A Statistical Study of Loss-Delay Tradeoff for RED Queues

Homayoun Yousefi'zadeh, *Senior Member, IEEE*, Amir Habibi, Xiaolong Li, *Member, IEEE*, Hamid Jafarkhani, *Fellow, IEEE*, and Claus Bauer

***Abstract*—Aside from the introduction of many new schemes, the use of TCP-based AQM schemes and in specific RED is anticipated to continue in foreseeable future as the de-facto standard of network congestion control. Therefore, conducting extra research work aiming at improving the performance of RED is still a topic of high interest. In this paper, we present an analytical study aiming at the fine tuning of the RED parameters. Utilizing a statistical analysis approach, we formulate an optimization problem aimed at addressing the loss and delay tradeoff of the RED queuing discipline. We provide a two-phase iterative solution to the problem in order to identify the settings of the RED parameters. We discuss the convergence characteristics of our solution and investigate its low complexity characteristics. Through extensive NS2 experiments, we illustrate the advantages of our proposed optimization approach by comparing its results to those of adaptive RED as well as standard RED with recommended parameter settings.**

***Index Terms*—RED, Markov chain queue modeling, optimal parameter fine tuning.**

I. Introduction

In the past years, a number of Active Queue Management (AQM) schemes [1] that utilize random early detection mechanisms have gained widespread acceptance as alternatives of improving loss and congestion characteristics of TCP. Random Early Drop (RED) [2] is arguably the most widely studied random early detection scheme. Although random early detection schemes can potentially outperform traditional drop-tail schemes in presence of TCP flows, it is often difficult to parameterize random early detection queues under different congestion scenarios. In addition, the effectiveness of such schemes in presence of UDP flows and under delay constraints are far less understood. Further, there is a need for constant fine-tuning of parameters to adapt to current network conditions. To that end and based on simplified models, guidelines have been proposed in [3], [4], [5], [6], [7], [8] for setting RED parameters in presence of TCP flows. However, most studies on RED are based on heuristics or simulations rather than a systematic approach.

Paper approved by T. T. Lee, the Editor for Switching Architecture Performance of the IEEE Communications Society. Manuscript received November 24, 2010; revised April 17, August 20, and October 28, 2011, and January 27, 2012.

H. Yousefi'zadeh, A. Habibi, X. Li, and H. Jafarkhani are with the Center for Pervasive Communications and Computing, University of California, Irvine, USA (e-mail: {hyousefi, habibi, xiaolonl, hamidj}@uci.edu).

C. Bauer is with Dolby Laboratories, USA (e-mail: cb@dolby.com).

This work appeared in part in the Proceedings of IEEE ICC 2008.

Digital Object Identifier 10.1109/TCOMM.2012.050812.100733

In what follows, we briefly review a subset of the very large volume of research work most closely related to the subject of this paper. A significant volume of research work [3], [9]–[12] has focused on improving the efficiency and fairness of TCP congestion control algorithms. Recently, the prevalence of high Bandwidth-Delay Product (BDP) networks have introduced significant challenges to the effectiveness of TCP congestion control algorithms. To that end, various mechanisms have been proposed to either adaptively adjust transmitting window sizes [13]–[15], use different congestion signals [16]–[19], or even explicitly signal the congestion information to the sender [20], [21]. Often times, such mechanisms fail to simultaneously achieve high utilization and fairness while maintaining low persistent queue lengths and reduced congestion-induced packet drop rates. In contrast, eXplicit Congestion-control Protocol (XCP) [22], Variable-structure Congestion-control Protocol (VCP) [23], and Multi Packet Congestion-control Protocol (MPCP) [24] decouple fairness control from efficiency. By encapsulating congestion related information into packet headers, all three protocols attempt at achieving high utilization, low persistent queue length, insignificant packet loss rate, and sound fairness. However, XCP requires multiple bits in the IP packet header introducing significant deployment obstacles. To the contrary, VCP and MPCP are able to achieve a performance comparable to XCP using only two ECN bits while keeping compatibility with a variety of existing protocols.

From the standpoint of barrier to entry for existing standards and network infrastructure, schemes such as RED, REM [12], VCP, MPCP, and other mechanisms only utilizing one or two bits of ECN in the IP packet header stack up more favorably than those using a larger number of bits in the IP packet header. The latter is also of special importance since the manipulation of multiple bits in the IP packet header is subject to major practicality implications in encrypted networks. For example, the IPSec protocol only allows for bypassing of six Differentiated Services (DiffServ) bits and two ECN bits in the IP header across encryption boundaries.

Aside from the introduction of many new schemes, the use of TCP-based AQM schemes such as RED is anticipated to continue in foreseeable future as the de-facto standard of network congestion control. Thus, conducting extra research work that aims at improving the performance of RED is still a topic of interest.

In this paper, we perform a systematic study on the optimal fine tuning of the RED parameters. Given the statistical properties of the arrival pattern of a RED queue, we identify the RED parameters yielding the minimal loss characteristic under a delay and a normalizing fixed size packet constraint. Our problem formulation appears in the form of a constrained optimization problem that is efficiently solved in two iterative phases. Compared to its conference version, this paper introduces an intelligent heuristic search algorithm with a linear complexity as oppose to a standard search algorithm with quadratic complexity, refines the optimization approach based on the literature of Block Coordinate Descent (BCD), and greatly extends the performance evaluation results.

The rest of this paper is structured as follows. In Section II, we provide a brief review of the RED algorithm along with a profiling of the traffic patterns feeding RED queues. In Section III, we formulate and solve an optimization problem targeted at minimizing the loss characteristic of a RED queue while satisfying an acceptable delay profile. Section IV provides numerical results associated with the proposed algorithms of Section III. Conclusions are provided in Section V and Appendix I formalizes the general case of our optimization algorithm.

II. RED Preliminaries

In this section, we briefly describe the RED algorithm. We also provide a traffic profiling discussion focusing on the steady-state distribution probability of the occupancy of a RED queue.

A. The RED Algorithm

The average queue size of a RED queue is calculated using a low-pass filter with an exponentially weighted moving average as

$$q_t = (1 - w_q)\, q_{t-1} + w_q\, \tilde{q}_t \tag{1}$$

where q_t is the current average queue size, q_{t-1} is the average queue size at the last time instant, w_q is the weighting function, and $\tilde{q}_t$ is the current instantaneous queue size. The value of q_t is then compared to two thresholds, a minimum threshold q_{min} and a maximum threshold q_{max}. Each arriving packet is dropped with probability p_t given by

$$p_t = \begin{cases} 0, & q_t < q_{min} \\ \epsilon_t = \frac{q_t - q_{min}}{q_{max} - q_{min}}\, p_{max}, & q_{min} \leq q_t < q_{max} \\ 1, & q_t \geq q_{max} \end{cases} \tag{2}$$

While in our study p_t is varied linearly from 0 to p_{max} when the RED queue occupancy is in the region between q_{min} and q_{max}, there are many other possibilities of choosing this drop probability. Examples include choosing p as a nonlinear convex, or nonlinear concave function of the queue size.

B. Traffic Profiling

We consider a queuing system with a capacity of K fixed size packets operating under the RED queuing discipline. The RED queuing system is described by its traffic pattern and is assumed to be operating in its steady-state regime. In [25] and [26], the authors develop a model for analyzing both transient and steady-state behavior of RED queues accommodating a large number of random traffic sources. The traffic generation pattern of each source is assumed to follow a Poisson process with a time varying rate. As the result of enforcing RED with a value of w_q in the order of 10^{-3} and for a slowly varying Poisson parameter, the RED queue is considered to be operating in a quasi-stationary state. As such, the behavior of the queue can be approximated with M/G/1/K queuing discipline.

In our study, we consider both UDP and TCP traffic scenarios. When dealing with UDP traffic, we utilize the M/D/1/K queuing discipline as the best practical alternative of today's Internet. Fig. 1 shows such queuing system representing the Markov chain embedded in an M/D/1/K process at departure instants, with the corresponding RED loss probabilities. RED's dropping behavior is then mapped to three subsets of the states before q_{min}, between q_{min} and q_{max}, and after q_{max}. For an M/D/1/K queue with a load factor ρ, we normalize the service time to indicate the time unit such that the arrival rate is equal to ρ. Then, the steady-state probabilities π_k of being in state k for $k \in \{0, \cdots, K\}$ form a discrete Probability Mass Function (PMF) the terms of which are calculated as

$$\pi_k = \begin{cases} \frac{\pi_k^\infty}{\pi_0^\infty + \rho G(K)}, & k \in \{0, \cdots, K-1\} \\ 1 - \frac{G(K)}{\pi_0^\infty + \rho G(K)}, & k = K \end{cases} \tag{3}$$

where $G(K) = \sum_{k=0}^{K-1} \pi_k^\infty$. Further, the steady-state probability π_k^∞ of state k for an infinite capacity M/D/1 queuing system with load ρ is identified in Page 44 of [27] as

$$\begin{aligned} \pi_k^\infty \; &= (1-\rho)\,\Big[\sum_{i=1}^{k} e^{\rho i}(-1)^{k-i}\frac{(i\rho)^{k-i}}{(k-i)!} \\ &\quad + \sum_{i=1}^{k-1} e^{\rho i}(-1)^{k-i}\frac{(i\rho)^{k-i-1}}{(k-i-1)!}\Big], \quad k \geq 2 \end{aligned} \tag{4}$$

where $\pi_0^\infty = 1 - \rho$ and $\pi_1^\infty = (1-\rho)(e^\rho - 1)$. We note that depending on the choice of ρ, the numerical evaluation of (4) faces stability issues when k is larger than 12. In such cases, the asymptotic approximation of Equation (15.1.4) of [28] can be used to identify the steady-state probabilities as

$$\pi_k^\infty \;\approx\; C_0 \left[e^{r_0(k-1)} - e^{r_0 k}\right] \tag{5}$$

where r_0 is the unique negative root of the equation $r = \rho(1 - e^{-r})$ and $C_0 = \frac{1-\rho}{\rho e^{-r_0} - 1}$.

We note that the case of TCP flows is best represented by G/G/1/K queuing model for which we do not provide a queuing analysis in this paper.

III. Optimal Fine Tuning of the RED Parameters

Given the statistical characteristics of a traffic pattern, the discussion of this section revolves around fine tuning of the RED parameters namely q_{min}, q_{max}, p_{max}, and w_q. We work with fixed size packets and assume a deterministic normalized service time of one packet per unit time.

A. The Case of Instantaneous Queue Size

In this subsection, we establish a foundation for our optimization problem by focusing on the case of instantaneous queue size, i.e., $w_q = 1$. Given the steady-state probabilities

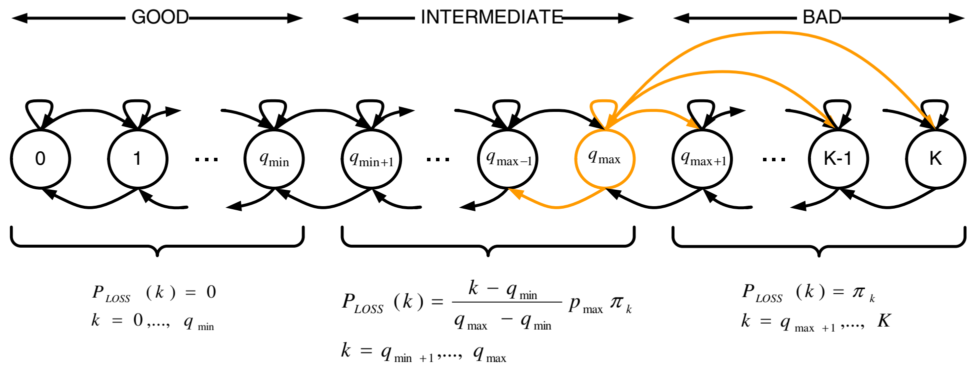


Fig. 1. M/D/1/K approximation of the steady-state behavior of RED under quasi-stationary assumptions. For clarity of the figure, only the full set of transitions associated with $k = q_{max}$ are shown.

π_k of being in state k where $k \in \{1, \cdots, K\}$, we define $P_{LOSS}(k)$, the conditional probability of loss in state k, as

$$P_{LOSS}(k) = \begin{cases} 0, & k < q_{min} \\ \frac{k-q_{min}}{q_{max}-q_{min}} p_{max}\pi_k, & q_{min} \le k \le q_{max} \\ \pi_k, & k > q_{max} \end{cases} \tag{6}$$

Hence, the statistical probability of loss for an arriving packet at a RED queue is expressed as

$$P_{LOSS} = \sum_{k=1}^{K} P_{LOSS}(k) = \sum_{k=q_{max}+1}^{K} \pi_k + \sum_{k=q_{min}}^{q_{max}} \frac{k-q_{min}}{q_{max}-q_{min}} p_{max}\pi_k \tag{7}$$

Note that Equation (7) represents a statistical average in which the probability of loss in each state is calculated based on the queue occupancy in comparison with the RED thresholds. In the presence of a FIFO service discipline, the statistical queuing delay of a packet arriving at a RED queue is calculated as

$$P_{DELAY} = \sum_{k=0}^{q_{min}} (k+1)\,\pi_k + \sum_{k=q_{min}+1}^{q_{max}} (1 - \frac{k-q_{min}}{q_{max}-q_{min}} p_{max})\ (k+1)\,\pi_k \tag{8}$$

We note that Equation (8) represents a statistical average in which the delay in each state is calculated based on the queue occupancy and the probability of drop in comparison with the RED thresholds. While the first summation term captures the contribution of the states below the minimum threshold of queue occupancy, the second summation term captures the contribution of the states between the two thresholds. Importantly, the statistical delay is only calculated for the packets that are accommodated but not those that are dropped. Utilizing Equation (7) and (8), we can now formulate a constrained optimization problem that attempts at minimizing the probability of packet loss subject to an upper bound D_{max} on its statistical queuing delay. The optimization problem is formulated as

$$\min_{q_{min}, q_{max}, p_{max}} \quad P_{LOSS} \tag{9}$$

$$\text{Subject To:} \quad P_{DELAY} \le D_{max} \tag{10}$$

$$0 \le q_{min} < q_{max} \le K \tag{11}$$

$$0 \le p_{max} \le 1 \tag{12}$$

In order to efficiently solve the problem, we describe a two-phase iterative solution to the problem formulated above and show that our solution converges to a local minimum. In the first phase, we analytically solve for the optimal value of p^*_{max} assuming q_{min} and q_{max} are fixed and given.

Phase 1: Given fixed thresholds q_{min} and q_{max}, the only decision variable in solving the optimization problem is p_{max}. While the cost function is minimized for the smallest value of p_{max}, the constraint function (10) enforces a lower bound on the value of p_{max}. The optimal value of p_{max} is then calculated at the boundary point of the constraint function (10) as

$$D_{max} = \sum_{k=0}^{q_{min}} \pi_k(k+1) + \sum_{k=q_{min}+1}^{q_{max}} \pi_k(k+1)(1 - \frac{k-q_{min}}{q_{max}-q_{min}} p^*_{max}) \tag{13}$$

The solution to the equation above appears as

$$p^*_{max} = \left[\frac{\sum_{k=0}^{q_{max}} \pi_k(k+1) - D_{max}}{\sum_{k=q_{min}+1}^{q_{max}} \pi_k(k+1)(k-q_{min})} \right] (q_{max} - q_{min}) \tag{14}$$

Note that the operation associated with deriving the value of p^*_{max} from Equation (14) has a time complexity in the order of $\mathcal{O}(K)$. Further, the value of p^*_{max} satisfies the constraint function (12).

In the second phase, we provide a reduced order search strategy in order to identify the values of q^*_{min} and q^*_{max} based on a fixed value of p_{max} given in the first phase.

Phase 2: Given a fixed value for p_{max} obtained by the solution of the first phase, the decision variables in solving the optimization problem are q_{min} and q_{max}. Noting that q_{min} and q_{max} are integers satisfying $q_{min} < q_{max}$ in the case of instantaneous queue size, we can perform a search in the 2D space of (q_{min}, q_{max}). We note that the upper triangle identified by vertices A, B, and C in Fig. 2 represents the feasible region of Constraint (11). Because the feasible region of Constraint (11) contains $\frac{K(K+1)}{2}$ points, the time complexity of performing the search is in the order of $\mathcal{O}(K^2)$. That said, we have experimentally observed an interesting phenomenon allowing us to develop a heuristic search algorithm with a linear time complexity. While we have no formal proof for our heuristic algorithm, we have consistently verified its accuracy in our large set of experimental findings. According to our observation, the

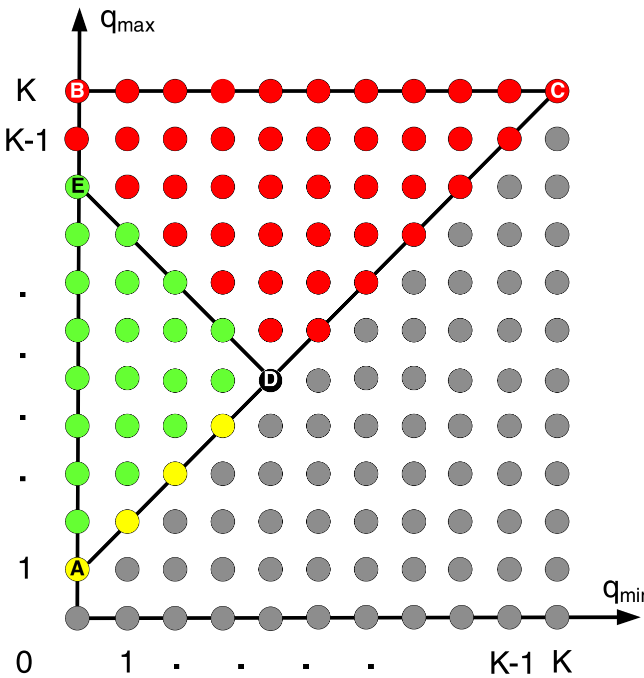


Fig. 2. An illustration of the optimization region of (q_{min}, q_{max}) in their 2D space. The green and yellow region identify a subset of the feasible region of Constraint (11) satisfying Constraint (10).

feasible region of Constraint (11) is always partitioned into a set of qualified points and a set of disqualified points. The partitioning of the two sets is the result of verifying which points satisfy Constraint (10) of the optimization problem. In Fig. 2, the set of green interior points or yellow boundary points of the triangle identified by vertices A, D, and E indicate qualified points, while the set of red points covering the rest of the feasible region of Constraint (11) indicate disqualified points. The single black point D is always located on the main diagonal line segment attaching point A to point C. It is also located at the boundary of the set of qualified and disqualified points. The set of red disqualified points is always bounded by segments DE, EB, BC, and CD. The set of green/yellow qualified points is also always bounded by segments AE, ED, and DA. In different experimental scenarios, we have observed that point E moves on segments AB and BC. Further, segment DE sometimes appears in the form of a curve as opposed to a straight line in some scenarios. Regardless of the scenario and the shape of DE, we have always observed that the black point D located at the intersection of segments DE and AC is the optimal point identified by the search.

As such, we can identify a revised search criterion with a linear time complexity of $\mathcal{O}(K)$. Such criterion traverses the points located on segment AC above the main diagonal starting from point A and moving up toward point C. The optimal point D is then identified as the one on segment AC above the main diagonal with the smallest value of the cost function (9). In fact, the pattern of the values of the cost function starting from point A is always non-decreasing until reaching the point after point D on segment AC. Thus, the search can be stopped at that point.

Below, we introduce an algorithmic representation of our two-phase iterative solution described above that can identify an optimal solution to the constraint NonLinear Integer Programming (NLIP) problem of (9), (10), (11), and (12).

Algorithm 1 Iterative Optimization Algorithm

1: **/* Step 1: Initialization */**
2: Initialize ($q_{min} \leftarrow \lfloor K/3 \rfloor$), ($q_{max} \leftarrow \lfloor 2K/3 \rfloor$), grid size width ($\varepsilon \leftarrow 1$), initial iteration number ($i \leftarrow 0$), maximum iteration number ($i_{max} \leftarrow 10^3$), intermediate variables ($L_3 \leftarrow 0$), ($L_4 \leftarrow 0$), and stoppage criterion variables ($L_1 \leftarrow 0$), ($L_2 \leftarrow 1$), ($\delta \leftarrow 10^{-6}$).
3: **repeat**
4:   **/* Step 2: Calculate p^*_{max} */**
5:   Calculate the optimal value of p^*_{max} from Equation (14)
6:   **/* Step 3: Calculate q^*_{min} and q^*_{max} */**
7:   Reset ($L_3 \leftarrow 1$).
8:   **for** ($q_{max} = 1 \quad to \quad K$) { **do**
9:     $q_{min} \leftarrow (q_{max} - \varepsilon)$
10:     **if** Constraint function (10) is satisfied **then**
11:       Store the value of the cost function (9) in L_4
12:     **end if**
13:     **if** $L_3 > L_4$ **then**
14:       ($q^*_{min} \leftarrow q_{min}$), ($q^*_{max} \leftarrow q_{max}$), and ($L_3 \leftarrow L_4$);
15:     **else**
16:       Break
17:     **end if**
18:   **end for**
19:   **/* Step 4: Update Stoppage Criterion Variables */**
20:   Set ($L_1 \leftarrow L_2$) and ($L_2 \leftarrow L_3$)
21:   Set ($i \leftarrow i + 1$)
22:   **/* Step 5: Check Stoppage Criterion */**
23: **until** $\{(\frac{|L_1 - L_2|}{L_1} < \delta)$ or $(i > i_{max})\}$

In the algorithm above, Step 1 in lines 1-2 initializes the variables used by the algorithm. The main iterative loop is shown in the **repeat-until** block spanning over lines 3-23. Within the loop, Step 2 for calculating p^*_{max} is described by lines 4-5, Step 3 for intelligently searching to identify q^*_{min} and q^*_{max} is described by lines 6-18, Step 4 for updating the stoppage criterion is identified by lines 19-21, and Step 5 for meeting the stoppage criterion and subsequently exiting the loop is covered by lines 22-23.

We note that the time complexity of implementing the above algorithm is $\mathcal{O}(IK^2)$ where I indicates the number of iterations. Further, we note that the two-phase iterative optimization algorithm of this section expressed for the case of instantaneous queue size with decision variables $p_{max}, q_{min}, q_{max}$ is a special case of the two-phase iterative optimization algorithm of Appendix I expressed for the case of average queue size with decision variables $p_{max}, q_{min}, q_{max}, w_q$. Based on the discussion of Appendix I, both of these two-phase iterative optimization algorithms converge to fixed points which are conjectured to be local minima.

B. The Case of Average Queue Size

In this subsection, we generalize the formulation of the previous section to the case of average queue size.

We open our discussion by indicating that our objective is to first express the current average queue size q_t in terms of the current instantaneous queue size $\tilde{q}_t$. The latter is equivalent to providing the solution to the first-order difference equation expressed by (1) with input $\tilde{q}_t$ and output q_t. Relying on the method of successive calculations and starting from the initial condition q_0, the following pattern is observed.

$$q_t = (1-w_q)^t q_0 + \sum_{j=1}^{t} w_q (1-w_q)^{t-j} \tilde{q}_j \qquad (15)$$

Since a unique solution exists, it is sufficient to verify that Equation (15) satisfies the original equation. Relying on induction, we start from Equation (1) to note that

$$\begin{aligned} q_t &= (1-w_q)q_{t-1} + w_q \tilde{q}_t \qquad (16) \\ &= (1-w_q)((1-w_q)^{t-1} q_0 \\ &\quad + \sum_{j=1}^{t-1} w_q (1-w_q)^{t-j-1} \tilde{q}_j) + w_q \tilde{q}_t \\ &= (1-w_q)^t q_0 + \sum_{j=1}^{t-1} w_q (1-w_q)^{t-j} \tilde{q}_j + w_q \tilde{q}_t \\ &= (1-w_q)^t q_0 + \sum_{j=1}^{t} w_q (1-w_q)^{t-j} \tilde{q}_j \end{aligned}$$

arriving at the right hand side of Equation (15).

Analyzing Equation (15), we notice that it consists of a transient and a steady-state term. Since $0 \leq 1 - w_q \leq 1$, the transient term $(1-w_q)^t q_0$ goes to zero in steady-state. The steady-state solution is thus expressed as

$$q_t = \sum_{j=1}^{t} w_q (1-w_q)^{t-j} \tilde{q}_j \qquad (17)$$

Our numerical evaluations have supported the observation that the set of discrete random variables $\{\tilde{q}_j\}_{j=1}^{t}$ are Independently and Identically Distributed (IID) in the steady-state[1]. Recall that the distribution of discrete random variables $\{\tilde{q}_j\}_{j=1}^{t}$ can be determined from the traffic and queuing profile. Relying on the IID assumption, the steady-state PMF of the random variable q_t appearing in the form of a weighted sum of t random variables $\{\tilde{q}_j\}_{i=j}^{t}$ can be numerically calculated as a scaled discrete convolution of a number of PMFs, [29]. Further, the PMF of q_t only depends on a small number of random variables q_t, q_{t-1}, q_{t-2}, and so on because the scaling factor $1 - w_q$ is smaller than one.

Once the PMF of q_t is calculated, we can revert back to the constrained optimization problem with the cost function (9) and constraint set (10), (11), and (12). In the latter case, a new constraint related to the variable w_q is added to the constraint set as shown below.

$$0 \leq w_q \leq 1 \qquad (18)$$

[1]Note that the IID assumption is only utilized to reduce the complexity of numerically calculating the PMF of q_t. The PMF of q_t can be numerically calculated even in the absence of the IID property albeit with a higher complexity.

For the case of average queue size, it is important to note that the steady-state probabilities appearing in the cost and constraint functions of the optimization problem depend on the RED parameter w_q. Thus, w_q represents a new decision variable for the optimization problem. While one can still utilize the two-phase recursive optimization approach to solve the resulting problem, the closed-form expression identified for p^*_{max} in the first phase does not hold any longer. Rather, a numerical optimization approach such as Sequential Quadratic Programming (SQP) [30] should be used in conjunction with a line search algorithm such as the one proposed by [31] to calculate the values p^*_{max} and w^*_q in the first phase.

Next, we investigate potential implications of utilizing average queue lengths rather than instantaneous queue lengths on the second phase of our proposed algorithm. In the latter scenario, we note that the search of the second phase has to be performed over a continuous $K \times K$ space to identify q^*_{min} and q^*_{max}. In order to perform the search over the region of the $K \times K$ space, a quantized grid covering the triangle with edges at the origin, point B, and point C of Fig. 2 is formed. Therefore, the complexity of the search is higher depending on the granularity of the quantization grid. Nonetheless, our observation of the case of instantaneous queue size with regards to the boundary of the feasible region of Constraint (11) still holds and our intelligent search algorithm described earlier can still be applied. The time complexity of implementing the above algorithm is $\mathcal{O}(I \max(GK^2, N \log N))$ where I, G, and N indicate the number of iterations, the grid size, and the degree of quadratic estimation identified by SQP method, respectively. With a large grid size and when w_q is fixed, e.g., $w_q = 0.002$, the complexity of the problem is reduced to that of instantaneous queue size.

In Appendix I, we provide a two-phase iterative optimization algorithm with decision variables $p_{max}, w_q, q_{min}, q_{max}$ for the case of average queue size.

At the end of this subsection, we note that the dual problem of our optimization problem for both cases of instantaneous and average queue size is obtained by swapping the cost function (9) with the constraint function (10). The dual problem can then be solved relying on a similar approach described in this section.

C. RED Design Guidelines

In this subsection, we provide some RED design insights based on the analysis of this section. First, we note that the derivation of our results is independent of the traffic model of Section II-B. In fact, the results are valid as long as the steady-state probabilities of being in each state can be identified either analytically or numerically, i.e., the details of the queuing model of Fig. 1 such as transition probabilities are not of significant importance. Second, the approach of this section can be utilized either on an online (closed-loop) or offline (open-loop) basis depending on the type of background traffic. While in the case of reactive TCP flows a closed-loop approach is to be used in which steady-state probabilities are treated as moving time averages, in the case of UDP flows an open-loop approach utilizing the existing queuing models can work better. Third, the validation results of our approach are

TABLE I
FIXED PARAMETER SETTINGS OF SRED AND ARED. FOR SRED, $p_{max} = 0.1$.

K	q_{min}	q_{max}	w_q
20	6.66	13.33	3.99e-5
30	10.00	20.00	3.99e-5
50	16.66	33.33	3.99e-5
100	62.50	133.33	3.99e-5
500	62.50	187.50	3.99e-5

pointing to the fact that under the optimal loss-delay tradeoff operating regime, specially as delay bounds become tighter, the optimal settings of a RED queue force its behavior to match that of a drop-tail queue. While at the first glance this result may not be expected, it is an important finding of this paper.

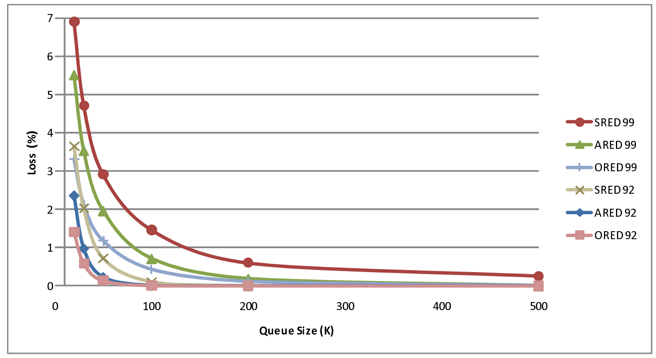
IV. NUMERICAL ANALYSIS

In this section, we validate the results of the algorithms of Section III. In order to manage the available space, we only report sample results selected from a large set of several tens of thousand experiments conducted by us using NS2 [32] discrete event simulation tool. Nonetheless, we note that the reported results are true indicators of the categorical behavior of our experiments.

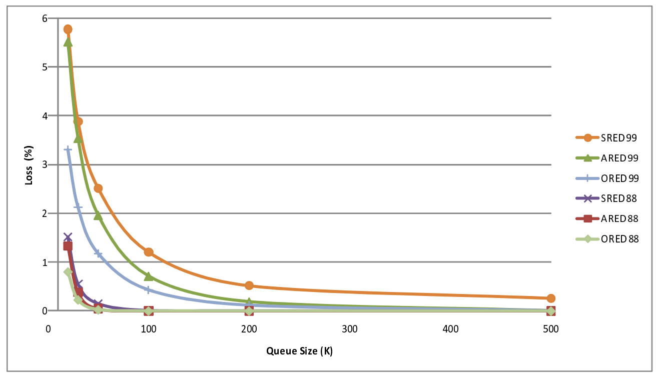
The topology of our experiments includes a single server queue the behavior of which is governed by RED. We experiment with fixed length data packets of size 1024 bytes and in the case of utilizing TCP flows ACKnowledgment (ACK) packets of size 40 bytes. In our experiments, all of the values of q_{min}, q_{max}, and K are *normalized* and expressed as multiples of a size of a data packet. The RED queue is fed with UDP Poisson arrival patterns, FTP arrival patterns utilizing TCP SACK1, and HTTP arrival patterns utilizing TCP SACK1. TCP traffic arrival patterns are generated directly by NS2 and are independent of Poisson patterns. TCP SACK1 is implemented based on TCPAgent which in turn uses TCP Tahoe with selective repeat ACKs following RFC2018. The queue is assumed to offer a *normalized* service rate of one data packet per second. The settings allow us to examine the performance of our algorithm for UDP traffic patterns mapped to M/D/1/K queuing model as well as TCP traffic patterns.

Viewing the queue capacity K and the maximum delay threshold D_{max} as our design parameters, our experiments span over two sets. In the first set of experiments, we investigate the loss performance of our proposed solution for a fixed D_{max} and varying queue sizes K. In the second set of experiments, we investigate the loss performance of our proposed solution for a fixed K and varying delay thresholds. We compare the performance of our solution with that of Standard RED (SRED) and Adaptive RED (ARED) [8]. The parameters of SRED and ARED (with the exception of p_{max} for ARED) are selected by NS2 representing the best heuristic and numerical findings in the literature. While the value of p_{max} is set to 0.1 in SRED experiments, it is chosen dynamically in ARED experiments in order to maintain a stable average queue length. Table I includes the values of the fixed parameters used by both SRED and ARED.

In the discussion and figures below, our Optimal RED algorithm is referred to as ORED. Fig. 3 illustrates the comparison results of ORED with those of SRED and ARED for the cases of instantaneous and average queue sizes utilizing two different choices of load factor ρ associated with a Poisson arrival pattern. As observed from the figure, the loss performance of ORED is best among the three for both cases of instantaneous and average queue size and both choices of ρ.



(a) Poisson Instantaneous Queue Size

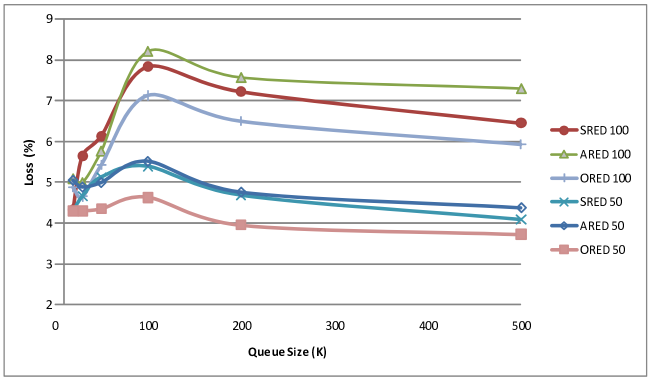


(b) Poisson Average Queue Size

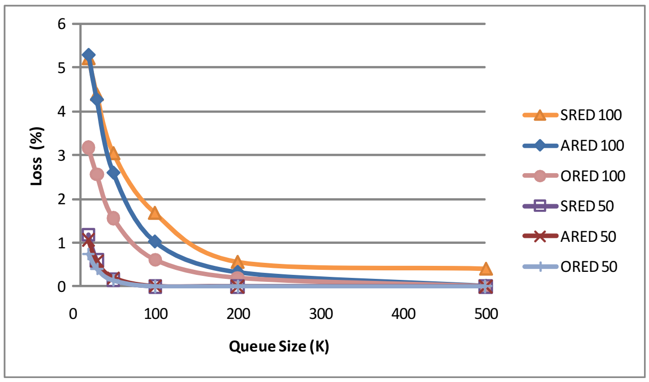
Fig. 3. A performance comparison of ORED, ARED, and SRED for a fixed normalized service rate of one packet per second, $D_{max} = 100$msec, and a Poisson arrival pattern with two different choices of load factors ρ. The cases of (a) instantaneous queue size with $w_q = 1$, $\rho \in \{92\%, 99\%\}$, and (b) average queue size with $\rho \in \{88\%, 99\%\}$ are considered.

Next, we investigate the performance of our scheme for TCP traffic. In TCP experiments, we numerically generate time-varying moving average values of π_k with $k \in \{0, \cdots, K\}$. For every time epoch consisting of the last N instances of discrete arrival time, we count the number of occurrences of each k. Then, we run our iterative algorithm for that time epoch with an initial condition set by the solution of the previous step.

Fig. 4 illustrates the results of feeding the queue with FTP and HTTP traffic patterns utilizing TCP SACK1. The two triplets of curves in Fig. 4(a) and Fig. 4(b) are associated with an aggregate traffic pattern generated by 50 and 100 sources. As observed from the figures, ORED is again outperforming both SRED and ARED. It is also interesting to observe that the effects of changing the number of sources for both experiments is only relevant for small values of K. As K increases from 20 to 100 in Fig. 4(a), all curves show an increased loss followed by a decreased loss for $K > 100$. The reason is that congestion-caused loss (but not RED-caused loss) is dominant when the queue capacity is less than 100. As the



(a) FTP Average Queue Size

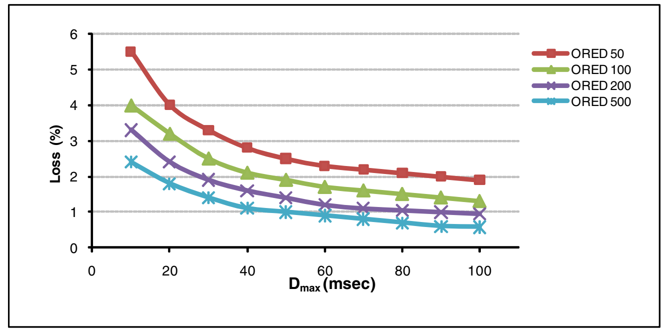


(b) HTTP Average Queue Size

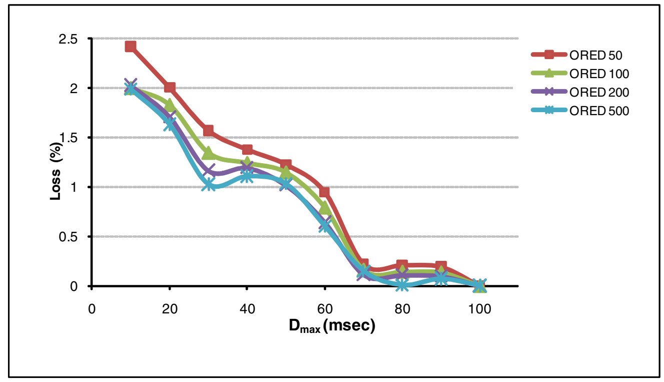
Fig. 4. A performance comparison of ORED, ARED, and SRED for aggregate traffic patterns generated by (a) $\{50, 100\}$ FTP, and (b) $\{50, 100\}$ HTTP sources utilizing TCP SACK1. A fixed normalized service rate of one packet per second, D_{max} = 100msec, and the case of average queue size are considered.

queue capacity increases from 20 to 100, the overall arrival rate of the queue increases proportional to the queue capacity yielding a higher overall loss rate. As the queue capacity increases beyond 100, RED-caused loss becomes dominant and the overall loss rate drops. The observation of Fig. 4(a) is absent in Fig. 4(b) because HTTP flows are short-lived. Another interesting observation in the figures is that ARED outperforms SRED for small values of K but not large values of K. This is because ARED is designed to maintain an average queue length around the value of $\frac{1}{2}(q_{min} + q_{max})$. To that end, ARED drops packets earlier and in larger numbers than SRED in order to be able to accommodate bursty traffic. Using the automatic configuration option available in the current implementation of NS2 for SRED, the parameters are configured according to the link bandwidth and link queue capacity leading to a better handling of the loss-delay tradeoff at the price of forming longer queues.

In the second set of experiments, the queue size is fixed. Utilizing a Poisson arrival with a load factor of $\rho = 0.99$, Fig. 5(a) illustrates the loss performance of ORED as a function of delay threshold for four different values of K. Fig. 5(b) shows similar results for a traffic pattern generated by 10 FTP sources. The curves in both figures show that increasing the value of D_{max} will result in reducing loss by servicing a larger number of packets that stay in the queue for a longer period of time.



(a) Poisson Instantaneous Queue Size



(b) FTP Instantaneous Queue Size

Fig. 5. A performance comparison of ORED as a function of D_{max} for four different choices of queue size $K \in \{50, 100, 200, 500\}$ with (a) a Poisson arrival pattern identified by $\rho = 0.99$, and (b) an aggregate traffic pattern of 10 FTP sources utilizing TCP SACK1. The case of instantaneous queue size with $w_q = 1$ is considered.

The results of SRED for both Fig. 5(a) and Fig. 5(b) as well as the results of ARED for Fig. 5(b) are not shown for clarity but are similar to reported results. While not shown, each curve of ORED is always below the curves of ARED and SRED for each choice of K. Further, our experiments in the case of average queue size and in a broad range of parameter selections as well as with implementations of TCP besides TCP SACK1 have led to observing results consistent with those reported here.

Considering the wide range of parameter settings in our experiments, our technique is able to perform well under different traffic loads and types. Further, experimenting with large values of ρ in the case of Poisson traffic patterns as well as a large number of sources in the case of TCP traffic patterns illustrate the robustness of our technique under severe congestion conditions.

We have observed that our iterative algorithm converges after an average value of $I = 10$ iterations for Poisson experiments. It converges after an average value of $I = 15$ and $I \leq 7$ iterations for the initial and subsequent epochs in TCP experiments, respectively. Thus, we also argue that applying our technique to identify optimal settings of RED parameters is relatively practical.

V. Conclusion

In this paper, we presented the results of our systematic study on the optimal fine tuning of RED parameters. Given the steady-state of queue occupancy values, we identified the

RED parameters locally minimizing the loss characteristic of the queue while satisfying an acceptable delay profile. Our problem formulation appeared in the form of a constrained optimization problem that could be efficiently solved in two iterative phases. Through extensive simulations the sample results of which reported here in comparison with standard RED and adaptive RED schemes, we investigated the accuracy and efficiency characteristics of our algorithms.

APPENDIX A
ITERATIVE OPTIMIZATION OF RED FOR THE CASE OF AVERAGE QUEUE SIZE

In this section, we claim that a generalization of the iterative optimization algorithm of Section III-A converges to a fixed point solution of the problem formulated by the objective function (9) and constraints (10), (11), (12), (18) along with decision variables $p_{max}, w_q, q_{min}, q_{max}$ for the case of average queue size. Further, we conjecture that the fixed point is a local optimal point.

Choosing a value of 1 for parameter ε^2, the generalized version of the iterative optimization algorithm of Section III-A is derived by changing Step 2 to calculate the optimal value of p^*_{max} and w^*_q utilizing SQP and the line search algorithm of [31] instead of Equation (14).

To support our claim, we note that both of our two-phase iterative algorithms find their roots in the literature of BCD [42]–[44]. The BCD algorithms are effective methods of minimizing real-valued continuously differentiable functions of partitioned multiple decision variables subject to bound constraints. They can also be used when additional non-bound constraints exists. A BCD algorithm is known to converge to a stationary point if the objective function (or the Lagrangian function formed by the objective and the nonlinear constraint functions) is convex or under milder conditions quasiconvex and hemivariate.

While we cannot mathematically prove our iterative algorithms converge to local minima, we have consistently observed the following phenomena through extensive simulation results. First, both two-phase iterative algorithms always converge to the vicinity of the optimal solution identified by exhaustive search either in the unpartitioned 4-dimensional space of decision variables in the case of average queue size or the 3-dimensional space of decision variables in the case of instantaneous queue size. Second, the plots of objective function and delay constraint against partitioned decision variables always appear to satisfy convexity or semi-convexity and hemivariation conditions.

Based on the discussion above and since the the cost function of Equation (9) can only decrease in each iteration of our two-phase algorithm, we conjecture that the algorithm converges to a local minimum.

REFERENCES

[1] R. Braden, D. Clark, J. Crowcroft, B. Davie, S. Deering, D. Estrin, S. Floyd, V. Jacobson, G. Minshall, C. Petridge, L. Peterson, K. Ramakrishnan, S. Shankar, J. Wroclawski, and L. Zhang, "Recommendations on queue management and congestion avoidance in the Internet," Internet Draft, Mar. 1997.
[2] V. Jacobson, "Presentations to the IETF performance and congestion control working group," Aug. 1989.
[3] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Trans. Netw.*, Aug. 1993.
[4] V. Firoiu and M. Borden, "A study of active queue management for congestion control," in *Proc. 2000 IEEE INFOCOM*.
[5] T. Ye and S. Kalyanraman, "Adaptive tuning of RED using on-line simulation," in *Proc. 2002 IEEE GLOBECOM*.
[6] W. Feng, D. Kandlur, D. Saha, and K. Shin, "A self-configuring RED gateway," in *Proc. 1999 IEEE INFOCOM*.
[7] T. Ziegler, S. Fdida, and C. Brandauer, "Stability criteria of RED with TCP traffic," Technical Report, May 2000. Available: http://www.salzburgresearch.at/∼cbrand/pub/report_redstability_00.pdf
[8] S. Floyd, R. Gummadi, and S. Shenker, "Adaptive RED: an algorithm for increasing the robustness of RED's active queue management," Technical Report, 2001. Available: http://www.icir.org/floyd/papers/adaptiveRed.pdf
[9] M. Goutelle, Y. Gu, and E. He, "A survey of transport protocols other than standard TCP," Data Transport Research Group, Apr. 2004. Available: https://forge.gridforum.org/forum/forum.php?forumid=410
[10] S. Kunniyur and R. Srikant, "Analysis and design of an adaptive virtual queue (AVQ) algorithm for active queue management," in *Proc. 2001 ACM SIGCOMM*.
[11] K. Ramakrishnan, S. Floyd, and D. Black, "The addition of explicit congestion notification (ECN) to IP," IETF RFC 3168, 2001.
[12] S. H. Low and D. E. Lapsley, "Optimization flow control I: basic algorithm and convergence," *IEEE/ACM Trans. Netw.*, Dec. 1999.
[13] L. Xu, K. Harfoush, and I. Rhee, "Binary increase congestion control (BIC) for fast long-distance networks," in *Proc. 2004 IEEE INFOCOM*.
[14] I. Rhee and L. Xu, "CUBIC: a new TCP-friendly high-speed TCP variant," in *Proc. 2005 PFLDNet*.
[15] S. Floyd, "High speed TCP for large congestion windows," RFC 3649, Aug. 2002. Available: ftp://ftp.rfc-editor.org/in-notes/rfc3649.txt
[16] D. Leith and R. Shorten, "H-TCP: TCP for high-speed and long-distance networks," in *Proc. 2004 PFLDNet*.
[17] T. Kelly, "Scalable TCP: improving performance in highspeed wide area networks," Feb. 2003. Available: http://wwwlce.eng.cam.ac.uk/ctk21/scalable/
[18] C. Jin, D. Wei, and S. Low, "FAST TCP: Motivation, architecture, algorithms, performance," in *Proc. 2004 IEEE INFOCOM*.
[19] S. Bhandarkar, S. Jain, and A. Reddy, "Improving TCP performance in high bandwidth high RTT links using layered congestion control," in *Proc. 2005 PFLDNet*.
[20] B. Wydrowski and M. Zukerman, "MaxNet: a congestion control architecture for maxmin fairness," *IEEE Commun. Lett.*, Nov. 2002.
[21] A. Jain and S. Floyd, "Quick-start for TCP and IP," IETF Internet Draft draft-amit-quick-start-02.txt, Oct. 2002.
[22] D. Katabi, M. Handley, and C. Rohrs, "Congestion control for high bandwidth-delay product networks," in *Proc. 2002 ACM SIGCOMM*.
[23] Y. Xia, L. Subramanian, I. Stoica, and S. Kalyanaraman, "One more bit is enough," in *Proc. 2005 ACM SIGCOMM*.
[24] X. Li and H. Yousefi'zadeh, "MPCP: multi packet congestion-control protocol," *ACM Computer Commun. Review*, Oct. 2009.
[25] P. Lassila and J. Virtamo, "Modeling the dynamics of the RED algorithm," in *Proc. 2000 ACM Quality Future Internet Services*.
[26] P. Kuusela, P. Lassila, J. Virtamo, and P. Key, "Modeling RED with idealized TCP sources," in *Proc. 2001 IFIP Performance Modeling Evaluation ATM IP Netw.*
[27] E. Gelenbe and G. Pujolle, *Introduction to Queueing Networks*. John Wiley & Sons, Inc., 1998.
[28] J. Roberto, U. Mocci, and J. Virtamo, "Broadband network traffic," Final Report of Action COST 242. Springer Verlag, 1996.
[29] A. Papoulis and S. U. Pillai, *Probability, Random Variables, and Stochastic Processes*, 4th edition. McGraw-Hill, 2002.
[30] D. P. Bertsekas, *Nonlinear Programming*, 2nd edition. Athena Scientific Publishing, 1999.
[31] D. F. Shanno, "Conditioning of quasi-Newton methods for function minimization," *Mathematics Computing*, vol. 24, pp. 647–656, 1970.
[32] —–, "The network simulator - NS2." Available: http://www.isi.edu/nsnam/ns/
[33] H. M. Alazemi, A. Mokhtar, and M. Azizoglu, "Stochastic approach for modeling random early detection gateways in TCP/IP networks," in *Proc. 2001 IEEE ICC*.
[34] T. Bonald, M. May, and J. Bolot, "Analytic evaluation of RED performance," in *Proc. 2000 IEEE INFOCOM*.

[2] The value of ε represents a design factor. We have experimented with a range of values in the interval $[0.1, 10]$ and observed that values close to 1 typically yield good results.

[35] D. Chiu and R. Jain, "Analysis of the increase/decrease algorithms for congestion avoidance in computer networks," *J. Computer Netw. ISDN*, June 1989.

[36] L. Guan, I. U. Awan, and M. E. Woodward, "Stochastic modeling of random early detection based congestion control mechanism for bursty and correlated traffic," in *Proc. 2004 IEE Software 151*.

[37] C. V. Hollot, V. Misra, D. Towsley, and W. Gong, "Analysis and design of controllers for AQM routers supporting TCP flows," *IEEE Trans. Autom. Control*, June 2002.

[38] V. Jacobson, "Congestion avoidance and control," in *Proc. 1988 ACM SIGCOMM*.

[39] R. Laalaoua and T. Czachorski, "Markovian model of RED mechanism," in *Proc. 2001 CCGRID*.

[40] S. Suthaharan, "Markov model based congestion control for TCP," Annual Simulation Symposium, 2004.

[41] Y.C. Wang, J. A. Jiang, and R. G. Chu, "Drop behaviour of random early detection with discrete-time batch Markovian arrival process," in *Proc. 2004 IEE Software 151*.

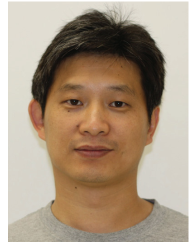
[42] J. M. Ortega and W. C. Rheinboldt, *Iterative Solutions of Nonlinear Equations in Several Variables.* Academic Press, 1970.

[43] I. W. Zangwill, *Nonlinear Programming*. Prentice-Hall, 1969

[44] P. Tseng, "Convergence of a block coordinate descent method for non-differentiable minimization," *J. Optimization Theory and Applications*, June 2001.

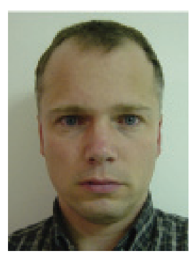
Homayoun Yousefi'zadeh received his B.S. degree from Sharif University of Technology in 1989, the M.S. degree from Amirkabir University of Technology in 1993, and the E.E.E. and Ph.D degrees from the University of Southern California in 1995 and 1997, respectively, all in electrical engineering. Currently, he is an Associate Adjunct Professor at the Department of EECS at UC, Irvine. He also holds a Chief Technologist position at the Boeing Company. In the recent past, he was the CTO of TierFleet, a Senior Technical and Business Manager at Procom Technology, and a Technical Consultant at NEC Electronics. He is the inventor of several US patents, has published more than sixty scholarly reviewed articles, and authored more than twenty design articles associated with deployed industry products. Dr. Yousefi'zadeh is with the editorial boards of the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS and the *Journal of Communications Networks*. Previously, he served as an editor of IEEE COMMUNICATIONS LETTERS, an editor of *IEEE Wireless Communications Magazine*, the lead guest editor of the IEEE JOURNAL OF SELECTED TOPICS ON SIGNAL PROCESSING, April 2008 issue, and the track chair as well as the TPC member of various IEEE and ACM conferences. He was the founding Chairperson of the systems management workgroup of the Storage Networking Industry Association, a member of the scientific advisory board of the Integrated Media Services Center at the University of Southern California (USC), and a member of the American Management Association. He is a senior member of IEEE.

Amir Habibi received his B.S. degree in computer science from Sharif University of Technology in 1994 and the M.S. degree in computer networks and distributed computing from the University of California, Irvine, in 2009. Throughout his professional career, he has worked in different areas of software engineering with a focus on distributed systems and applications. Some of his large scale software projects include implementation of NetBEUI and CIFS protocols for the Linux operating system, as well as the object-oriented scalable distributed message processing engine for interactive web applications. He has also worked on algorithms for optimized routing of ADA trips in the transportation industry. Currently, he is with Aryosys, a company he founded in 2010, specializing in a variety of design and information technology tools and services. His research interests include distributed processing, operating systems, and computer networks.



Xiaolong Li is currently a Research Specialist at the Department of EECS at UC, Irvine. He received his M.S. degree from the Department of Computer Science and Engineering at the University of Notre Dame in 2006, and his Ph.D. from the department of EECS at the University of California, Irvine, in 2009. His research interests are in the areas of wireless congestion control and wireless routing. Dr. Li is a member of IEEE.

Hamid Jafarkhani received the B.S. degree in electronics from Tehran University in 1989 and the M.S. and Ph.D. degrees, both in electrical engineering, from the the University of Maryland at College Park in 1994 and 1997, respectively. In 1997, he was a Senior Technical Staff Member at AT&T Labs-Research and was later promoted to Principal Technical Staff Member. He is currently a Chancellor's Professor at the Department of Electrical Engineering and Computer Science, University of California, Irvine, where he is also the Director of the Center for Pervasive Communications and Computing and the Conexant-Broadcom Endowed Chair. Dr. Jafarkhani ranked first in the nationwide entrance examination of Iranian universities in 1984. He was a co-recipient of the American Division Award of the 1995 Texas Instruments DSP Solutions Challenge. He received the best paper award of ISWC in 2002 and an NSF Career Award in 2003. He received the UCI Distinguished Mid-Career Faculty Award for Research in 2006 and the School of Engineering Fariborz Maseeh Best Faculty Research Award in 2007. Also, he was a co-recipient of the 2006 IEEE Marconi Best Paper Award in Wireless Communications and the 2009 best paper award of the *Journal of Communications and Networks*. He was an Associate Editor for IEEE COMMUNICATIONS LETTERS from 2001–2005, an editor for the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS from 2002–2007, an editor for the IEEE TRANSACTIONS ON COMMUNICATIONS from 2005–2007, and a guest editor of the special issue on "MIMO-Optimized Transmission Systems for Delivering Data and Rich Content" for the IEEE JOURNAL OF SELECTED TOPICS IN SIGNAL PROCESSING in 2008. Currently, he is an area editor for the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS. He is listed as a highly cited researcher in http://www.isihighlycited.com. According to Thomson Scientific, he is one of the top 10 most-cited researchers in the field of "computer science" during 1997–2007. He is a Fellow of AAAS, an IEEE Fellow, and the author of the book *Space-Time Coding: Theory and Practice*.



Claus Bauer received his M.S. and Ph.D. degrees in pure mathematics from the University of Freiburg, Germany, in 1994 and 1996, respectively. From 1994–1996, he was a visiting scholar at the Department of Mathematics, Shandong University, China. He has held research positions with Siemens, Tellabs, and Dolby Laboratories. Currently, he is heading Dolby's Research group in Beijing. His research interests are in analytic number theory, performance modeling, multimedia analysis and processing, switching and routing technologies, as well as wireless technologies.