Generalized Distributed Rate Limiting

Rade Stanojević, Robert Shorten
Hamilton Institute, NUIM, Ireland

***Abstract*— The Distributed Rate Limiting (DRL) paradigm is a recently proposed mechanism for decentralized control of cloud-based services. DRL is a simple and efficient approach to resolve the issues of pricing and resource control/engineering of cloud based services. The existing DRL schemes focus on very specific performance metrics (such as loss rate and fair-share) and their design heavily depends on the assumption that the traffic is generated by elastic TCP sources. In this paper we tackle the DRL problem for general workloads and performance metrics and propose an analytic framework for the design of stable DRL algorithms. The closed-form nature of our results allows simple design rules which, together with extremely low communication overhead, make the presented algorithm practical and easy to deploy with guaranteed convergence properties under a wide range of possible scenarios.**

***Index Terms*— Rate limiting, CDN, Cloud control, Consensus agreement, Stability and convergence.**

I. Introduction

In the early days of the Internet, services were centric, meaning that a user connects to the specific location that provides the service. Recently, we see a trend of moving from a centric model of providing services to a so called cloud-based model in which a user obtains a service from a massive network of "cloud servers". Nowadays, many internet services are structured in a "cloud" around a large number of servers that are distributed worldwide to decrease the costs and to improve content availability, robustness to faults, end-to-end delays, and data transmission rates [6]. Examples include most of Yahoo! and Google services, Amazon's Simple Storage Service (S3) and Elastic Compute Cloud (EC2) as well as Akamai's Content-Delivery Network (CDN). Some other applications, such as Google Docs or Microsoft Groove Office, have integrated software-as-a-service paradigm and allow desktop users to utilize cloud-based services in hosted environments.

The ability to control cloud-based service usage is critical for several important functions of a cloud-based service provider (CBSP):

(1) *The pricing* of service by most of the existing CBSPs is usage-based [1], [29]. Namely, services are charged at a rate that is an increasing (usually concave) function of the total resources used. However, in the history of communications, pricing of various services (eg. ordinary mail, the telegraph, the telephone, and the Internet) followed similar pattern: it started with usage-based pricing and converged to some form of flat-fee pricing. Moreover, enterprizes tend to prefer a fixed cost of an IT service rather than unlimited/unpredictable usage-based cost, see [8] and [23].

(2) *Provisioning* of high quality services depends on the nature of the service demand pattern. The ability to regulate the usage of individual service allows CBSPs to design networks with predictable performance bounds.

(3) *Fault tolerance* of large-scale distributed services is an important performance objective that is enhanced by resource control by means of fast fault discovery and quick response to these faults.

The paper [29] introduces the notion of Distributed Rate Limiting (DRL) as a mechanism for resource control in cloud-based services. Briefly, DRL stands for any mechanism that controls the aggregate service used by a customer of a cloud-based service. The idea is to enhance a set of cloud-servers with the ability to exchange information among them towards the global goal: control of the aggregate usage that a cloud-based service uses. The main obstacle in the design of a DRL algorithm is the fairness postulate [29]:

Fairness postulate: The performance levels at different servers should be (approximately) equal.

Thus, in DRL, a job utilizing one server competes for resource (bandwidth, CPU, RAM, storage, etc.) with jobs that utilize the same server and with all other jobs utilizing other servers (the formal definition of the problem is given in the section I-A). The worldwide scale[1] of such clouds raises important issues as to how to efficiently control resource usage in such large distributed environments.

The algorithms proposed in [29] and [32] deal with the DRL problem for very specific performance metrics: namely loss-rates and fair-share. They also assume that the traffic is generated by elastic, best-effort, sources that adapt their sending rate based on the available resources[2]. However, in many cases, the relevant performance metric is application dependent and it is also tied to the nature of the traffic sources (that can be elastic or non-elastic). For example, in VoIP servers the relevant performance metric could be the call-discard-rate, or latency, or some function of these two; in gaming applications it makes sense to evaluate the performance through experienced latency; etc. Basically, each application has its own performance goals, and requires some specific metric to measure the performance. See [30] for a nice overview of typical workloads and performance metrics in various real-world examples from networking, storage and operating systems. While existing DRL proposals show promise

[1]For example, Google's services run on several hundreds of thousands servers distributed worldwide [29], [5]. Akamai's content distribution network utilized tens of thousand servers [6].

[2]Throughout this paper we refer to workloads generated by elastic sources as *closed-loop* workloads. In contrast, *open-loop* workloads are generated by sources that do not adapt their sending rate based on the available resources [30], [28].

in several particular cases, it remains unclear how to design DRL algorithms that would be suitable for arbitrary workloads and performance metrics. In this paper our goal is to design scalable DRL algorithms that can be employed for arbitrary workloads and performance metrics.

From the theoretical point of view one can see DRL as dual to the well known concept of load balancing in the following sense. Suppose that there are N servers servicing the total demand $\mathcal{D}$ with the aggregate capacity C. In load balancing the capacities $(C_1, C_2, \ldots, C_N)$ of N servers are fixed and the problem is how to allocate the demand set $\mathcal{D}$ to N servers $(\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_N)$ $(\cup_{i=1}^{N} \mathcal{D}_i = \mathcal{D})$ such that the performance at each server is uniform. In DRL we have a fixed set of demands $(\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_N)$ and N servers, and the goal is to allocate the capacities at each server such that performance at each server is uniform (subject to the rate-invariance condition $\sum_{i=1}^{N} C_i = C = const$). Roughly speaking, load balancing can be seen as a demand partitioning, while DRL can be seen as a capacity partitioning. As we will see the distributed nature of DRL introduces a dynamic system approach for analyzing the queueing system arising in these generalized environments and solving the problem of interest that we now define.

A. Problem formulation

Let a cloud-based service provider controls N hosting centers with each hosting center $i \in \{1, 2, \ldots, N\}$ able to limit (throttle) locally the service rate C_i of a particular customer, serving the job population $\mathcal{D}_i$. The first constraint of DRL is to keep the aggregate service rate of all N servers for the given subscriber[3] at a prescribed level C.

$$\sum_{i=1}^{N} C_i = C. \tag{1}$$

Then, the local limiters (one at each server) should collaborate to realize the fairness postulate (formulated above). In order to formalize the fairness postulate we need to have a definition of the performance metric. The "performance metric" is a rather general notion. Examples include the mean response time, discard rate, utilization level, and "spare capacity". However, the framework described here is general and in what follows we will formalize the problem of interest.

Let q_i be the performance indicator (eg. "spare capacity", loss rate or mean response time) at server i serving (fixed) set of jobs $\mathcal{D}_i$ with (variable) capacity C_i. Then q_i is a function of $\mathcal{D}_i$ and C_i:

$$q_i = f(\mathcal{D}_i, C_i) =: f_i(C_i). \tag{2}$$

Now the DRL problem translates into finding $C_1, C_2, \ldots, C_N$ such that aggregate-rate invariance (1) is satisfied for which

$$q_1 = q_2 = \ldots = q_N. \tag{3}$$

In terms of communication infrastructure we allow each limiter i to cooperate with its neighbors in a connected undirected graph G (with nodes given by N limiters) to adapt its capacity limit C_i. The goal of our work is to develop a fully distributed algorithm (in which each server exchange only local information with its neighbors in G) that solves the system of equations (1) and (3). Our approach will be iterative, using local measurements, to drive the system to the solution independently of the initial state.

Before we proceed, we state the key technical assumption that will allow us to perform a detailed analysis of the dynamical system describing the DRL dynamics in the later sections.

Assumption 1: The functions $f_i : R \to R$, describing the relationship between capacity, C_i, and performance indicator q_i are decreasing, convex, differentiable functions, and have continuous first derivative[4].

We stress that for many performance indicators used in the literature, Assumption 1 is valid. Indeed for *spare-bandwidth* and *utilization*, it is straightforward to check the validity of the Assumption. A little bit more challenging is the case of mean-response-time for which it has been noticed empirically that for many scheduling strategies Assumption 1 is valid [28]. From an analytical perspective there are some partial results reported in [16], [17], but for general scheduling disciplines the convexity is still an open problem.

B. Our contributions

As we have said, the main concern of this paper is a principled design of algorithms for DRL under general performance indicators, and workload environments. Briefly, the main contributions of our work are following:

- The algorithm for solving the DRL problem under arbitrary performance metrics satisfying Assumption 1. The proposed algorithm has a very small communication overhead and can accommodate a wide range of performance metrics.
- The stability and convergence properties are analyzed for the presented algorithm and simple (closed form) design rules are derived.
- Empirical evaluation of the proposed scheme is presented, that supports our analytical findings.

The dynamical system that describes the dynamics of the algorithms are nonlinear and implicit. This makes the task of their analysis quite challenging. Namely, the standard theory of consensus algorithms (see [9] and references therein) cannot be employed in our case. The convergence result established in Theorem 1 is highly nontrivial and represents the main theoretical contribution of this paper.

We also performed a number of representative simulations to test the behavior of our algorithms in various settings. We found that various performance metrics closely match our analytical predictions capturing one of the goals of the present

[3]The subscriber is either an enterprize, bank, corporation, or any other set of users paying for service under one account.

[4]The set of differentiable functions $h : R \to R$, with continuous first derivative is usually denoted as $C^1(R)$.

paper: namely principled and performance-predictable design of DRL algorithms in open-loop workloads.

C. Related work

An early DRL-like proposal appeared in [10] which discussed a general framework for monitoring and control of distributed systems, with a particular application on Planetlab DRL control. The paper [29] introduces DRL in context of cloud-based service control. Here we briefly review two algorithms proposed in [29]: Global Random Drop (GRD) and Flow Proportional Share (FPS). GRD works as follows: each local limiter tracks its demand and broadcasts that information using the algorithm from [12]. Then the total demand T is computed as a sum of demands at all limiters and an arriving packet is dropped with probability $(T-C)/T$. As it is noted in [29], GRD exhibits poor performance for large number of limiters. To cite [29]: *"... Beyond 50 limiters, GRD fails to limit aggregate rate, but this is not assuaged with an increasing communication budget. Instead it indicates GRD's dependance on swiftly converging global arrival rate estimates."*. FPS works similarly, using the notion of node's "weight" that is broadcasted using the same gossiping algorithm that computes the aggregate weight at each node. Then each node utilizes this aggregate-weight information to adapt its behavior.

In [32] DRL algorithms are designed that emulate single best effort and processor sharing queues utilized by a set of elastic TCP users. In both scenarios (best-effort and processor sharing) the algorithms heavily exploit the TCP dynamics, and the design rules are heavily influenced by the square-root formula [26].

The problem of choosing relays in overlay networks, such as Skype, with the objective of making the performance on each of the relay nodes independent (of the choice of relay-node) has been studied recently in [22]. The presented solution utilizes the method of *two random choices* [20] and is essentially a load-balancing method neglecting the cost of choosing a relay for a given job. In the context of distributed cloud management and control, the authors of [35] proposed a tree-based algorithm for fast and reliable distributed identification of so-called threshold crossing events that are related to the DRL concept of keeping the global consumption bellow the threshold. Similar proposals to DRL can be found in the security literature as protection against DDoS [33], where a network of downstream routers throttles traffic to and from the server that is to be protected against DDoS.

The DRL problem, formulated in Section I-A, can be seen as instance of the consensus agreement problem. Consensus algorithms have attracted significant attention over last several years being applied in various topics, such as flocking [24], time synchronization, multi-agent coordination [9], sensor, peer-to-peer and ad hoc networks [4]. In most existing applications consensus algorithms can be modelled as positive linear systems, which then allow the elegant theory of nonnegative matrices and Markov chains to be employed to capture the convergence properties of the algorithms. However, little is known about implicit nonlinear consensus problems (see [13], [21]) and one of the main contributions of this paper is the proof of global stability for the implicitly given nonlinear systems describing the dynamics of the algorithm presented in the next section.

```
1  UpdateCapacities()
2       Once every Δ units of time do
3          for i = 1 : N
4             C_i ← C_i + η Σ_{(i,j)∈E} (q_i − q_j)
5          endfor
6       enddo

7    InitializeCapacities()
8       for i = 1 : N
9          C_i ← C/N
10      endfor
```

Fig. 1. Pseudo-code of GDRL

II. DRL UNDER GENERAL PERFORMANCE INDICATORS

We now present the GDRL algorithm that solves the DRL problem introduced in Section I-A: allocate the network resources in a manner that equalizes performance among local-limiters. As we said earlier, the rationale for doing this is to ensure that all end-users experience a similar quality of service.

Our basic setup is as follows. We use N local limiters to control aggregate service rate at level C. The local limiter i has a capacity C_i that can be adjusted, and this limiter can exchange information with the limiter j if (i,j) is an edge in the communication graph $G=(N,E)$ (in that case we write $(i,j)\in E$). For a given capacity C_i, and a family $\mathcal{D}_i$ of jobs, demanding the service from server i, the performance indicator q_i at limiter i can be directly measured, and is a function of $\mathcal{D}_i$ and C_i:

$$q_i = f(C_i, \mathcal{D}_i).$$

The goal here is to obtain a fully decentralized algorithm, for adjusting the values of C_i's such that

$$q_1 = q_2 = \cdots = q_N.$$

At each server we use a local limiter that throttles the service allocated to the subscriber at rate C_i. The performance indicator is measurable directly, and it depends on the demand pattern. In general the higher demand is at the limiter (meaning that aggregate aggressiveness is bigger) the "worse" the performance indicator is. Recall also that q_i is a decreasing function of C_i: $q_i = f_i(C_i)$.

The pseudo–code for the control of $(C_1,\ldots,C_N)$ is given in Figure 1. Initially, all the C_i are set by the $1/N$-rule. Then C_i is updated in discrete time steps by the simple rule: $C_i \leftarrow C_i + \eta\sum_{(i,j)\in E}(q_i - q_j)$. The rationale for this update step is the following. The q_i - the performance indicator of the quality of service is a decreasing function in terms of C_i. If the q_i at limiter i is higher than performance indicator q_j at some neighbor j of i (in G), then this indicates that some extra capacity should be allocated to limiter i which should be compensated by reducing the capacity of limiter j. Giving more capacity to limiters with high performance indicators affects improving the performance at those limiters.

The parameter $\eta > 0$ determines responsiveness and stability properties of the algorithm and its choice is discussed in the next subsection.

While the basic algorithm makes sense intuitively, many questions need to be answered before it can be deployed. Paramount among these concerns under which conditions does the algorithm GDRL converge to the desired (unique) equilibrium, and if so, how fast. These questions provide the focus for the investigation presented in the next section.

A. Model and analysis of GDRL

In this section we analyze the GDRL in a very general model. Thus we do not make any assumptions on the scheduling scheme utilized at any of the schedulers, neither on the job size distributions, job arrival pattern, etc. The only technical assumption is one stated as Assumption 1 that ensures smoothness, monotonicity and convexity of the functions relating capacities and the performance indicators.

Now, suppose that limiter i serves the population of jobs $\mathcal{D}_i$. We model the DRL system in discrete time t. At time t, denote by $C_i(t)$ the capacity and by $q_i(t)$ the performance indicator and limiter i. From the discussion in Section I-A we know that there is a function f_i such that

$$q_i(t) = f_i(C_i(t)). \tag{4}$$

Let us denote with $g_i = f_i^{-1}$ the inverse function of f_i (defined on the appropriate domain). Then

$$C_i(t) = g_i(q_i(t)). \tag{5}$$

Equation (4) represents the key relationship between $C_i(t)$ and $q_i(t)$. Given this, the dynamical system describing the evolution of $C_i(t)$, is given by:

$$C_1(0) = C_2(0) = \ldots = C_N(0) = C/N, \tag{6}$$

$$C_i(t+1) = C_i(t) + \eta \sum_{(i,j)\in E} (q_i(t) - q_j(t)). \tag{7}$$

The following lemma is a straightforward consequence of the fact that G is an undirected graph.

Lemma 1: For all t, the capacity constraint is satisfied:

$$C_1(t) + C_2(t) + \ldots + C_N(t) = C. \tag{8}$$

Proof: For $t = 0$ the statement is true from the definition. Suppose that it is valid for $t = k$, then for $t = k + 1$:

$$\sum_{i=1}^{N} C_i(t+1) = \sum_{i=1}^{N} \left[C_i(t) + \eta \sum_{(i,j)\in E} (q_i(t) - q_j(t)) \right] =$$

$$\sum_{i=1}^{N} C_i(t) + \eta \sum_{(i,j)\in E} ((q_i(t) - q_j(t)) + (q_j(t) - q_i(t))) =$$

$$= \sum_{i=1}^{N} C_i(t) = C.$$

■

The following theorem gives a sufficient condition under which system (6)-(7) converge.

Theorem 1: Let d_i be the degree of limiter i in the communication graph. Then if η satisfies:

$$0 < \eta < \frac{1}{2} \min_{1\le i\le N}(-g_i'(q_i(0))) \min_{1\le i\le N} \frac{1}{d_i}, \tag{9}$$

the following limits exist

$$\lim_{t\to\infty} C_i(t) = C_i^*$$

and

$$\lim_{t\to\infty} q_i(t) =: q^*.$$

Proof: We find it more convenient to write the dynamics of (7) in terms of $q_i(t)$:

$$g_i(q_i(t+1)) = g_i(q_i(t)) + \eta \sum_{(i,j)\in E} (q_i(t) - q_j(t)), \tag{10}$$

We denote

$$m(t) = \min_{1\le i\le N} q_i(t),$$

and

$$M(t) = \max_{1\le i\le N} q_i(t).$$

Step 1. First we prove that under condition (9) the sequence $m(t)$ is nondecreasing and the sequence $M(t)$ is nonincreasing.

Let $q_i(t) = m(t) + \lambda$, for some $\lambda \ge 0$. Then from the equation (10) we have:

$$g_i(q_i(t+1)) = g_i(q_i(t)) + \eta \sum_{(i,j)\in E} (m(t) + \lambda - q_j(t)) \le$$

$$g_i(q_i(t)) + \eta \sum_{(i,j)\in E} \lambda = g_i(q_i(t)) + \eta\lambda d_i =$$

$$= g_i(m(t) + \lambda) + \eta\lambda d_i. \tag{11}$$

On the other hand, since by Assumption 1 f_i is in C^1, decreasing and convex, g_i is also differentiable decreasing and convex. Therefore from the mean value theorem, there exists $q_i^*(t) \in (m(t), m(t) + \lambda)$ such that

$$g_i(m(t)+\lambda) = g_i(m(t)) + g_i'(q_i^*(t))\lambda \le g_i(m(t)) + g_i'(m(t))\lambda$$

Combining the last inequality with (11) we get

$$g_i(q_i(t+1)) \le g_i(m(t)) + g_i'(m(t))\lambda + d_i\eta\lambda. \tag{12}$$

For $t = 0$, the condition (9) implies that $g_i'(m(0))\lambda + d_i\eta\lambda \le 0$ implying that $g_i(q_i(1)) \le g_i(m(0))$. Since g_i is decreasing, we have $q_i(1) \ge m(0)$ for all i and $m(1) \ge m(0)$. Now we prove by using mathematical induction that $m(t+1) \ge m(t)$. For $t = 0$ we just proved that $m(1) \ge m(0)$. Suppose that for all $k < t$: $m(k+1) \ge m(k)$. Then for $k = t$, $m(t) \ge m(0)$. Now, the condition (9) and the bound (12) imply:

$$g_i(q_i(t+1)) \le g_i(m(t)) + g_i'(m(t))\lambda + d_i\eta\lambda \le$$

$$g_i(m(t)) + g_i'(m(0))\lambda + d_i\eta\lambda \le g_i(m(t)).$$

The last inequality, together with the fact that g_i is a decreasing function implies

$$m(t+1) = \min_{1\le i\le N} q_i(t+1) \ge m(t).$$

One can conclude using similar arguments that for all $t \ge 0$

$$M(t+1) = \max_{1\le i\le N} q_i(t+1) \le M(t).$$

Thus the range of q_i's: $[m(t), M(t)]$ is a nested sequence of closed intervals. In the next step of the proof we use this information to write the dynamics in the more convenient form.

Step 2. In this step we rewrite the dynamics of $q_i(t)$ in a more practical form. Consider the representation of the dynamics of $q_i(t)$ given by (10). From the Lagrange's mean value theorem there exist $r_i(t) \in (q_i(t+1), q_i(t)) \subset (m(0), M(0))$ such that

$$g_i(q_i(t+1)) - g_i(q_i(t)) = (q_i(t+1)) - q_i(t))g_i'(r_i(t)).$$

Thus the recurrence equation (10) can be rewritten as

$$q_i(t+1) = q_i(t) + \frac{\eta}{g_i'(r_i(t))} \sum_{(i,j)\in E} (q_i(t) - q_j(t)).$$

Therefore, the evolution of the state-vector $\mathbf{q}(t) = (q_1(t), \dots, q_N(t))$ can be written as:

$$\mathbf{q}(t+1) = B(t)\mathbf{q}(t),$$

where the matrix $B(t)$ is given by $B(t) =$

$$\begin{bmatrix} 1+\frac{d_1\eta}{g_1'(r_1(t))} & -\frac{\eta}{g_1'(r_1(t))}e_{1,2} & \cdots & -\frac{\eta}{g_1'(r_1(t))}e_{1,N} \\ -\frac{\eta}{g_2'(r_2(t))}e_{2,1} & 1+\frac{d_2\eta}{g_2'(r_2(t))} & \cdots & -\frac{\eta}{g_2'(r_2(t))}e_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ -\frac{\eta}{g_N'(r_N(t))}e_{N,1} & \cdots & \cdots & 1+\frac{d_N\eta}{g_N'(r_N(t))} \end{bmatrix}$$

with $e_{i,j}$ being the elements of the adjacency matrix of G, ie. if $(i,j) \in E$, then $e_{i,j} = 1$ otherwise $e_{i,j} = 0$.

Step 3. We now use the monotonicity of sequences $M(t)$ and $m(t)$ proved in Step 1, to prove that nonzero elements of $B(t)$ are positive and uniformly bounded away from zero. Indeed, recall that $r_i(t) \in (q_i(t), q_i(t+1))$, and therefore $r_i(t) \ge \min(m(t), m(t+1)) \ge m(0)$, and therefore for the diagonal entries we have that

$$1 + \frac{d_i\eta}{g_i'(r_i(t))} \ge 1 + \frac{d_i\eta}{g_i'(m(0))} \ge 1 - \frac{1}{2} = \frac{1}{2}.$$

For nonzero off-diagonal entries note that $r_i(t) \le \max(M(t), M(t+1)) \le M(0)$ and thus

$$-\frac{\eta}{g_i'(r_i(t))} \ge \frac{\eta}{-g_i'(M(t))} \ge \frac{\eta}{-g_i'(M(0))} =: \delta_i > 0.$$

Take $\delta = \min\{\frac{1}{2}, \delta_1, \dots, \delta_N\} > 0$. Then for all t, all nonzero elements of $B(t)$ are not smaller than δ.

Step 4. Finally, we use the fact that G is connected to show that $M(t) - m(t)$ converges to zero. This implies that $\lim_{t\to\infty} m(t) = \lim_{t\to\infty} M(t) = \lim_{t\to\infty} q_i(t) = q^*$. Let k be the diameter of graph G, i.e. the smallest integer such that there exist a path in G between each two nodes of length not greater than k. Then for all t:

$$D(t) = B(t+k-1)B(t+k-2)\cdots B(t)$$

is a stochastic matrix[5] with strictly positive entries and each entry of $D(t)$ is greater or equal than δ^k. Denote by j_0 the index for which $q_{j_0}(t) = m(t)$. Then

$$q_i(t+k) = \sum_{j=1}^N D_{ij}(t)q_j(t) = \sum_{j\ne j_0} D_{ij}(t)q_j(t) + D_{ij_0}(t)m(t)$$

$$\le \sum_{j\ne j_0} D_{ij}M(t) + D_{ij_0}(t)m(t) =$$

$$(1 - D_{ij_0}(t))M(t) + D_{ij_0}(t)m(t) \le M(t)(1-\delta^k) + m(t)\delta^k.$$

And similarly

$$q_i(t+k) = \sum_{j=1}^N D_{ij}(t)q_j(t) \ge m(t)(1-\delta^k) + M(t)\delta^k.$$

Thus

$$M(t+k) - m(t+k) \le (1-2\delta^k)(M(t) - m(t)). \quad (13)$$

Since $M(t) - m(t)$ is a nonincreasing sequence and $\delta > 0$ is independent of t, we conclude that $M(t) - m(t) \to 0$, as $t \to \infty$. Thus

$$\lim_{t\to\infty} M(t) = \lim_{t\to\infty} m(t) = \lim_{t\to\infty} q_i(t) = q^*.$$

Now, the convergence of $C_i(t)$ follows directly from (5) and the continuity of the mappings g_i. ■

Comment 1: From the bound (13), we can observe that the system converges to the equilibrium geometrically, with a rate bounded above by $(1-2\delta^k)^{\frac{1}{k}}$. Indeed, let us introduce the following quantity:

$$\theta = \frac{1}{1-2\delta^k} \max_{0\le s\le k} (M(s) - m(s)).$$

Then from (13):

$$M(t) - m(t) \le (1-2\delta^k)^{\lfloor \frac{t}{k} \rfloor}(M(s) - m(s)) \le$$

$$(1-2\delta^k)^{\frac{t}{k}} \frac{1}{1-2\delta^k}(M(s) - m(s)) \le \theta \left((1-2\delta^k)^{\frac{1}{k}}\right)^t.$$

Comment 2: In the networking community, a widely used approach (see [18], [34]) for analysis of nonlinear dynamical systems is linearization around the equilibrium, and presentation of some kind of local stability result: if system is close to equilibrium then it will stay there. We stress that our result posses an extra feature, it says that the system will actually reach the equilibrium from an initial

[5]A stochastic matrix is a square matrix with nonnegative entries and sum of each row is 1. Since each of $B(t)$ is stochastic, their product is stochastic as well [2].

N	number of servers
C_i	service rate at node i
C	the aggregate service rate
q_i	performance indicator at node i
d_i	degree of node i in the graph G
λ_i	traffic intensity at node i
MRT_i	mean response time at node i
JFI	Jain's fairness index
η	the gain parameter
α	the filtering parameter

TABLE I
SYMBOL MAP

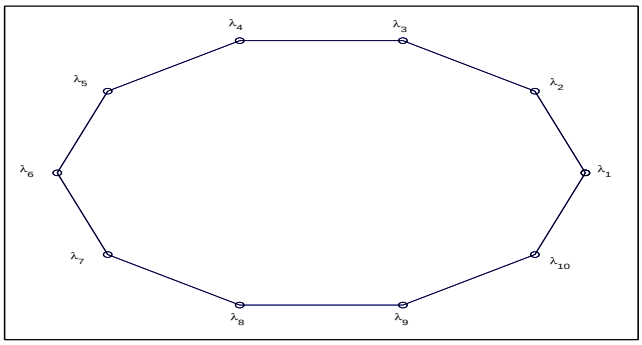

Fig. 2. The ring structure of the communication graph used for the simulations in Sections III-A and III-B.

state lying outside of the linearization regime.

Comment 3: While in the presented model we assume synchronous updates of C_i, this property of the model is not critical. See Section IV and Appendix for more details.

Comment 4: The presented algorithm, GDRL, can be seen as instance of distributed equation solving. Suppose that N agents want to solve the following equation in a distributed manner

$$G(x) = \sum_{i=1}^{N} g_i(x) = C.$$

If each agent i is able to solve the equation $g_i(x) = y$ for every y then GDRL-like algorithm with appropriate η converges to the solution of the above equation.

III. EVALUATION

A. Basic setup

In our first simulation we use a network of N servers communicating along the edges of d-regular[6] graph. Limiter i receives packets arriving according to a Poisson process with intensity λ_i. The service rate is given by $C_i(t)$ and each limiter represents $M/M/1$ queue with arrival and service rate specified above. Is such system, the mean-response time at limiter i is given by

$$MRT_i(t) = \frac{1}{C_i(t) - \lambda_i}.$$

Since we find it easier to directly estimate the arrival rate than the expected response-time, we will use as performance indicator the "spare-bandwidth":

$$q_i = f_i(C_i) := \lambda_i - C_i. \tag{14}$$

We use the low-pass filter to estimate λ_i with parameter α:

$$\hat{\lambda}_i(t+1) = (1-\alpha)\hat{\lambda}_i(t) + \alpha\delta(t),$$

where $\delta(t)$ is the random variable that corresponds to the job arrival process, ie. $\delta(t) = 1$ if a packet arrived at the time slot t and $\delta(t) = 0$ otherwise.

Using "spare-bandwidth" as a performance indicator as defined above, the function $f_i : x \to x - \lambda_i$ clearly satisfies Assumption 1 making Theorem 1 applicable for obtaining a sufficient condition on η for stability. Since the g_i function has form:

$$C_i = g_i(q_i) := \lambda_i - q_i.$$

The condition (9) on η that ensures stability of the algorithm translates simply to:

$$\eta \leq \min \frac{1}{2d_i} = \frac{1}{2d}. \tag{15}$$

In order to measure how "equal" the performance indicators $(q_1(t), \ldots, q_N(t))$ are, we use the quantity in the networking literature known as Jain's Fairness Index (JFI)[11]:

$$JFI(\mathbf{q}(t)) = \frac{\left(\sum_{i=1}^{N} q_i(t)\right)^2}{N \sum_{i=1}^{N} q_i^2(t)}. \tag{16}$$

JFI is a quantity that lies between 0 and 1, and the closer JFI is to 1 the "more" uniform elements of the vector are.

The simulation setup we run consisted of $N = 10$ limiters, communicating over the edges of graph G that has a ring structure (each node has two neighbors, $d = d_i = 2$). The demand intensity at node i is

$$\lambda_i = \frac{i}{N+1}, \text{ for } i = 1, 2, \ldots, N.$$

The aggregate service rate C is 10% larger than the aggregate traffic intensity $\Lambda = \sum_{i=1}^{N} \lambda_i$. The filtering parameter is set to $\alpha = 10^{-3}$ and the gain parameter is chosen at the level that guarantees stability $\eta = \frac{1}{4}$.

Figures 3 and 4 depict the evolution of the vector of performance indicators: $\mathbf{q}(t) = (q_1(t), \ldots, q_N)$ and its JFI. As we can see the components of vector $\mathbf{q}(t)$ converge to the (approximately) same value and the metric measuring how uniform those values are - $JFI(\mathbf{q}(t))$, converge to the region very close to 1. The offset between the measured JFI and 1 is caused by the noise in the estimation of real λ_i and can be made arbitrarily small by choosing small enough low-pass filter parameter α.

B. Dynamic demands

In this simulation we evaluate the effects of the change in the demand pattern. We use the same configuration of $N =$

[6]A graph is d-regular if its every node has degree d.

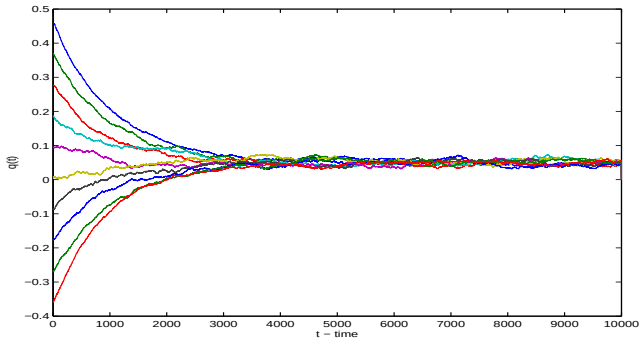

Fig. 3. The evolution of the vector $\mathbf{q}(t) = (q_1(t), \ldots, q_N)$.

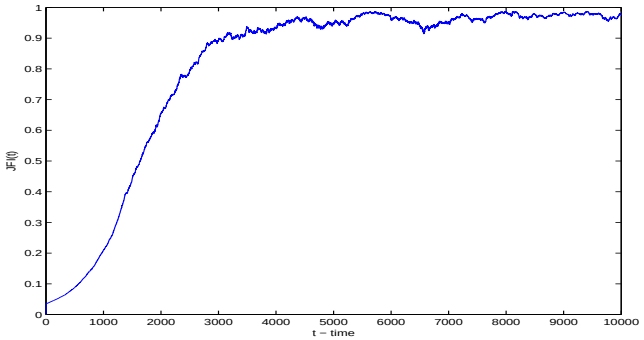

Fig. 4. $JFI(\mathbf{q}(t))$ dynamics.

10 nodes with ring communication structure and performance indicator function given by (14). The GDRL parameters used are $\eta = \frac{1}{4}$ and $\alpha = 10^{-3}$ and the load is 90%.

The demands, λ_i, in previous subsection are constant. In this subsection we vary the demands in time t:

$$\lambda_i(t) = \frac{i}{N+1}, \text{ for } t \in [0, \tau_0] \cup [3\tau_0, 4\tau_0],$$

$$\lambda_i(t) = 1 - \frac{i}{N+1}, \text{ for } t \in [\tau_0, 2\tau_0],$$

$$\lambda_i(t) = \frac{t-2\tau_0}{\tau_0}(1-\frac{i}{N+1}) + \frac{3\tau_0 - t}{\tau_0}\frac{i}{N+1}, \text{ for } t \in [2\tau_0, 3\tau_0],$$

$\tau_0 = 10^4$. Figure 5 depicts the $\lambda_i(t)$ as well as the estimates obtained using the low pass filter.

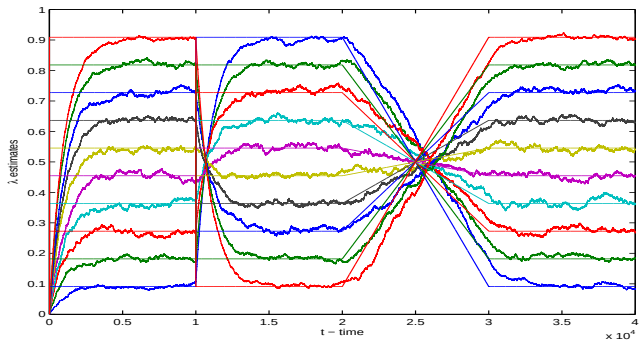

Fig. 5. Estimates of $\lambda_i(t)$ and real values of $\lambda_i(t)$ (for each i, $\lambda_i(t)$ is a piecewise linear function consisted of 4 segments).

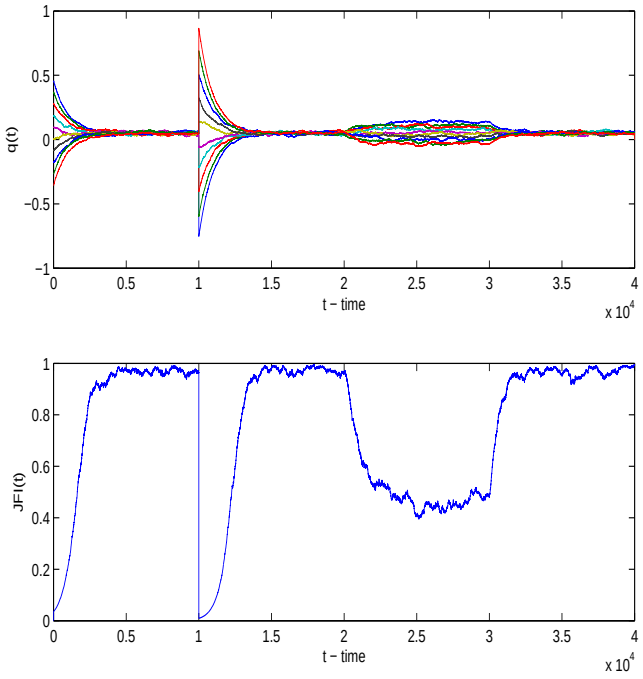

Fig. 6. The evolution of $\mathbf{q}(t)$ and $JFI(\mathbf{q}(t))$.

The GDRL algorithm is run with the η parameter specified above and the resulting $\mathbf{q}(t)$ and $JFI(\mathbf{q}(t))$ is depicted in Figure 6. As we have said, this simulation presents the behavior of the GDRL in the events of changes in the demand pattern. Both an abrupt change (discontinuous change in $\lambda(t)$ at $t = \tau_0$) and a slow smooth change (linear shift of λ's during the interval $[2\tau_0, 3\tau_0]$) have been evaluated. As it is predicted by Theorem 1, GDRL stabilizes the $\lambda_i(t)$ after the transient periods caused by the mentioned changes.

C. Scalability in larger networks

The 10-node ring structure depicted in Figure 2 that was used in the previous subsection is somewhat small. In this subsection we evaluate how GDRL scales in larger networks.

We borrow the setup established in the Section III-A, using random d-regular communication graphs, low pass filtering parameter $\alpha = 10^{-3}$, load at 90% and the gain parameter η set at the value that guarantees stability (15). We use $N = 10, 100, 1000$, spanning 2 orders of magnitude and $d = 2, 4, 8$. The evolution of $JFI(\mathbf{q}(t))$ is depicted for each of those nine cases in Figure 7. Notice that convergence to the steady-state is slowest for $d = 2$. However, a somewhat surprising observation is that for gain parameter set at the stability condition, $\eta = \frac{1}{2d}$, the speed of convergence is very close for a range of d, implying that low d exhibits convergence that is (almost) as fast as the one for large d. Therefore, a cheap ring structure, with a very low communication overhead appears to converge as quickly as dense structures. We do not have an analytical explanation for this phenomenon, and we will seek one as a part of future work.

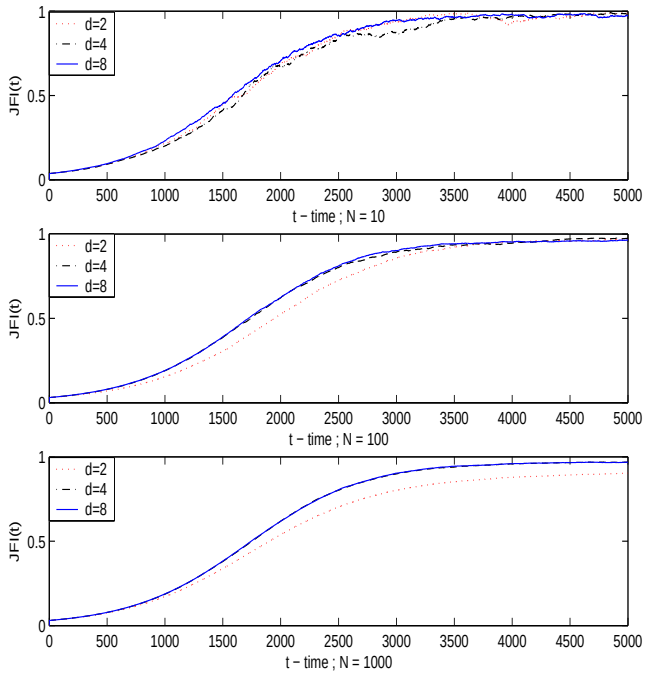

Fig. 7. $JFI(\mathbf{q}(t))$ dynamics in random d-regular graphs of 10, 100 and 1000 nodes. $d \in \{2, 4, 8\}$.

IV. Implementation issues

Asynchronous updates. The algorithm GDRL assumes that local-limiter capacities are updated in a synchronized manner using a connected undirected communication graph G. However, this is not necessary to ensure convergence. To see this, suppose that in each time instant t only some subset of nodes exchange information, and that that information exchange is characterized by an undirected graph G_t. If there is some $T > 0$ such that for every τ, $\cup_{t=\tau}^{\tau+T} G_t$ is connected, then the method developed in [9] can be used to establish the convergence of the GDRL. See Appendix for formal description of the asynchronous model.

Message passing. Communication between two local limiters is performed via small packets containing information on the performance indicator as well as some control overhead to ensure that if a loss of a communication packet occurs no local limiter gains or loses extra capacity, and that the capacity constraint (1) is not violated.

Communication delays. Message passing between two local limiters causes some communication delay on a time scale from few milliseconds up to a couple of hundreds of milliseconds. These communication delays could cause some issues related to the stability of the distributed algorithms if the update interval is on some small time scale. However, the time between updates, given by Δ, is on the order of magnitude of several seconds. This is necessary to obtain a good estimate of performance indicators. This resulting separation of time-scales ensures that effects of the communication delays on the stability of our algorithms may be neglected. Notwithstanding this fact, the issue of delays is a topic for future research.

Node Failures. In cases of a node (local-limiter) failure, it is possible for a loss of aggregate bandwidth to occur (since the capacity constraint (1) would be violated). A simple method for resolving this issue is the following. Let each local-limiter i choose a *best-friend* local-limiter[7] b_i among neighbor nodes in the communication graph G, and let each node inform node b_i of its local rate limit C_i. In the case of failure, local limiter b_i inherits bandwidth of node i, by simply setting $C_{b_i} = C_{b_i} + C_i$. Then the algorithms themselves will eventually adapt capacities of the non-failed limiters to the desired regime.

Performance indicator estimation. The choice of performance indicators has a key effect on the results of the GDRL algorithm. What is relevant performance indicator is somewhat driven by the application needs and it is hard to isolate the single performance metric usable in all conditions; see, for example [25]. This was the main motivation for the general presentation in the previous sections. However, often the performance indicator is a function of a random variable which needs to be estimated. While some performance metrics can be estimated quickly and accurately there are also cases in which the direct estimation of the performance metric can require many samples to give an accurate estimate. Example of this is the estimation of the mean waiting time in $M/M/1$ queue, see Chapter 11 of [19] and references therein.

V. Summary

Issues related to service reliability, service availability, and fault tolerance, have encouraged many service providers in the Internet to shift from traditional centric services to cloud based services. This trend appears to be a dominant mechanism for ensuring robustness of internet services with many "big players", such as Google, Yahoo!, Akamai, Amazon, already offering a suit of cloud-based services.

Pricing, usage control, and resource allocation of cloud based services represent important technical challenges for the networking community. The Distributed Rate Limiting paradigm is a step forward in resolving those issues. The DRL algorithms presented in [29] and [32] deal with the closed-loop workloads in which the job demands are elastic (driven by the best-effort TCP users). Therefore, those algorithms utilize TCP-related quantities (loss-rates, TCP-"weight", etc.) and are not suitable for non-elastic (open-loop) workloads. This motivated us to develop GDRL, the algorithm for solving the DRL problem in general workloads utilizing a very general notion of performance metric. Namely, our analysis shows that GDRL solves the DRL problem for any performance indicator that is a convex smooth function of the allocated server capacity. Theorem 1 provides a closed form expression that guarantees the stability that largely simplifies the design. Our evaluation section illustrated the behavior of GDRL in the simple $M/M/1$ scenario, showing that simulation results match the analytical predictions. We conclude the discussion of the paper raising two open questions related to the design of DRL algorithms.

Open question 1. *Unified framework for load balancing and DRL algorithms.* In Section I we briefly discussed the

[7]Note that if j is best-friend of node i, that it does not necessarily mean that i is the best-friend of node j.

connection between load balancing and DRL. Both paradigms strive to equalize the performance on different servers; load balancing achieves that by allocating the jobs to different servers while DRL allocates the service capacity to different servers. Can we unify those two paradigms into one framework that takes into account the cost of allocating certain job to the certain server (or some other approach)?

Open question 2. *Kelly-like framework for DRL algorithms.* Is there a nice interpretation of DRL algorithms through the convex optimization Kelly-like framework [31]?

APPENDIX

Suppose that instead of synchronous updates, GDRL updates the vector $\mathbf{C}(t) = (C_1(t), \ldots, C_N(t))$ in an asynchronous manner such that within each time interval I of length δ the union of all edges over which a communication has been established during I is a connected graph. Formally the asynchronous model we consider is characterized by the following:

Assumption 2: Let $\tau_1 < \tau_2 < \ldots < \tau_t < \ldots$ be instances of time at which communication between two nodes appear. Denote by $e_t = \{v_t', v_t''\}$ the edge over which the communication is established during the time instance t_k. Let $\mathbf{C}(t)$ be the vector of local capacities at time instance τ_t. Then in the asynchronous model we allow updates of the following form:

$$C_i(t) = C_i(t-1), \text{ if } i \notin \{v_t', v_t''\}$$

and

$$C_i(t) = C_i(t-1) + \eta(q_i(t-1) - q_j(t-1)), \text{ if } \{i,j\} = \{v_t', v_t''\}.$$

Then we have the following result analogous to the Theorem 1.

Theorem 2: Suppose that vector $\mathbf{C}(t)$ of local capacities is updated in the asynchronuous model defined by Assumption 2. Then if η satisfies:

$$0 < \eta < \frac{1}{2} \min_{1 \le i \le N} (-g_i'(q_i(0))), \tag{17}$$

the following limits exist

$$\lim_{t \to \infty} C_i(t) = C_i^*$$

and

$$\lim_{t \to \infty} q_i(t) =: q^*.$$

The proof can be derived following the lines of the proof of Theorem 1 and is omitted here.

REFERENCES

[1] Amazon Elastic Compute Cloud (EC2): http://aws.amazon.com/ec2.

[2] A. Berman, R. Plemmons. "Nonnegative matrices in the mathematical sciences". SIAM, 1979.

[3] S. Bhatnagar, B. Nath. "Distributed admission control to support guaranteed services in core-stateless networks". IEEE INFOCOM, 2003.

[4] S. Boyd, A. Ghosh, B. Prabhakar, D. Shah. "Gossip algorithms: Design, analysis and applications". In Proc. of IEEE INFOCOM, 2005

[5] D. F. Carr. "How Google works". Baseline Magazine, July 2006.

[6] Akamai. "The state of the Internet: 1st Quarter 2008". Vol 1(1).

[7] N. Duffield, P. Goyal, A. Greenberg, P. Mishra, K. Ramakrishnan, J. van der Merive "A flexible model for resource management in virtual private networks". In Proc. of ACM SIGCOMM 1999.

[8] D. Hinchcliffe. "2007: The year enterprises open thier SOAs to the Internet". Enterprise Web 2.0, Jan. 2007.

[9] A. Jadbabaie, J. Lin, and A. S. Morse, "Coordination of groups of mobile autonomous agents using nearest neighbor rules". IEEE Transactions on Automatic Control, vol. 48(6), 2003

[10] A. Jain, J. M. Hellerstein, S. Ratnasamy, D. Wetherall. "A wakeup call for internet monitoring systems: The case for distributed triggers". In Proc. of HotNets-III, 2004.

[11] R. Jain. "The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling". John Wiley and Sons, INC., 1991.

[12] D. Kempe, A. Dobra, J. Gehrke. "Gossip-based computation of aggregate information". In Proc. of IEEE FOCS, 2003.

[13] C. King, R. Shorten, F. Wirth, M. Akar. "Growth Conditions for the Global stability of Highspeed Communication Networks". To appear in IEEE Transactions on Automatic Control, 2008.

[14] M. Kodialam, T. Lakshman, S. Sengupta. "Maximum Throughput Routing of Traffic in the Hose Model". In Proc. of IEEE INFOCOM 2006.

[15] A. Kumar, R. Rastogi, A. Siberschatz, B. Yener. "Algorithms for provisioning virtual private networks in the hose model". IEEE/ACM Trans. on Networking, vol 10(4), 2002.

[16] K. Kumaran, M. Mandjes. "The buffer-bandwidth trade-off curve is convex". Queueing Systems, vol. 38(4), 2001.

[17] K. Kumaran, M. Mandjes, A. Stolyar. "Convexity properties of loss and overflow functions". Operations Research Letters, vol. 31(2), 2003.

[18] S. Kunniyur, R. Srikant. "Analysis and Design of an Adaptive Virtual Queue (AVQ) Algorithm for Active Queue Management". *IEEE/ACM Trans. on Networking*, vol. 12(2).

[19] S. Meyn. "Control techniques for the complex networks". Cambridge University Press, Cambridge, 2008.

[20] M. Mitzenmacher. "The Power of Two Choices in Randomized Load Balancing". IEEE Transactions on Parallel and Distributed Systems, vol. 12(10), 2001.

[21] L. Moreau. "Stability of multiagent systems with time-dependent communication links". IEEE Transactions on Automatic Control, 2005.

[22] H. X. Nguyen, D. Figueiredo, M. Grossglauser, P. Thiran. "Balanced Relay Allocation on Heterogeneous Unstructured Overlays". In Proc. of IEEE Infocom 2008, Phoenix, AZ, USA.

[23] A. Odlyzko. "Internet pricing and the history of communications". Computer Networks, vol. 36, 2001.

[24] R. Olfati-Saber. "Flocking for multi-agent dynamic systems: algorithms and theory". IEEE Trans. on Auto. Control, 2006.

[25] T. Osogami. "Accuracy of measured throughputs and mean response times". In Proc. of MAMA 2007, San Diego, CA, USA.

[26] J. Padhye, V. Firoiu, D. F. Towsley, J. F. Kurose. "Modeling TCP Reno performance: a simple model and its empirical validation". *IEEE/ACM Trans. on Networking*, vol 8(2), 2000.

[27] R. S. Prasad, C. Dovrolis. "Measuring the Congestion Responsiveness of Internet Traffic. In Proc. of PAM 2007.

[28] R. S. Prasad, C. Dovrolis. "Beyond the Model of Persistent TCP Flows: Open-Loop vs Closed-Loop Arrivals of Non-persistent Flows". In Proc. of Annual Simulation Symposium, 2008.

[29] B. Raghavan, K. Vishwanath, S. Rambhadran, K. Yocum, A. Snoeren. "Cloud Control with Distributed Rate Limiting". In Proc. of ACM SIGCOMM 2007.

[30] B. Schroeder, A. Wierman, M. Harchol-Balter. "Open vs. closed systems: A cautionary tale". In Proc. of NSDI 2006.

[31] R. Srikant. "Internet congestion control". Control theory, 14, Birkhäuser Boston Inc., Boston, MA, 2004.

[32] R. Stanojevic, R. Shorten.""Fully decentralized emulation of best-effort and processor sharing queues". In Proc. of ACM SIGMETRICS 2008.

[33] C.W. Tan, D.M. Chiu, J.C.S. Lui, D.K.Y. Yau. "A Distributed Throttling Approach for Handling High Bandwidth Aggregates". IEEE Transactions on Parallel and Distributed Systems, vol 18(7), 2007.

[34] D. Wei, C. Jin, S. Low, S. Hegde. "FAST TCP: motivation, architecture, algorithms, performance". IEEE/ACM Trans. on Networking, 2007.

[35] F. Wuhib, M. Dam, R. Stadler. "Decentralized Detection of Global Threshold Crossings Using Aggregation Trees". Computer Networks, vol. 52(9), 2008.