Price Computation in Random Early Marking (REM)

Sanjeewa Athuraliya and Steven Low
Department of Electrical & Electronic Engineering
University of Melbourne, Australia
{sadsa,slow}@ee.mu.oz.au

Abstract— **We proposed earlier a flow control algorithm derived from solving the dual of a welfare maximization problem. The algorithm however requires communication between network links and sources that is not achievable on the current Internet. We then extended the basic algorithm to a Random Early Marking (REM) scheme which can be implemented using only binary feedback. In this paper we proposed a new price computation algorithm for REM and present simulation results to illustrate its superior performance over the previous versions.**

I. Introduction

We have proposed in [8], [11] a flow control algorithm derived from solving the dual of a welfare maximization problem introduced in [6]. The algorithm however requires communication between network links and sources that is not achievable on the current Internet. In [9] we describe a Random Early Marking (REM) algorithm which is a practical implementation of the basic algorithm in [11] using binary feedback. It can be implemented, e.g., with the proposed explicit congestion notification (ECN) bit in the IP (Internet Protocol) header [3], [13]. The purpose of this paper is to propose and evaluate an enhancement to the REM algorithm of [9].

In the basic algorithm of [8], [11], each link computes a congestion measure we call a 'price' based on the aggregate rate of sources that go through the link. A source adjusts its transmission rate based on the sum of link prices in its path, which must be fed back to the source. The rate required by a link for price computation is the aggregate *source* rate, which is generally different from the *offered* rate at the link (see Section III-A), and hence also must be explicitly communicated from sources to links. REM consists of two extensions, one to simplify the price computation and the other the price feedback. In this paper we propose a new price computation scheme and illustrate its superior performance over the ones in [9].

The current TCP flow control relies on implicit feedback where a source must infer network congestion, *after the fact*, from observed round trip time and loss. To provide more timely feedback, link algorithms have recently been proposed for TCP which probabilistically mark packets based on local congestion [4], [5], [9]. Ideally link algorithm, which feeds back congestion information, and source algorithm, which adapts traffic rate to congestion, are designed jointly and work in concert to steer the network as a whole towards a (possibly moving) desirable operating point. The optimal reaction to a mark depends both on the objective of the flow control and the information a mark conveys [7], [5], [11], [9]. For RED [4] a mark is to be interpreted by a source as a request to reduce its rate. In contrast, in the optimization based approach of [7], [5] and [11], [9], a mark conveys the necessary information for a source to decide whether to increase or decrease its rate based on its own marginal utility. In [5] the marks per unit flow a source receives is the shadow price at a single bottleneck link, where the shadow price is defined as the marginal increment in expected loss at the link with a marginal increment in load. With REM, the marks allow a source to estimate the sum of shadow prices at the links in its path. Here, however, shadow price is defined to be the marginal increase in aggregate source utility with a marginal increase in link capacity. Simulation evaluation of REM and comparison with RED is reported in [9].

In Section II we review the model and the basic algorithm of [8], [11]. In Section III we present the REM algorithm and the new price computaion scheme. In Section IV we present simulation results to demonstrate the superiority of the new scheme.

II. Model

A. The optimization problem and its dual

Consider a network that consists of a set $L = \{1, \ldots, L\}$ of unidirectional links of capacity c_l, $l \in L$. The network is shared by a set $S = \{1, \ldots, S\}$ of sources. Source s is characterized by four parameters $(L(s), U_s, m_s, M_s)$. The path $L(s) \subseteq L$ for a *point-to-point* connection is a set of links that source s uses, $U_s : \Re_+ \to \Re$ is a utility function, $m_s \geq 0$ and $M_s < \infty$ are the minimum and maximum transmission rates, respectively, required by source s. Source s attains a utility $U_s(x_s)$ when it transmits at rate x_s that satisfies $m_s \leq x_s \leq M_s$. We assume U_s is increasing, strictly concave and twice continuously differentiable in its argument. For each link l let $S(l) = \{s \in S \mid l \in L(s)\}$ be the set of sources that use link l. By definition $l \in L(s)$ if and only if $s \in S(l)$.

Our objective is to choose source rates $x = (x_s, s \in S)$ so as

0-7803-6283-7/00/$10.00 © 2000 IEEE

to:

$$\max_{m_s \le x_s \le M_s} \quad \sum_s U_s(x_s) \tag{1}$$

$$\text{subject to} \quad \sum_{s \in S(l)} x_s \le c_l, \quad l = 1, \ldots, L. \tag{2}$$

The constraint (2) says that the aggregate source rate at any link l does not exceed the capacity. A unique maximizer exists since the objective function is strictly concave, and hence continuous, and the feasible solution set is compact.

Since the source rates x_s are coupled by the constraint (2), solving the primal problem (1–2) directly requires coordination among possibly all sources and is impractical in real networks. A distributed and decentralized solution is provided by its dual [11]:

$$\min_{p \ge 0} \quad D(p) = \sum_s B_s(p^s) + \sum_l p_l c_l$$

where

$$B_s(p^s) = \max_{m_s \le x_s \le M_s} U_s(x_s) - x_s p^s \tag{3}$$

$$p^s = \sum_{l \in L(s)} p_l$$

If we interpret p_l as the price per unit bandwidth at link l then p^s is the total price per unit bandwidth for all links in the path of s. Hence $x_s p^s$ represents the bandwidth cost to source s when it transmits at rate x_s, and $B_s(p^s)$ represents the maximum benefit s can achieve at the given price p^s. A source s can be induced to solve maximization (3) by bandwidth charging. For each p, a unique maximizer of (3), denoted by $x_s(p)$, exists since U_s is strictly concave. Moreover, by duality theory, there exists a dual optimal price $p^* \ge 0$ such that $(x_s(p^{*s}), s \in S)$ that is individually optimal, i.e., solves (3) for each s, is also socially optimal, i.e., solves (1–2).

B. Notations

Unless otherwise specified, z usually denotes a vector whose ith component is some z_i defined before z is introduced. For a scalar z, $z^+ = \max\{z, 0\}$; for a vector z, z^+ is the vector whose ith component is z_i^+. Given a price vector p, $p^s = \sum_{l \in L(s)} p_l$ is the sum of link prices along the path of s, sometimes referred to as a *path price*. Given a rate vector x, $x^l = \sum_{s \in S(l)} x_s$ is the aggregate source rate at link l.

Let $x_s(p)$ be the unique maximizer in (3). We will abuse notation and use $x_s(\cdot)$ both as a function of scalar price $p \in \Re_+$ and of vector price $p \in \Re_+^{|L|}$. When p is a scalar, by the Kuhn–Tucker theorem, $x_s(p)$ is given by

$$x_s(p) = [U_s'^{-1}(p)]_{m_s}^{M_s} \tag{4}$$

where $[z]_a^b = \min\{\max\{z, a\}, b\}$. Here $U_s'^{-1}$ is the inverse of U_s', which exists over the range $[U_s'(M_s), U_s'(m_s)]$ when U_s is continuously differentiable and *strictly* concave. When p is a vector, $x_s(p) = x_s(p^s) = x_s(\sum_{l \in L(s)} p_l)$. The meaning should be clear from the context. Let $x(p) = (x_s(p), s \in S)$.

C. Basic algorithm

In [11], we solve the dual problem using gradient projection method (e.g., [12], [2]) where link prices are adjusted in opposite direction to the gradient $\nabla D(t)$ whose components are $\nabla_l D(t) = c_l - x^l(t)$. We abuse notation and use $x_s(\cdot)$ both as a function of time t and a function of price $p(t)$ given by (4); the meaning should be clear from the context. Then the price computations and rate adjustments are given by:

Basic algorithm:

$$p_l(t+1) = [p_l(t) + \gamma(x^l(t) - c_l)]^+, \quad l \in L \tag{5}$$

$$x_s(t+1) = x_s(p(t)), \quad s \in S \tag{6}$$

where $\gamma > 0$ is a step-size. Hence the prices are adjusted in proportion to excess demand. The algorithm takes the familiar form of reactive flow control where each link computes a price based on local source rates, and each source selects a rate based on the price on its path.

In [11] we prove that the basic algorithm (5–6) converges to the optimal rates provided the utility functions are strictly concave increasing and their second derivatives are bounded away from zero. Specifically if $\{(x(t), p(t))\}$ is a sequence generated by (5–6) then any limit point (x^*, p^*) is primal–dual optimal. Moreover, provided that the sources and links perform their updates frequently enough, convergence is maintained even in an asynchronous environment where sources and links may compute and communicate at different times with different frequencies, and where feedback delays are substantial and time–varying.

III. Random Early Marking

Under the basic algorithm (5–6) a link l needs the aggregate source rate $x^l(t)$ for price computation and a source s needs a scalar feedback of path price $p^s(t)$ for rate adjustment. This communication requirement is not achievable on the current Internet. In this section, we explain how to perform price computation based on offered rate and buffer occupancy, thus eliminating the need for explicit communication from sources to links, and how to feed back the prices to sources using only a single bit. The combination is the REM algorithm.

A. Price computation

Let $x_{ls}(t)$ be the offered rate from source s at link l at time t. Let $\hat{x}^l(t) = \sum_{s \in S(l)} x_{ls}(t)$ be the aggregate offered rate at link l. This rate is generally different from the aggregate source

rate $x^l(t) = \sum_{s \in S(l)} x_s(t)$ used in the basic algorithm, except in equilibrium [10, Lemma 2]. The source rate $x^l(t)$ is not available at link l but the offered rate $\hat{x}^l(t)$ can be measured at link l. We assume each link has a large buffer so that no packets are lost. Let $b_l(t)$ be the buffer backlog at link l at time t, and $b_{ls}(t)$ be the fraction of $b_l(t)$ that is from source s. The aggregate buffer occupancy evolves according to:

$$b_l(t+1) \;=\; [b_l(t) + \hat{x}^l(t) - c_l]^+. \qquad (7)$$

We now present three algorithms for price computation. All three are based on the idea of approximating the gradient $\nabla_l D(t) = c_l - x^l(t)$ in carrying out the gradient projection algorithm (5).

The first algorithm approximates the gradient by estimating the aggregate source rate $x^l(t)$ by the offered rate $\hat{x}^l(t)$ (cf. (5)):

$$\textbf{PC1:} \qquad p_l(t+1) \;=\; [p_l(t) + \gamma(\hat{x}^l(t) - c_l)]^+$$

Multiplying both sides of (7) by the positive step-size γ, we see that the buffer process automatically performs the price computation PC1, *provided that c_l is the true link capacity available to serve the sources in $S(l)$* and that we identify $p_l(t)$ with $\gamma b_l(t)$. Our second algorithm thus simply sets the price to a fraction of the buffer occupancy:

$$\textbf{PC2:} \qquad p_l(t) \;=\; \gamma b_l(t)$$

PC1 and PC2 are proposed, and their convergence to optimality proved, in [10]. Their performance in REM is evaluated and comparison with RED is made in [9]. PC2 is simpler to implement as links do not need to measure the offered rates. It however does not scale: as the number of sources increases, the equilibrium price vector p^*, and hence the equilibrium buffer vector $b^* = \gamma^{-1}p^*$, increases steadily. This not only necessitates large buffer in the network, but worse still, it leads to large feedback delays. Algorithm PC1 can alleviate the problem by setting c_l in PC1 to be a fraction $\rho \in (0,1)$ of the true link capacity. Then in equilibrium, the offered rate $\hat{x}^l(t) = c_l$ is strictly less than the true link capacity and hence backlogs will clear. However, to be effective, ρ needs to be significantly less than 1, leading to low utilization.

These considerations motivate our new algorithm:

$$\textbf{PC3:} \qquad p_l(t+1) \;=\; [p_l(t) + \gamma(b_l(t) + \hat{x}^l(t) - c_l)]^+$$

where c_l can be the true link capacity. In equilibrium, price p^* stabilizes and hence (for a saturated link) $b_l^* + \hat{x}^{*l} = c_l$. If the equilibrium buffer is nonzero $b_l^* > 0$, then the offered rate is strictly less than the capacity $\hat{x}^{*l} < c_l$, and hence the buffer b_l^* could not have been in equilibrium. Hence, by contradiction, we must have both zero buffer $b_l^* = 0$ and full utilization $\hat{x}^{*l} = c_l$ in equilibrium.

B. Price feedback

The basic idea is for a source s to estimate the path price $p^s(t)$ from binary feedback and adjust its rate according to (6) using the estimate $\hat{p}^s(t)$ in place of the true value $p^s(t)$. We now describe the method for price feedback in a simplified synchronous model where time is slotted into update periods. Sources and links update their prices and rates at the beginning of each period. We assume that every source receives a sufficient number of (acknowledgment) packets in each period so that a reasonable estimation can be made.

On packet arrival in period t, if it is not marked, a link l marks it with a probability that is exponential in its current price $p_l(t)$, independent of all other packets:

$$m_l(t) \;=\; 1 - \phi^{-p_l} \qquad (8)$$

where $\phi > 1$ is a constant. Hence the higher the price the more likely packets are marked. The end–to–end marking probability for each packet of source s is then

$$m^s(t) \;=\; 1 - \Pi_{l \in L(s)}(1 - m_l(t)) \;=\; 1 - \phi^{-p^s(t)} \qquad (9)$$

A mark is placed in the ECN bit of a packet en-route to its destination and is carried back to its source in the ECN bit of the packet's acknowledgment, unmodified in the return path.

A source estimates $m^s(t)$, and hence the price $p^s(t)$, by the fraction of marked packets in period t. Suppose source s receives acknowledgment for packets $1, 2, \ldots, N(t)$ in period t. Let $E_k(t)$ be 1 if the kth packet in period t is marked and 0 otherwise, $k = 1, 2, \ldots, N(t)$. Let $\hat{m}^s(t)$ be an estimate of the end–to–end marking probability $m^s(t)$:

$$\hat{m}^s(t) \;=\; \frac{1}{N(t)} \sum_{k=1}^{N(t)} E_k(t)$$

Then from (9) a price estimate is

$$\hat{p}^s(t) \;=\; -\frac{\log(1 - \hat{m}^s(t))}{\log \phi} \qquad (10)$$

C. REM algorithm

Putting both the price computation and price feedback methods together yields the REM algorithm.

Link l's algorithm:

1. On packet arrival, if it is not marked, mark it with probability $m_l(t)$ given by (8), independent of all other packets.

2. At the beginning of each period t, update price using either PC1, PC2, or PC3.

Source s's algorithm:

1. At the end of each period t, estimate path price $\hat{p}^s(t)$ from the fraction $\hat{m}^s(t)$ of its packets marked using (10).

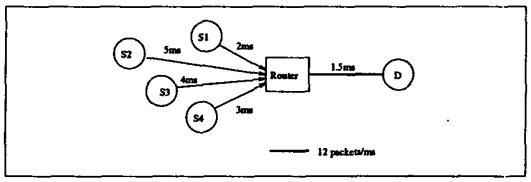
2. Compute a new rate $x_s(t+1) = x_s(\hat{p}^s(t))$ for the next period, where $x_s(\cdot)$ is given by (4).

We now compare the performance of PC1, PC2 and PC3 in the REM algorithm.

IV. Performance

Our simulation model consists of four sources transferring data to a common destination, as shown in Figure 1. Each source is connected to a router via a link of capacity 15 packets/ms and then to the destination via a shared link of capacity 12 packets/ms. The propagation delay of these links are labeled in the figure. The single bottleneck link allows close observation of the behavior of REM in a simple environment. Notice that the bandwidth–delay product is very significant: if each packet is 1,000 bytes then the largest round trip bandwidth–delay product for the shared link is 156kB. This should be compared with the small amount of buffering (averaging around 10kB) under PC1 and PC3; see Figures 2(b) and 4(b).

The starting times of the sessions are staggered by 1 s: The first source, S1, is active from 0s-5s, S2 from 0s-3s, S3 from 1s-4s and S4 from 2s-5s. The utility function of the REM sources was $a_s \log(x_s)$, where a_s was set to C, C is the bottleneck link capacity in packets/s. We used $\phi = 1.2$ in (8).



(a) Single-link topology

Fig. 1. Network Topology

Figure 2 shows the performance of REM under PC1 with γ set to 0.005. The thick lines show the equilibrium (optimal) values. The large oscillations when only a single source is active are due to the high sensitivity of the log utility function at low prices. As more sources activate their windows converge to a neighborhood of the optimal values. The oscillation around the optimal values are mainly due to randomness in the price feedback mechanism. Here c_l was set to 65% of the true link capacity to avoid buildup of the buffer. Hence, as discussed in Section III-A, the link is not fully utilized. For example, until 2s the window size oscillates around 52 packets while a window size of 80 is required for 100% utilization.

Figure 3 shows the results for PC2 with γ set to 0.01. This high value was chosen to keep the equilibrium buffer at a lower level. Large γ leads to large oscillations in price and window size. As buffer builds up round trip delay and its fluctuation increases, preventing the window from converging.

Figure 4 shows the results for PC3 with γ set to 0.005. The price and the congestion window converge to 100% utilization while the buffer remains at a low level.

V. Conclusion

We have proposed a new price computation for REM algorithm proposed in [9] which overcomes the problems of low utilization or large buffer and delay suffered by the previous schemes. Simulation results confirms the superior performance of the new scheme. REM with extensive simulation studies illustrating its stability, robustness and fairness is presented in [1].

References

[1] Sanjeewa Athuraliya, Steven Low, and David Lapsley. Random Early Marking. Submitted for publication, http://www.ee.mu.oz.au/staff/slow/research/, January 2000.

[2] Dimitri P. Bertsekas and John N. Tsitsiklis. *Parallel and distributed computation*. Prentice-Hall, 1989.

[3] S. Floyd. TCP and Explicit Congestion Notification. *ACM Computer Communication Review*, 24(5), October 1994.

[4] S. Floyd and V. Jacobson. Random early detection gateways for congestion avoidance. *IEEE/ACM Trans. on Networking*, 1(4):397–413, August 1993.

[5] R. J. Gibbens and F. P. Kelly. Resource pricing and the evolution of congestion control. *Automatica*, 35, 1999.

[6] F. P. Kelly. Charging and rate control for elastic traffic. *European Transactions on Telecommunications*, 8:33–37, 1997. http://www.statslab.cam.ac.uk/frank/elastic.html.

[7] Frank P. Kelly, Aman Maulloo, and David Tan. Rate control for communication networks: Shadow prices, proportional fairness and stability. *Journal of Operations Research Society*, 49(3):237–252, March 1998.

[8] David E. Lapsley and Steven H. Low. An optimization approach to ABR control. In *Proceedings of the ICC*, June 1998.

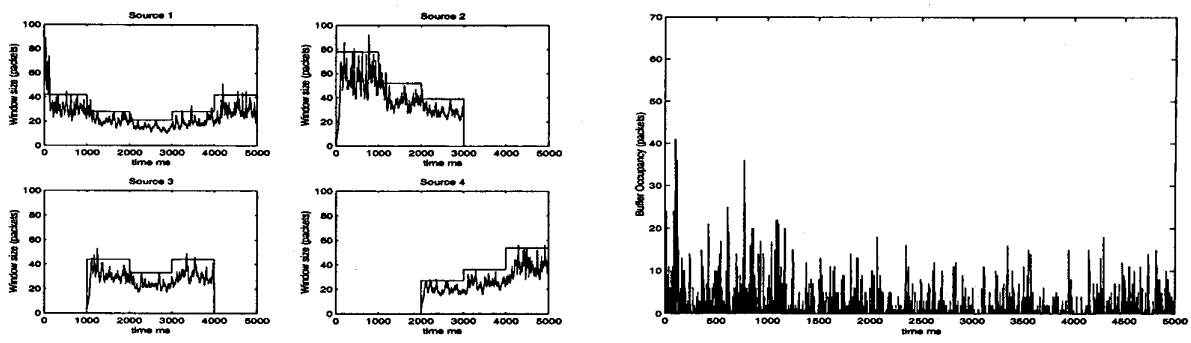
[9] David E. Lapsley and Steven H. Low. Random Early Marking for Internet Congestion Control. In *Proceedings of IEEE Globecom'99*, December 1999.

[10] Steven H. Low. Optimization flow control with on-line measurement. In *Proceedings of the ITC*, volume 16, June 1999.

[11] Steven H. Low and David E. Lapsley. Optimization flow control, I: basic algorithm and convergence. *IEEE/ACM Transactions on Networking*, 7(6):861–874, December 1999. http://www.ee.mu.oz.au/staff/slow/research/.

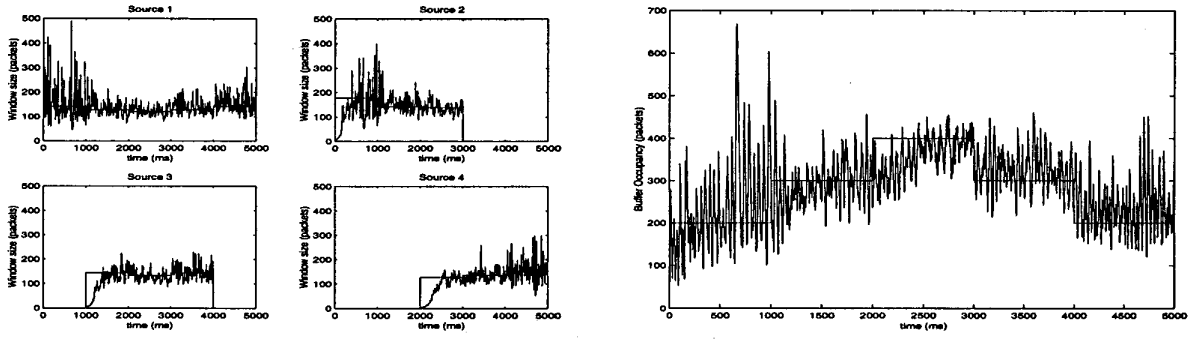
[12] David G. Luenberger. *Linear and Nonlinear Programming, 2nd Ed.* Addison-Wesley Publishing Company, 1984.

[13] K. K. Ramakrishnan and S. Floyd. A Proposal to add Explicit Congestion Notification (ECN) to IP. Internet draft draft-kksjf-ecn-01.txt, July 1998.



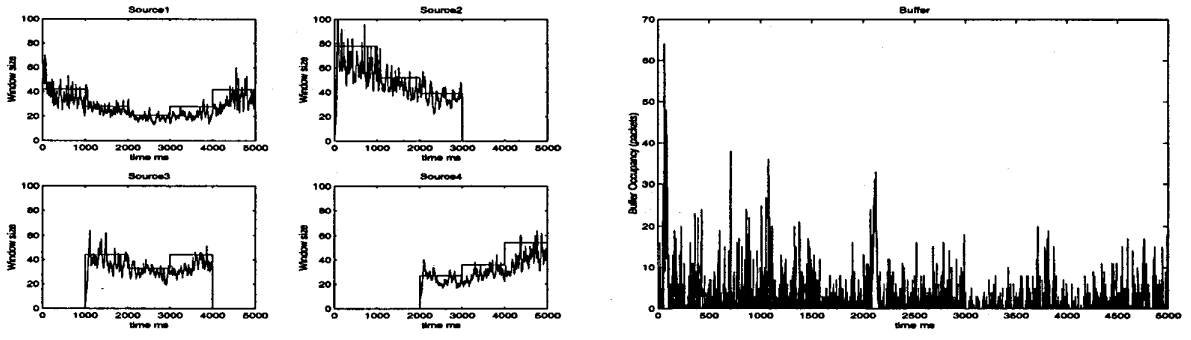
(a) Window. (b) Buffer.

Fig. 2. PC1: Window and Buffer.



(a) Window. (b) Buffer.

Fig. 3. PC2:Window and Buffer.



(a) Window. (b) Buffer.

Fig. 4. PC3: Window and Buffer.