^{(n)} + \alpha_n v^{(n)} - \tilde{x}_\epsilon||^2 \qquad (18) \\ &= ||x^{(n)} - \tilde{x}_\epsilon||^2 + \alpha_n^2 ||v^{(n)}||^2 \\ &\quad + 2\alpha_n (x^{(n)} - \tilde{x}_\epsilon, v^{(n)}) \\ &\leq ||x^{(n)} - \tilde{x}_\epsilon||^2 + \alpha_n^2 ||v^{(n)}||^2 \\ &\quad - 2\alpha_n \tilde{r}_\epsilon ||v^{(n)}|| \qquad (19) \\ &\leq ||x^{(n)} - \tilde{x}_\epsilon||^2 + A^2 \alpha_n^2 - 2a\tilde{r}_\epsilon \alpha_n \qquad (20) \end{aligned}$$

Note that (18) follows from the fact that $\tilde{x}_\epsilon \in X_S$, and (19) follows from (17).

Since $\alpha_n \to 0$, $\alpha_n \leq (a\tilde{r}_\epsilon / A^2)$ when n is sufficiently large. For all such n, from (20), we get

$$||x^{(n+1)} - \tilde{x}_\epsilon||^2 \leq ||x^{(n)} - \tilde{x}_\epsilon||^2 - a\tilde{r}_\epsilon \alpha_n \tag{21}$$

Case 2: $x^{(n)} \notin X_L$:

In this case, $g(x^{(n)}) > 0$. Thus it follows that $D_\epsilon \subset X_L \subset \{x : g(x) \leq g(x^{(n)})\}$. Therefore $(\tilde{e}^{(n)}, x) \leq (\tilde{e}^{(n)}, x^{(n)})$ for all $x \in D_\epsilon$. In particular choose $x = \tilde{x}_\epsilon + \tilde{r}_\epsilon \frac{\tilde{e}^{(n)}}{||\tilde{e}^{(n)}||} \in D_\epsilon$ (note $||\tilde{e}^{(n)}|| \geq 1$ when $g(x^{(n)}) > 0$). Thus

$$\begin{aligned} (\tilde{e}^{(n)}, x^{(n)}) &\geq (\tilde{e}^{(n)}, \tilde{x}_\epsilon + \tilde{r}_\epsilon \frac{\tilde{e}^{(n)}}{||\tilde{e}^{(n)}||}) \\ &= (\tilde{e}^{(n)}, \tilde{x}_\epsilon) + \tilde{r}_\epsilon ||\tilde{e}^{(n)}|| \end{aligned} \tag{22}$$

Also, from (16), it is easy to see that $(\tilde{v}^{(n)}, \tilde{e}^{(n)}) = 0$ and $||\tilde{v}^{(n)}|| \leq ||v^{(n)}|| \leq A$. Therefore,

$$\begin{aligned} &||x^{(n+1)} - \tilde{x}_\epsilon||^2 \\ &= ||[\, x^{(n)} + \alpha_n \tilde{v}^{(n)} - \beta_n \tilde{e}^{(n)} \,]_{X_S} - \tilde{x}_\epsilon||^2 \\ &\leq ||x^{(n)} + \alpha_n \tilde{v}^{(n)} - \beta_n \tilde{e}^{(n)} - \tilde{x}_\epsilon||^2 \\ &= ||x^{(n)} - \tilde{x}_\epsilon||^2 + \beta_n^2 ||\tilde{e}^{(n)}||^2 \\ &\quad + \alpha_n^2 ||\tilde{v}^{(n)}||^2 - 2\beta_n (x^{(n)} - \tilde{x}_\epsilon, \tilde{e}^{(n)}) \\ &\quad + 2\alpha_n (x^{(n)} - \tilde{x}_\epsilon, \tilde{v}^{(n)}) \\ &\leq ||x^{(n)} - \tilde{x}_\epsilon||^2 + |S|^2 |L|^2 \beta_n^2 \\ &\quad + A^2 \alpha_n^2 - 2\beta_n (x^{(n)} - \tilde{x}_\epsilon, \tilde{e}^{(n)}) \\ &\quad + 2\alpha_n (x^{(n)} - \tilde{x}_\epsilon, \tilde{v}^{(n)}) \end{aligned} \tag{23}$$

Since $(\alpha_n / \beta_n) \to 0$, for sufficiently large n,

$$|S|^2 |L|^2 \beta_n^2 + A^2 \alpha_n^2 \leq 2|S|^2 |L|^2 \beta_n^2 \tag{24}$$

Also, from (22), it follows that

$$\begin{aligned} -2\beta_n (x^{(n)} - \tilde{x}_\epsilon, \tilde{e}^{(n)}) &\leq -2\beta_n \tilde{r}_\epsilon ||\tilde{e}^{(n)}|| \\ &\leq -2\beta_n \tilde{r}_\epsilon \end{aligned} \tag{25}$$

Let $B = \max_{s \in S}(B_s - b_s)$. Since $x^{(n)}, \tilde{x}_\epsilon \in X_S$, therefore $||x^{(n)} - \tilde{x}_\epsilon|| \leq |S|B$. Thus

$$\begin{aligned} &2\alpha_n (x^{(n)} - \tilde{x}_\epsilon, \tilde{v}^{(n)}) \\ &\leq 2\alpha_n ||x^{(n)} - \tilde{x}_\epsilon||\, ||\tilde{v}^{(n)}|| \\ &\leq 2\alpha_n |S| B A \end{aligned} \tag{26}$$

Since $(\alpha_n/\beta_n) \to 0$, $2AB|S|\alpha_n \leq \tilde{r}_\epsilon \beta_n$ for sufficiently large n. For all such n, from (25),(26), we get

$$\begin{aligned} &-2\beta_n(x^{(n)} - \tilde{x}_\epsilon, \tilde{e}^{(n)}) + 2\alpha_n(x^{(n)} - \tilde{x}_\epsilon, \tilde{v}^{(n)}) \\ &\leq -2\tilde{r}_\epsilon\beta_n + 2AB|S|\alpha_n \\ &\leq -\tilde{r}_\epsilon\beta_n \end{aligned} \tag{27}$$

Now from (23),(24),(27), we get

$$\begin{aligned} ||x^{(n+1)} - \tilde{x}_\epsilon||^2 \quad \leq \quad & ||x^{(n)} - \tilde{x}_\epsilon||^2 + 2|S|^2|L|^2\beta_n^2 \\ & -\tilde{r}_\epsilon\beta_n \end{aligned} \tag{28}$$

Since $\beta_n \to 0$, $\beta_n \leq \tilde{r}_\epsilon/(4|S|^2|L|^2)$ for sufficiently large n. For all such n, from obtain from (28),

$$||x^{(n+1)} - \tilde{x}_\epsilon||^2 \leq ||x^{(n)} - \tilde{x}_\epsilon||^2 - \frac{\tilde{r}_\epsilon}{2}\beta_n \tag{29}$$

Since $(\alpha_n/\beta_n) \to 0$, $\alpha_n \leq \beta_n$ for sufficiently large n. For all such n, from obtain from (29),

$$||x^{(n+1)} - \tilde{x}_\epsilon||^2 \leq ||x^{(n)} - \tilde{x}_\epsilon||^2 - \frac{\tilde{r}_\epsilon}{2}\alpha_n \tag{30}$$

Considering cases 1 and 2 ((21) and 30)), and letting $r_\epsilon = \min(a\tilde{r}_\epsilon, \tilde{r}_\epsilon/2)$, the lemma follows. □

Lemma 2: Given any $\epsilon > 0$, there exists an infinite sequence $n_{1,\epsilon} < n_{2,\epsilon} < n_{3,\epsilon} < ...$ such that $x^{(n_{i,\epsilon})} \in D_\epsilon$ for all $i = 1, 2, 3, ...$

Proof:

We prove by contradiction. Let us assume that there exists a $N'_\epsilon < \infty$ such that $x^{(n)} \notin D_\epsilon$ for all $n \geq N'_\epsilon$. Choose $N_\epsilon \geq N'_\epsilon$ be such that Lemma 1 holds for all $n \geq N_\epsilon$. Therefore, from Lemma 1, it follows that for all $n \geq N_\epsilon$,

$$||x^{(n+1)} - \tilde{x}_\epsilon||^2 \leq ||x^{(n)} - \tilde{x}_\epsilon||^2 - r_\epsilon\alpha_n \tag{31}$$

for some $r_\epsilon > 0$ and some $\tilde{x}_\epsilon \in \text{int}(D_\epsilon) \cap X_S$. Summing up the inequalities obtained from (31) for $n = N_\epsilon$ to $N_\epsilon + m$, we obtain

$$||x^{(N_\epsilon+m+1)} - \tilde{x}_\epsilon||^2 \leq ||x^{(N_\epsilon)} - \tilde{x}_\epsilon||^2 - r_\epsilon \sum_{n=N_\epsilon}^{N_\epsilon+m} \alpha_n \tag{32}$$

which implies that $||x^{(N_\epsilon+m+1)} - \tilde{x}_\epsilon|| \to -\infty$ as $m \to \infty$, since $\sum \alpha_n$ diverges. This is impossible, since $||x^{(N_\epsilon+m+1)} - \tilde{x}_\epsilon|| \geq 0$. Hence our assumption was incorrect, thus proving the lemma. □

Proof of Theorem 1: Choose an arbitrary $\delta > 0$. Let $\delta' = (\delta/4)$. Define $\tilde{D}_\epsilon = D_\epsilon \cap X_S = \{x : x \in X, U(x) \geq U^* - \epsilon\}$. It follows from Theorem 27.2 of [4] there exists an $\epsilon = \epsilon(\delta') > 0$ such that

$$\tilde{D}_\epsilon \quad \subset \quad \{x : ||x - x^*|| \leq \delta'\} \tag{33}$$

From Lemma 2, there exists an infinite sequence $n_{1,\epsilon} < n_{2,\epsilon} < n_{3,\epsilon} < ...$ such that $x^{(n_{i,\epsilon})} \in D_\epsilon$ for all $i = 1, 2, 3, ...$ Thus there exists an i_1 such that Lemma 1 holds for all $n \geq n_{i_1,\epsilon}$. Also, since $\alpha_n \to 0$, there exists an i_2 such that $\alpha_n < (\delta'/A)$ for all $n \geq n_{i_2,\epsilon}$. Let $i' = \max(i_1, i_2)$. We show that $||x^{(n)} - x^*|| \leq \delta$ for all $n \geq n_{i',\epsilon}$. Pick any $n \geq n_{i',\epsilon}$. There can be three cases:

<u>*Case 1*</u> : $n = n_{j,\epsilon}$ for some $j \geq i'$:

In this case, $x^{(n)} \in D_\epsilon$. Since $x^{(n)} \in X_S$, hence the fact $x^{(n)} \in D_\epsilon$ also implies that $x^{(n)} \in \tilde{D}_\epsilon$. Thus from (33), it trivially follows that

$$\begin{aligned} ||x^{(n)} - x^*|| \quad &\leq \quad \delta' & (34) \\ &< \quad \delta & (35) \end{aligned}$$

<u>*Case 2*</u> : $n = n_{j,\epsilon} + 1$ for some $j \geq i'$:

Note that $x^{(n_{j,\epsilon})} \in \tilde{D}_\epsilon$ (see Case 1). This implies $x^{(n_{j,\epsilon})} \in X = X_S \cap X_L$. Thus

$$\begin{aligned} &||x^{(n)} - x^{(n_{j,\epsilon})}|| \\ &= ||x^{(n_{j,\epsilon}+1)} - x^{(n_{j,\epsilon})}|| \\ &= ||[\, x^{(n_{j,\epsilon})} + \alpha_{n_{j,\epsilon}} v^{(n_{j,\epsilon})}\,]_{X_S} - x^{(n_{j,\epsilon})}|| \\ &\leq ||x^{(n_{j,\epsilon})} + \alpha_{n_{j,\epsilon}} v^{(n_{j,\epsilon})} - x^{(n_{j,\epsilon})}|| \\ &= \alpha_{n_{j,\epsilon}} ||v^{(n_{j,\epsilon})}|| \\ &\leq A\alpha_{n_{j,\epsilon}} \\ &< \delta' \end{aligned} \tag{36}$$

From (36) and the fact that $||x^{(n_{j,\epsilon})} - x^*|| \leq \delta'$ (Case 1), we get

$$\begin{aligned} ||x^{(n)} - x^*|| \quad &\leq \quad ||x^{(n_{j,\epsilon})} - x^*|| + ||x^{(n)} - x^{(n_{j,\epsilon})}|| \\ &< \quad \delta' + \delta' = 2\delta' & (37) \\ &< \quad \delta & (38) \end{aligned}$$

<u>*Case 3*</u> : $n_{j,\epsilon} + 1 < n < n_{j+1,\epsilon}$ for some $j \geq i'$:

Note that $x^{(n')} \notin D_\epsilon$ for all n' satisfying $n_{j,\epsilon} < n' < n_{j+1,\epsilon}$. Therefore, from Lemma 1, it follows that there exists a $\tilde{x}_\epsilon \in \text{int}(D_\epsilon) \cap X_S \subset \tilde{D}_\epsilon$, such that

$$||x^{(n'+1)} - \tilde{x}_\epsilon|| \quad < \quad ||x^{(n')} - \tilde{x}_\epsilon|| \tag{39}$$

for all n' satisfying $n_{j,\epsilon} < n' < n_{j+1,\epsilon}$. Summing up the inequalities obtained for $n' = n_{j,\epsilon} + 1$ to $n - 1$, we obtain

$$||x^{(n)} - \tilde{x}_\epsilon|| \quad < \quad ||x^{(n_{j,\epsilon}+1)} - \tilde{x}_\epsilon|| \tag{40}$$

Since $\tilde{x}_\epsilon \in \tilde{D}_\epsilon$, hence from (33) it follows that $||\tilde{x}_\epsilon - x^*|| \leq \delta'$. Using this fact, (40) and (37),

$$\begin{aligned} &||x^{(n)} - x^*|| \\ &\leq ||x^{(n)} - \tilde{x}_\epsilon|| + ||\tilde{x}_\epsilon - x^*|| \\ &< ||x^{(n_{j,\epsilon}+1)} - \tilde{x}_\epsilon|| + ||\tilde{x}_\epsilon - x^*|| \\ &\leq ||x^{(n_{j,\epsilon}+1)} - x^*|| + ||\tilde{x}_\epsilon - x^*|| + ||\tilde{x}_\epsilon - x^*|| \\ &= ||x^{(n_{j,\epsilon}+1)} - x^*|| + 2||\tilde{x}_\epsilon - x^*|| \\ &< 2\delta' + 2\delta' = 4\delta' \\ &= \delta \end{aligned} \tag{41}$$

From cases 1, 2, 3, ((35), (38) and (41)), if follows that $||x^{(n)} - x^*|| < \delta$ for all $n \geq n_{i',\epsilon}$. By virtue of the arbitrariness of δ, $x^{(n)}$ converges to x^*. □