Decentralized Control and Optimization of Networks with QoS-Constrained Services

Iraj Saniee
Bell Laboratories, Alcatel-Lucent
600 Mountain Avenue, Murray Hill, NJ 079741
iis@research.bell-labs.com

Abstract—We consider data networks in which real-time/near real-time applications require not only successful transmission of packets from source to destination, but also specific end-to-end delay bounds, such as voice over IP. Although there is a well-developed general theory for control of best-effort packet traffic in data networks (elastic traffic), little is known about decentralized control mechanisms that ensure end-to-end performance bounds (inelastic traffic). In this paper we propose and analyze a simple, distributed and self-stabilizing rate control scheme that uses only end-to-end delay feedback to ensure QoS while using the network resources efficiently. In particular, we show that while for short paths (up to two hops long) the proposed scheme guarantees end-to-end delay budgets for all node pairs and also maximizes the total network throughput, when there are long paths in the network the resulting solution, even though still self-stabilizing and QoS-compliant, can deviate from the global network throughput. We present numerical results and conclude with a discussion of possible implementations of the proposed scheme in multi-service networks involving a mixture of best-effort and QoS-constrained services.

Index terms—end-to-end delay guarantee; potential function; Lyapunov function; non-linear network optimization

I. INTRODUCTION

With the wide availability of high bandwidth data networks, there is growing migration of all communication services, be they transactional (credit card), elastic (best effort data), streaming (off line audio or video), or real time (real-time voice, video) to the Internet Protocol (IP). Such a migration raises several concerns. Chief among these is lack of wide-spread mechanisms in standard data networks to support applications that require strict quality of service (QoS) guarantees. For example, successive packets in a real-time voice or video application need to be within a small fraction of a second of each other (~250ms for audio) [1]. Violations of such end-to-end latencies for a small percentage of the packets render the audio or video unintelligible or at best low quality.

Even though there are mature standards for QoS enforcement in data networks, e.g., ATM or MPLS, their underlying protocols are not widely used except perhaps in the core of carefully managed service provider networks. Part of the reason for their lack of wide-spread use is reliance on extensive and centralized controls and compared to the standard IP networks these protocols require relatively large state and messaging. This limits scalability and generates potentially unresolved peering issues between different operators that need to exchange such state. A typical data network today must, therefore, rely on the availability of large amounts of excess bandwidth to meet not only an application’s basic rate requirement, but also, implicitly, its packet loss, delay and jitter bounds [1].

An important development for (best effort) data applications in the Internet was the invention of scalable and decentralized protocols such as UDP and TCP. These mechanisms make efficient data transfer and loss recovery possible while preventing the network from overload collapse [2, 3]. As durable and robust as these protocols have proven to be, and as mature as their underlying theories are, e.g., see [4,5,6], these transport and networking protocols were neither designed for, nor are able to meet, explicitly, the multiplicity of QoS needed by emerging applications. It is not even yet clear that the multiplicity of QoS requirements of the emerging applications can all be met through decentralized control mechanisms like TCP that solely use end-to-end feedback. However, an interesting question is whether purely end-to-end mechanisms can, in principle, efficiently meet new QoS constraints, such as end-to-end delay. A pertinent development in this direction was the adaptive traffic engineering methodology [7,8]. But, as in other work dealing with such QoS issues as delay, see [9,10], the traffic engineering approach does not incorporate end-to-end constraints explicitly in the model, and neither do any of the flavors of TCP, see [9]. In [11] some control heuristics for delay-sensitive VoIP is proposed and simulated in the context of a possible future generation Internet to show that services with explicit end-to-end delays could be managed well using relatively light state. However, overall there has been a dearth of literature dealing directly with light-state Internet-like protocols for the emerging multi-QoS services.

In this paper we propose and investigate a new flow control mechanism that enables tight end-to-end QoS, such as delay in VoIP or real time video, while using the network bandwidth efficiently. By “efficiency” we mean optimal or near-optimal usage of the network capacity, at least in some asymptotic sense and under prescribed conditions that we will establish. Our approach follows the seminal framework of [7, 8] in the sense of casting the flow control problem as global...
decentralized network optimization subject to limited state in the form of end-to-end feedback.

II. THE GLOBAL OPTIMIZATION MODEL

To make a complete analysis possible, we make some simplifying assumptions that are commonly made in the literature but clearly are abstractions of reality. Specifically, we adopt a fluid modeling framework, where

1. The (total) flow through a link induces a (average) delay that is given by the convex function 1/(c-f), where c is the capacity of the link and f is the total flow through the link.

2. End-to-end delay for a flow is simply the sum of delays on each of the links on its path and is instantaneously experienced by the end packet of the flow.

3. The accumulated forward delay of a received packet at the destination is instantaneously transmitted to the flow source as feedback.

Assumption 1 invokes the average \( \text{M/M/1} \) queuing delay for each edge and is reasonable given our focus on delay as a key QoS metric. Assumptions 2 and 3 can be found in similar work, such as [4-11] where a fluid model with instantaneous feedback is used to quantify packet loss on a path and establish stability of the control scheme. Ref. [9] discusses the implications of packet-level queuing behavior on the performance and stability of TCP while using a fluid model. Let

\[
G = (V, E), \text{ the network and its set of vertices and edges} \\
\text{ } \text{ } c_e = \text{ capacity of the edge } e \\
\text{ } \text{ } P = \text{ set of all paths for all node }- \text{ pairs and } P = |P| \\
\text{ } \text{ } N = \text{ length of the longest path in } G \\
x_p = \text{ amount of flow on path } p \\
(x) = \text{ vector of } x_p \text{s or } (x_1, ... , x_p) \\
d_p = \text{ upper bound on the flow } x_p \text{ on path } p \\
\tau = \text{ end }- \text{ to }- \text{ end delay threshold}
\]

We are interested in quantifying the value of the following (global and centralized) optimization problem:

In P1, \( f_e(x) = \sum_{p \in \text{edge } p} x_p \) is the total flow through edge \( e \) and \( \Delta_p \) is the total end-to-end delay on path \( p \) for each path of the set \( \{x_p\} \). We shall refer to \( \tau - \Delta_p \) as the residual delay budget on path \( p \). In P1, the total network throughput is to be maximized subject to specified bound on the flow on each path and -- the key differentiator of this optimization model -- an end-to-end delay constraint (2) is imposed on each path only when the path is used, i.e., when \( x_p > 0 \). Observe that there may be multiple paths for the same node-pair, each with its own upper bound.

The purpose of the global optimization model is to provide a tight bound on the performance of any decentralized flow control mechanism that may be proposed. In particular, we wish to use the optimum value of P1 as a benchmark for the quality of our proposed flow control mechanism. Unfortunately, the combinatorial form of this optimization problem prevents us from a direct solution, e.g., via the Lagrangian saddle points and first order conditions. It is, of course, possible, in principle, to obtain the optimum value of P1 as follows. For each subset \( Q \) of paths in \( P \), solve the following modified optimization problem:

\[
P1(Q): v^*(Q) = \max_{x_p} \sum_{p \in \text{Q}} x_p \\
\text{such that:} \\
(1) \quad x_p \leq d_p \quad \forall p \in P \\
(2) \quad \Delta_p = \sum_{e \in p} \frac{1}{c_e - f_e(x)} \leq \tau \quad \forall p \in P \\
(3) \quad x_p \geq 0 \quad \forall p \in P.
\]

Unlike P1, P1(Q) is a standard convex optimization problem, since \( 1/(c_e - f_e) \) is a convex function of \( x_p \) for all delay constraints and condition (5) is enforced whether or not the path is used. Now \( Q^* = \arg \max v^*(Q) \) must be the same optimum as \( v^*(P) \). To see this, consider the optimal solution to P1 and let \( P^* \) be its set of paths with non-zero flows. By definition \( \Delta_p \leq \tau \) for all \( p \in P^* \), and thus by setting \( Q=P^* \), we get \( v(Q^*) \geq v(P^*) = v^*(P) \), as claimed. The downside of this approach is that since there are \( 2^{|P|} \) subsets in \( P \), this would require solution of \( 2^{|P|} \) convex optimization problems, a practical impossibility even when \( P \) consists of 10-20 paths.

Although we will not pursue a direct solution to P1, we establish a key property of an optimal solution to it for networks consisting of short paths. We will see in Section III that this result will establish the optimality of our flow control scheme in such a setting.

**Lemma 1.** When \( G \) consists of paths of 1 or 2 hops long, then the optimal solution \( x^* \) of P1 has the following property.

For each \( x^*_p, p \in P \), one of the following must hold: (1) If
Let us increase \( x_p^* \) by a small enough but positive amount \( \delta x_p^* \) and decrease \( x_1^* \) and \( x_2^* \) to \( x_1^* - \delta x_1^* \) and \( x_2^* - \delta x_2^* \), respectively so that all three constraints above remain intact, which is possible due to the continuity of \( \Delta_1, \Delta_2, \) and \( \Delta_p \).

That is, let \( x_p^* + \delta x_p^*, x_1^* - \delta x_1^* \) and \( x_2^* - \delta x_2^* \) be such that

\[
\Delta_1(x_1^* - \delta x_1^*, x_2^* + \delta x_2^*) = \Delta_2(x_2^* - \delta x_2^*, x_p^* + \delta x_p^*) = \tau
\]

and \( \Delta_p(x_1^* - \delta x_1^*, x_2^* - \delta x_2^*, x_p^* + \delta x_p^*) < \tau \). Expanding \( \Delta_1 \) and \( \Delta_2 \) to the first order around \( (x^*) \) and using (8)-(9) we get

\[
\begin{align*}
\frac{\partial \Delta_1}{\partial x_{p, x_1^*}} \delta x_p^* + \frac{\partial \Delta_1}{\partial x_{i, x_1^*}} \delta x_1^* & = 0, \quad i = 1, 2.
\end{align*}
\]

Define \( \alpha_i = 1/\tau(c_i - x_i - x_p) \) for \( i = 1, 2 \). Then (7)-(9) can be re-written more compactly as

\[
\begin{align*}
\alpha_1 + \alpha_2 & < 1, \quad 0 \leq \frac{1}{b_1 - x_1} = 1 - \alpha_1 \leq 1 \quad \text{and} \\
0 \leq \frac{1}{b_2 - x_2} = 1 - \alpha_2 \leq 1.
\end{align*}
\]

Taking partial derivatives of \( \Delta_1, \Delta_2, \) and \( \Delta_p \) in (8)-(9), substituting in (10) and leveraging (11), we get

\[
\alpha_i^2 \delta x_p^* = \frac{\partial \Delta}{\partial x_{i, x_1^*}} \delta x_1^* (\alpha_i^2 + (1 - \alpha_i)^2) \quad i = 1, 2
\]

Using (12), we can now quantify the impact on the total network throughput of the local variations on paths \( p, 1 \) and \( 2 \)

\[
\delta x_p^* - \delta x_1^* - \delta x_2^* = (1 - \sum_{i=1,2} \alpha_i^2 (1 - \alpha_i)^2) \delta x_p^*
\]

which is strictly positive when \( \alpha_1, \alpha_2 \geq 0 \) and \( \alpha_1 + \alpha_2 < 1 \), which are the conditions (11) and hold by assumption (see the Appendix for a detailed proof). This means a small increase in \( x_p^* \) can improve the solution, contradicting its optimality and thus completing the proof. □

III. FLOW CONTROL AS A GRADIENT SYSTEM

We wish to construct a decentralized flow control mechanism using minimal state. Since we are interested in

\[
x_p^* = d_p, \text{ then } \Delta_p \leq \tau, \text{ (2) If } 0 < x_p^* < d_p, \text{ then } \Delta_p = \tau \text{ and (3) if } x_p^* = 0 \text{ then } \Delta_p \geq \tau.
\]

Proof. Case (1) merely states the feasibility of \((x_p^*)\) for points at their upper bounds. We prove that if \( x_p^* < d_p \) then it is not possible to have \( \Delta_p < \tau \), which together with feasibility of \((x_p^*)\) proves cases (2)-(3) jointly. Assume otherwise. That is, suppose for some path \( p \) its flow is below its upper bound, \( x_p^* < d_p \), and also we have \( \Delta_p < \tau \). Let \( p \) consist of edges 1 and 2 with capacities \((c_1, c_2)\), as shown in Fig. 1. (The case of a single hop path \( p \) is a simpler variant of the same set of arguments that we present below which we will not replicate for brevity.) If \( p \) intersects no other paths, since \( x_p^* \) is not at its upper bound, then by continuity of \( \Delta_p = 1/(c_1 - x_p) + 1/(c_2 - x_p) \) \((< \tau)\) for small enough \( \delta 0 \), \( x_p^* \rightarrow x_p^* + \delta \) is still feasible and we have strictly increased the total throughput, contradicting the optimality of \((x_p^*)\). If \( p \) intersects only single hop paths, then if either of these two single-hop flows is at its delay budget, then at least on one of the two edges of \( p \) we have \( 1/(c_{1,2} - f_{1,2}) = \tau \), thus \( \Delta_p \geq \tau \), contradicting the assumption \( \Delta_p < \tau \). If neither of these two single-hop flows are at their upper bound then as in the previous case, there is a \( \delta 0 \) such that \( x_p^* + \delta \) is feasible, again, a contradiction. The only remaining case would be when path \( p \) intersects one or more two-hop paths as shown in Fig. 1. Clearly, many more paths with non-zero flows may be intersecting \( p \) (not shown) but we need only two such paths to derive a contradiction. The case of a single intersecting two-hop path follows a simpler argument than two paths that we shall discuss further.

\[
\begin{align*}
\Delta_1 &= 1/c_1 - x_1^* + 1/c_2 - x_2^* < \tau \\
\Delta_2 &= 1/b_1 - x_1 + 1/c_1 - x_1^* = \tau \\
\Delta_p &= 1/b_2 - x_2 + 1/c_2 - x_2^* = \tau
\end{align*}
\]

\[
\begin{align*}
(7) \quad \Delta_p &= \frac{1}{c_1 - x_1^* - x_p} + \frac{1}{c_2 - x_2 - x_p} < \tau \\
(8) \quad \Delta_1 &= \frac{1}{b_1 - x_1} + \frac{1}{c_1 - x_1^*} = \tau \\
(9) \quad \Delta_2 &= \frac{1}{b_2 - x_2} + \frac{1}{c_2 - x_2^*} = \tau
\end{align*}
\]
ensuring that end-to-end delays are met, it is fair to assume that each source is aware of the delay due its current level of flow and receives feedback on each path from the path destination continuously. Is it possible to construct a meaningful flow control mechanism that uses only this amount of feedback?

Consider the following adaptive flow control mechanism executed asynchronously by each source node of each path at each instant: 1) increase the flow on path if there is a positive residual delay budget, 2) decrease it when the residual delay budget is negative and 3) hold stationary when the flow is precisely at the budget. Even though this is a simple recipe, paths intersect and the delay budget for flows may become positive or negative due to the change of flows on other paths intersecting it, and we need to know if the system as a whole stabilizes, oscillates or experiences unpredictable fluctuations. Additionally, even if the system stabilizes, we need to know if the stable points use the network capacity efficiently. We set out to first establish the stability and second benchmark the stable points use the network capacity efficiently. We set

P2: Gradient System

\[
\begin{aligned}
\dot{x}_p &= \begin{cases} 
  k(\tau - \Delta_p) & \text{if } 0 < x_p < d_p \\
  k(\tau - \Delta_p)^+ & \text{if } x_p = 0 \\
  k(\tau - \Delta_p)^- & \text{if } x_p = d_p 
\end{cases} \\
\forall p \in P
\end{aligned}
\]

(14)

defined on the compact and convex set

\[ X = \{ x; 0 \leq x_p \leq d_p, \forall p \in P \} \subset R^p. \]

We say a point \( \bar{x} \) of P2 is invariant under P2 when one of the following conditions hold for every coordinate \( p \) of \( \bar{x} \):

\[
\begin{cases}
  \text{if } \bar{x}_p = 0 \text{ then } \Delta_p \geq \tau \\
  \text{else if } \bar{x}_p = d_p \text{ then } \Delta_p \leq \tau \\
  \text{else if } 0 < \bar{x}_p < d_p \text{ then } \Delta_p = \tau.
\end{cases}
\]

(15)

All other points of \( X \) are regular. Intuitively, an invariant point of P2 is "stuck" and cannot move in any direction under P2. (An invariant point is essentially a stable point of the gradient system P2 over \( X \), see the footnote on next page.) Consider the single-valued and infinitely differentiable function \( L(x) \)

\[
L(x) = \tau \left( \sum_{p \in P} x_p + \sum_{e \in E} \ln(c_e - f_e(x)) \right)
\]

defined on \( X \). Observe that \( L(x) \) is bounded on \( X \),

(16)

\[ E \ln(\tau^{-1}) \leq L(x) \leq \tau \left( \sum_{e \in E} c_e - \sum_{e \in E} \ln(c_e) \right), \forall x \in X \]

(and thus \( L(x) \) can be made strictly positive over \( X \) by the addition of a constant term). Observe also that P2 can be equivalently written as a gradient system \([12]\) with respect to \( L(x) \). That is,

\[
\dot{x}_p = \frac{\partial}{\partial x_p} L(x) \quad \forall p \in P
\]

(17)

To see this, notice first that

\[
\frac{\partial}{\partial x_p} \{ f_e(x) \} = \frac{\partial}{\partial x_p} \{ \sum_{r \in e} x_r \} = \begin{cases} 1 & \text{if } e \in p \\
 0 & \text{otherwise} \end{cases} \equiv \delta_{ep}
\]

Thus,

\[
\frac{\partial}{\partial x_p} L(x) = \frac{\partial}{\partial x_p} \left( \sum_{e \in E} \ln(c_e - f_e(x)) \right) = \tau + \sum_{e \in E} \frac{\partial}{\partial x_p} \ln(c_e - f_e(x))
\]

(18)

\[ = \tau + \sum_{e \in E} \frac{(c_e - f_e(x) - c_e + f_e(x))}{(c_e - f_e(x))} \]

and using (18)

\[
= \tau + \sum_{e \in E} (- \delta_{ep}) / (c_e - f_e(x))
\]

and therefore

\[
\frac{\partial}{\partial x_p} L(x) = \tau - \sum_{e \in E} 1/(c_e - f_e(x)) = \tau - \Delta_p
\]

(19)

In addition,

\[
\dot{L}(x) = \sum_{p \in P} \frac{\partial}{\partial x_p} L(x) \dot{x}_p = \sum_{p \in P} (\tau - \Delta_p) \dot{x}_p
\]

(20)

Substituting in (20) for time derivative of \( x_p \) from (17)

\[
\dot{L}(x) = \sum_{p \in P} k(\tau - \Delta_p)^2 + \sum_{x_p = 0} \sum_{x_p = d_p} k(\tau - \Delta_p)^+ (\tau - \Delta_p)^- + \sum_{x_p = 0} \sum_{x_p = d_p} k(\tau - \Delta_p)^- (\tau - \Delta_p)^- 
\]

(21)

as each sum is strictly positive at regular points of \( X \) under P2 and zero at its invariant points. Thus \( L(x) \) is strictly increasing on trajectories of P2 that start at regular points and therefore there are no closed orbits. Together with the boundedness of \( L(x) \) and positivity of \( \dot{L}(x) \), from (16) & (21), it follows that

1) \( L(x) \) is a Lyapunov function for P2 and thus every trajectory converges (possibly asymptotically) to an invariant point of P2 and
2) the invariant points of P2 are precisely the local maxima of \( L(x) \) in \( X \). But \( L(x) \) is strictly concave

\[ 2 \text{In a standard gradient system (GS) the Lyapunov function } L(x) \text{ is defined on } R^p \text{ whereas P2 is defined on a sub-set } X \subset R^p. \text{ However, since all trajectories of (14) remain within } X, \text{ we can replace the notions of critical points of } L(x) \text{ and the (asymptotically) stable points of GS with their natural counterparts in a compact domain, namely local extrema of } L(x) \text{ and the (asymptotically) invariant points of the GS. GSs over bounded sets are discussed in [17].} \]
over $X$ as seen by taking partial derivative of $L(x)$ once more in (19) for all $p$

\[
\frac{\partial^2}{\partial x_p^2}L(x) = \frac{\partial}{\partial x_p}(\tau - \Delta_p) = \frac{\partial}{\partial x_p}(\tau - \sum_{c \in p} \frac{1}{c} - f_c(x))
\]
\[
= \sum_{c \in p} \frac{\partial}{\partial x_p} \left( \frac{1}{c} - f_c(x) \right)
\]
\[
= \sum_{c \in p} \frac{1}{(c - f_c(x))^2} < 0 \quad \forall p \in P \quad \forall x \in X.
\]

Hence, there is only a single local maximum of $L(x)$ over any convex and compact domain, such as $X$, and consequently this point must coincide with its global maximum $\bar{x}$ in $X$. Putting all of the above together, we have:

**Theorem 1.** P2 has a unique and (asymptotically) invariant point $\bar{x}$ in $X$ which is the same point as the global maximum $\bar{x}$ of $L(x)$ in $X$, where all trajectories of P2 originating in $X$ terminate.

Lastly, notice that for networks with short paths ($N=1, 2$)

**Lemma 1** shows that an optimal solution $x^*$ of P1 is precisely an invariant point of P2. Thus, it must coincide with the unique and asymptotically invariant point $\bar{x}$. From these two observations we also conclude that:

**Theorem 2.** For networks with 1-2 hop paths, the dynamical system prescribed by P2 has a unique invariant point in $X$ which also globally maximizes throughput. In other words, starting from any point in $X$, the decentralized feedback control system defined by (14) follows a trajectory that converges to the globally optimal point of P1.

**IV. THROUGHPUT BOUNDS FOR NETWORKS WITH LONG PATHS**

How bad are solutions of P2 as proxies for P1 for networks with long paths? Fig. 2 shows that when paths are longer than two (hops), the optimal solution to P1 need not be an invariant point of P2. Here the flow $y$ has a positive residual delay budget when $x_1, x_2 = 1$ and yet it is equal to zero in the optimal solution of P1.

In this example the optimal solution $x^*$ to P1 is $(1, 1, 0)$ with total throughput 2, whereas $\bar{x}^*$, the asymptotically invariant point of P2, is obtained from

\[
\dot{x}_i = 1 - 2/(1 + y_{i-1} - y_{i-2} - x_i - y_i) = 0 \quad i = 1, 2
\]
\[
\dot{y} = 1 - 1/(1 + 1 - x_i - y_i) = 0
\]

which is seen to be $\approx (0.57, 0.57, 0.7)$ with total throughput equal to $\approx 1.84$. Thus, in this example, the solution of P2 is $\approx 8\% = (2-1.84)/2$ lower than the optimal solution to P1. Examples with progressively larger deviation between the throughput of P1 and P2 can be constructed by picking arbitrarily long paths with tight flows (similar to $x_i$ or $x_2$ in Fig. 2) where a single transversal path (similar to $y$ in Fig. 2) with positive residual delay budget (i.e., $\tau - \Delta_p > 0$) intersects each path once. Let us call this special network a transversal system of paths. We show that even though long paths can cause large divergence between solutions of P1 and P2 in terms of the total throughput for the same network, the divergence turns out to be bounded. We quantify this bound via a worst case analysis for a transversal system of paths (Lemma 2) and then extend to arbitrary networks (Theorem 3). Proofs are elementary and are left out.

**Lemma 2.** For a transversal system of paths with maximum path length $N$, $v(P2)/v(P1) \geq 1/[\sqrt{N}]$ where $v(P1)$ and $v(P2)$ are, respectively, the total throughputs corresponding to the solutions to P1 and P2 models for the transversal system, and $[r]$ is the closest integer to $r$.

**Theorem 3.** As path lengths (in terms of the number of hops) increase beyond 2, there is progressively a larger penalty in terms of total throughput when the solution of P2 is used as a proxy for P1. However, the penalty is bounded above by $1/[\sqrt{N}]$ where $N$ is the maximum length of a path. In other words, the solution of P2 in place of the solution to P1 results in a throughput value no worse than $v(P1)/[\sqrt{N}]$.

**V. NUMERICAL EXAMPLE**

Fig. 4(a) shows a sample network with 10 nodes, 16 edges and 13 paths. Paths 6, 9 and 3 are 3, 3 and 4 hops long, respectively, and all others are 1 and 2 hops. Edge capacities are 0.75, 1 and 2 units and end-to-end delay is 5 units of time for all node-pairs. The two sets of trajectories of P2 for the initial conditions (0) and (0.05, 0.1, 0.0, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5, 0.55, 0.6, 0.65) are shown in Fig 4(b)(c).

![Diagram](image-url)
The same final solution (0.438, 0.251, 0, 0.187, 0.937, 0, 0.60, 0.751, 0.710, 0.550, 0.502, 0.689, 0.8) for both initial points from the non-zero initial point. In both cases convergence of flows on shorter paths or paths with ample capacity typically results in the total throughput of 6.416 units. It is not hard to see that this solution is actually optimal for P1 even though there are 3 and 4 hop paths. Convergence to the steady-state (optimal) solution is seen to be slightly faster from (0, 0, 0) than from the non-zero initial point. In both cases convergence of the 3-hop flows id 6 and id 9 is slower than other flows. Observe that the longest path (id 3) does increase initially for initial condition \((0, 0, 0)\) but is stifled later by flows on shorter paths. This phenomenon is actually common: Pressure from flows on shorter paths or paths with ample capacity typically eliminates long ones and one can expect the solution to P2 not to be far from P1 in practice.

VI. CONCLUSIONS

We introduced a new distributed flow control mechanism that uses only end-to-end feedback for applications requiring end-to-end delay guarantees over data networks. We showed, by way of a gradient dynamical system, that the proposed flow control mechanism is stable and always converges to the same feasible point. Further, we showed that when the network consists of short paths, the stable point of the feedback system also maximizes the total network throughput. When the network contains long paths, the throughput at the stable point may be smaller than the maximal network throughput that can be derived from a centralized solution by a factor roughly equal to inverse of square root of the maximal path. This is the price paid for use of only end-to-end feedback to solve the global throughput optimization problem with tight end-to-end bounds in the context of a distributed flow control.

It is not hard to combine the proposed flow control mechanism with existing data networking protocols. A simple protocol would use end-to-end delays (e.g., \(\frac{1}{2}\) of the ICMP ping values) repeatedly measured at the source of each path and adjust the delay-sensitive flows based on this feedback. For example, for VoIP services, the adaptive changes to the flows should be interpreted in terms of call admission policy rather than immediate action on existing calls, e.g., dropping an existing call. For further details of such a solution, see [11].

REFERENCES


Figure 4. (a) An example with capacities marked on links and path ids marked above them. The (same) stable point is obtained starting from initial condition (b) \(x(i)=0\) for all \(i\) and (c) \(x(i)\neq 0\).