Optimal Scheduling and Routing for Maximum Network Throughput

Optimal Scheduling and Routing for Maximum Network Throughput

Emilio Leonardi, *Member, IEEE*, Marco Mellia, *Member, IEEE*, Marco Ajmone Marsan, *Fellow, IEEE*, and Fabio Neri, *Member, IEEE*

***Abstract*—In this paper we consider packet networks loaded by admissible traffic patterns, i.e., by traffic patterns that, if optimally routed, do not overload network resources. We prove that simple distributed dynamic routing and scheduling algorithms based upon link state information can achieve the same network throughput as optimal centralized routing and scheduling algorithms with complete traffic information.**

Our proofs apply the stochastic Lyapunov function methodology to a *flow-level* abstract model of the network, and consider *elastic* traffic, i.e., we assume that flows can adapt their transmission rates to network conditions, thus resembling traffic engineering and quality-of-service approaches being currently proposed for IP networks.

Although the paper mainly brings a theoretical contribution, such dynamic routing and scheduling algorithms can be implemented in a distributed way. Moreover we prove that maximum throughput is achieved also in case of temporary mismatches between the actual links state and the link state information used by the routing algorithm. This is a particularly relevant aspect, since any distributed implementation of a routing algorithm requires a periodic exchange of link state information among nodes, and this implies delays, and thus time periods in which the current link costs are not known.

***Index Terms*—Asymptotic stability, computer network performance, Lyapunov methods.**

I. Motivation and Previous Work

In RECENT years, dynamic routing algorithms have attracted the attention of the networking community. Several aspects have been investigated, such as: 1) protocol convergence [1], [2]; 2) overhead impact [3], [4]; 3) implementation issues [5]; 4) impact of update policies [3]; and 5) performance issues.

The last topic is of particular interest, and has often been associated with QoS routing. Indeed, the core of any routing algorithm is a state-dependent cost function that is used to find the optimal (or at least a good) route across the network by solving an optimization problem. In particular, given the best-effort nature of the current Internet, where the majority of data flows are elastic (i.e., they can adapt to network conditions), the most commonly used optimization metric aims either at the maximization of user throughput or at the minimization of network utilization. Several QoS routing proposals introduced dynamic algorithms and protocols that provide advantages over traditional, topology-based algorithms, such as the shortest path routing presently used in TCP/IP networks [6]–[8], [10]–[13].

Within QoS routing studies, two main research areas can be identified. The first area, sometimes also called "traffic engineering," considers as input to the design of routing algorithms the information about the traffic pattern that users offer to the network. We call this class of algorithms "traffic-aware" QoS routing algorithms. Given the knowledge of the network topology, and a utility function, the goal of the algorithm is to find a set of routes that maximizes the utility function, under technological or performance constraints. Traffic-aware QoS routing algorithms can be formalized as linear programming (LP) problems, and efficient solutions are available since the early seventies [6]. The result of the optimization is, for each traffic relation, a set of paths and a corresponding set of probabilities that specifies the traffic splitting over each path. The implementation of such a routing algorithm in the Internet can be obtained either by using explicit path mechanisms, such as those offered by MPLS [14], or by exploiting the recent results in [7], [8], which show that with current IP routing constraints it is possible to find: 1) an equivalent set of link costs that implements the same routes as those of an explicit path mechanism; and 2) a set of rules to implement the same traffic splitting. The difficulties to quickly detect instantaneous traffic pattern variations [9], and to solve in a on-line distributed fashion the LP problems, makes these algorithms better suited for a static centralized implementation. For this reason, in the following we refer to them as *static routing* algorithms.

The second area within QoS routing research assumes that the traffic pattern is not known. We call the corresponding class of algorithms "traffic-unaware" QoS routing. In this case, the optimization of routing is based on direct measurements of the performance of network elements, like link utilization, queue lengths, etc. Some exchange of limited state information between network nodes is often required. Given the current state of the network, the goal of a traffic-unaware QoS routing algorithm is to find a set of paths that can be used to accommodate traffic requests. Among the proposed heuristics, the "Widest-Shortest" [10] and the "Minimum-Distance" [11] approaches are generally considered to be good routing algorithms. These findings have been generalized by recent results that consider resilience to traffic load variations [12], and more general cost functions [13]. The characteristics of traffic-unaware routing algorithms

Manuscript received August 23, 2005; revised July 18, 2006; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor A. Fumagalli. This work was supported in part by the European FP6 Network of Excellence EURO-NGI. A previous version of this paper appeared in the Proceedings of IEEE INFOCOM 2005.

The authors are with the Dipartimento di Elettronica, Politecnico di Torino, 24 Torino, Italy (e-mail: leonardi@tlc.polito.it; mellia@tlc.polito.it; ajmone@tlc.polito.it; neri@tlc.polito.it).

Digital Object Identifier 10.1109/TNET.2007.896486

favor a distributed implementation in which routes are dynamically updated to follow the instantaneous traffic variations and network state. For this reason, in the following we refer to them as *dynamic routing* algorithms.

The performance evaluation of dynamic QoS routing algorithms was traditionally carried out by simulation only. Thus, to the best of our knowledge, no analytical results are available to characterize their properties, and in particular to define the network stability region when these algorithms are used.

The recent introduction of powerful theoretical approaches in network studies, such as the stochastic Lyapunov function methodology [15], opened new perspectives to theoretical studies on the impact of routing algorithms. Using this approach, for example, the authors of [16] considered ad hoc wireless networks, and defined a packet-by-packet dynamic routing algorithm and a power allocation scheme, whose joint use is shown to maximize the network stability region.

In this paper, we consider a more general and abstract flow-level network model, that allows us on one side to consider the intrinsic elasticity of Internet traffic, and on the other side to almost completely neglect the packet-level dynamics, thus greatly reducing the complexity of the system under study. Using this model, we analyze the stability properties of different classes of dynamic routing algorithms and bandwidth allocation policies.

Our goal is to provide a theoretical support to the performance evaluation of routing schemes under dynamic traffic. We, therefore, focus on distributed traffic-unaware routing schemes rather than on traffic-aware approaches, and we study whether the former can guarantee the same performance of centralized solutions operating on a perfect knowledge of the network traffic.

More in details, we consider packet networks loaded by admissible traffic patterns, i.e., traffic patterns that, if optimally routed, do not overload network resources. Under these conditions, we prove that simple distributed dynamic routing and scheduling algorithms using link state information can achieve the same network throughput as optimal centralized routing and scheduling algorithms with complete traffic information.

We also show that such dynamic routing and scheduling algorithms can be implemented in a distributed way, and we prove that maximum throughput is achieved also in case of temporary mismatches (possibly due to delays in exchanging state information with proper signalling schemes) between the actual link costs and those used by the routing algorithm. This is a particularly relevant aspect, since any distributed implementation of a routing algorithm requires a periodic exchange of link state information among nodes, and this implies delays, and thus time periods in which the updated state of links is not known, and the optimal costs cannot thus be computed.

We remark the three main advantages of our flow-level model with respect to previously considered packet-level models (e.g., in [16]). First, optimal dynamic routing strategies defined in our flow-level model are flow oriented and implement route pinning, so that they guarantee in-sequence delivery of packets at the destination. On the contrary, each packet is routed at every router according to some instantaneous local congestion index by the optimal strategies in [16], thereby possibly causing out-of-sequence deliveries at the destination. Moreover, flow-oriented routing schemes can be easily integrated with flow admission mechanisms, so as to meet traffic QoS constraints. This is one of the reasons why most of the recently defined QoS routing schemes operate at the flow level (implementing route pinning); see [10]–[13]. Second, in packet-level models, analytical results can be obtained usually only under the assumption that packets are generated by sources according to a memoryless process; in flow-level models just flows arrivals must follow a memoryless process. While the former assumption appears rather unrealistic, the latter is much more acceptable (many LRD traffic models also assume that flows are generated according to a memoryless process). Third, flow-level models permit to consider effects related to the elasticity of traffic, i.e., to the fact that flows duration are related to the bandwidth that flows obtain in the network.

Even if our contribution is mostly theoretical, some of the proposed dynamic routing and scheduling algorithms are compatible with a practical implementation in a real network scenario.

The rest of this paper is organized as follows. Section II presents our modeling assumptions, as well as the notation that is used throughout the paper. Section III introduces the traffic admissibility and sustainability definitions, and recalls the stability criterion based on the stochastic Lyapunov function that is used in later sections. Section IV presents preliminary results, relating to an idealized scenario, where link utilization are overestimated. The system state is defined in terms of flows backlog, i.e., of the remaining amount of information to be transfered in the network. In this section we prove that the so-called min-backlogged-path routing algorithm can achieve the same network throughput as an optimal static routing algorithm under any admissible traffic pattern. The proof is also extended to the case of a finite error in the link cost estimation. Section V presents our main results, proving that, under any admissible traffic pattern, a network adopting the min-backlogged-flow routing algorithm, similar to the min-backlogged-path routing, in conjunction with rate control at the source and a feasible scheduling policy, called max-scalar policy, can achieve the same network throughput as an optimal static routing algorithm. The system state is always defined in terms of information backlog, and rate control at the source avoids buffering along flow paths. Some variations and generalizations of the basic routing and scheduling schemes are also discussed. Section VI instead defines the system state in terms of number of active flows, and particularizes the result to the case of exponentially distributed flow sizes, proving throughput maximization for significant simplifications of previous routing and scheduling schemes. Section VII extends previous results when the amount of information buffered at network nodes is not negligible. Finally, Section VIII concludes the paper. Finally, Section VIII concludes the paper, with some discussion of implementation issues.

II. Modeling Assumptions and Notation

To simplify the notation we consider discrete-time systems, even if all results can be easily generalized to continuous-time systems. The slot duration is assumed to be equal to 1. The discrete-time steps can be seen as corresponding to equally spaced

snapshots of the system state; the scheduling and routing decisions are updated in correspondence with these snapshots.[1]

A network is represented by a directed graph $G = (V, E)$ where V is the set of vertexes (which represent the network switching nodes), and E is the set of edges (which represent the network transmission links). $C^l = C^{l(i\to j)}, l \in E, i, j \in V$ denotes the capacity of link l, that connects node i to node j. To simplify notation, in the remainder of the paper we assume all links to be of the same capacity: $C^l = C = 1, \forall l \in E$. However, this assumption can be easily relaxed, as we will show at the end of the paper in a special case.

The traffic in the network is modeled at the level of *flows*, neglecting individual packets. Elastic flows arrive at the network according to a Poisson process; each flow is associated with a source $s \in V$ and a destination $d \in V$, as well as a size, that represents the amount of information (in bits, bytes, or any other unit—we will refer to bytes without loss of generality) that has to be transferred. Flow sizes are assumed to be independent random variables, distributed according to a general distribution with mean $1/\mu$, and finite polynomial moments.

Each flow is routed through the network according to a given algorithm, and it remains active until its last bit is successfully delivered to the destination. During its active period, each flow receives a portion of the network capacity, which varies with the network congestion (for example, with the number of concurrent flows), i.e., flow sources can adapt their sending rate to network congestion; the link capacity is allocated to flows according to a scheduling policy, which may derive either from the end-to-end congestion control mechanism implemented at source nodes, like with the TCP congestion control, or from the network nodes, like with GPS scheduling policies [20]. Propagation delays (i.e., signals latencies) are neglected in this paper.

Each link l is modeled as a flow-level queue. The queue $q^l = q^{l(i\to j)}$, $l \in E$, $i, j \in V$ corresponds to link l, that connects node i to node j.

Let $\lambda_{\rm sd}$ be the average flow arrival rate for each source-destination pair; $\rho_{\rm sd} = \lambda_{\rm sd}/\mu$ is the average workload associated with flows from s to d. The traffic in the network is defined by the matrix $\boldsymbol{\rho} = [\rho_{\rm sd}]$, that is called the traffic matrix. $a_{\rm sd}(n)$ denotes the new amount of work arrived during time slot n due to flows with origin s and destination d. We note that $\rho_{\rm sd} = E[a_{\rm sd}]$.

As soon as flow f from s to d arrives at the network, it is routed over a path, call it π, which comprises several links. All the information associated with flow f are transferred over path π for the whole flow duration.

Let $x^l(n)$ represent the virtual backlog at link l; i.e., $x^l(n)$ represents the total amount of information in bytes scheduled for link l. Note that the virtual backlog at time n does not necessarily equal the physical backlog, i.e., the amount of information physically enqueued at queue l at time n, since some information may be physically queued at previous links along the path or it may be still at the source. More precisely, $x^l(n)$ is the sum three contributions: 1) the amount of information (bytes) physically enqueued at q^l; 2) the amount of information displaced in in upstream queues belonging to active flows that have been routed through q^l; and 3) the amount of information possibly still waiting for transmission at the source and belonging to the active flows that have been routed through q^l.

Let[2] $\mathbf{X}(n) \in \mathbb{N}^{|E|}$ be an $|E|$-dimensional vector,[3] whose components are the $x^l(n)$.

The routing algorithm is used to associate a path with each new flow that arrives to the network. A **static** (traffic-aware) routing algorithm is defined by matrix $\hat{\mathbf{R}} \in \mathbb{R}_+^{|V|^2\times|E|}$, whose element $\hat{r}^l_{\rm sd}$ is the percentage of flows from source s to destination d that, according to routing algorithm $\hat{\mathbf{R}}$, is routed through link l. Since the routing algorithm is traffic-aware, $\hat{\mathbf{R}}$ is a function of the traffic matrix $\boldsymbol{\rho}$, $\hat{\mathbf{R}} = \hat{\mathbf{R}}(\boldsymbol{\rho})$

Given any source-destination pair, let $\Pi(sd)$ be the set of possible paths from source s to destination d, and $\Pi = \bigcup_{s,d}\Pi(sd)$ be the set of all possible paths. Let $L(\pi)$ be the set of links (i.e., queues) traversed by path $\pi \in \Pi$. Let $\Pi(l)$ be the set of paths (from any source to any destination) traversing link l. An equivalent representation of the routing algorithm $\hat{\mathbf{R}}$ can be given by assigning the set of values $\hat{r}^\pi_{\rm sd}$ that represent[4] the percentage of $s \to d$ flows that, according to routing algorithm $\hat{\mathbf{R}}$, is routed through path $\pi \in \Pi(sd)$. We define accordingly the path routing matrix $\hat{\mathbf{R}}^\pi \in \mathbb{R}_+^{|V|^2\times|\Pi|}$.

Notice that it is always possible to decompose the traffic routed over link l as the sum of the traffic routed over paths π that use link l; indeed

$$\hat{r}^l_{\rm sd} = \sum_{\pi\in\Pi(l)\cap\Pi(sd)} \hat{r}^\pi_{\rm sd}. \tag{1}$$

A **dynamic** (traffic-unaware) routing algorithm is defined by a sequence of matrices $\mathbf{R}(n)$. The element $r^l_{\rm sd}(n)$ is the percentage of flows from s to d that, according to the dynamic routing algorithm, is routed through link l at time n. We consider dynamic routing algorithms in which $\mathbf{R}(n)$ is a function of the network state $\mathbf{X}(n)$, $\mathbf{R}(n) = \mathbf{R}(\mathbf{X}(n))$. An equivalent representation of the routing algorithm $\mathbf{R}$ can be given with the path routing matrix $\mathbf{R}^\pi(n)$, whose elements $r^\pi_{\rm sd}(n)$ represent the fraction of flows from s to d routed over path π at time n.

III. Traffic Admissibility and Sustainability

We say that a traffic matrix $\boldsymbol{\rho}$ is admissible (and the corresponding traffic pattern is admissible) if there exists a static routing algorithm $\hat{\mathbf{R}}$ under which all the queues (links) in the network are not overloaded. More formally:

Definition 1: A traffic pattern is *admissible* if there exists a *static* routing algorithm $\hat{\mathbf{R}}$, such that

$$\sum_{\rm sd} \hat{r}^l_{\rm sd}\rho_{\rm sd} < 1 \quad \forall l \in E. \tag{2}$$

[1]The slot duration should be short compared to the flow duration, but sufficiently long to neglect fast packet dynamics, e.g., in the order of a round-trip-time.

[2]We denote with $\mathbb{N}$ the set of nonnegative integers, with $\mathbb{R}$ the set of real numbers and with $\mathbb{R}_+$ the set of nonnegative real numbers.

[3]We use column vectors throughout the paper.

[4]We identify $|V|^2$ flows, each labeled by its source/destination nodes s, d; the notation $\hat{r}^\pi_{\rm sd}$ is redundant, since path π uniquely identifies the source and destination nodes s and d. When not strictly required, we will omit to explicitly indicate sd.

The problem of finding the static routing algorithm $\hat{\mathbf{R}}$ can be formalized as a LP problem. Let $f_{\mathrm{sd}}^{l(i\to j)} = \hat{r}_{\mathrm{sd}}^{l}\rho_{\mathrm{sd}}$ be variables that represent the average normalized amount of information from s to d that is routed through link l from i to j. Variables $f_{\mathrm{sd}}^{l(i\to j)}$ must satisfy

$$\sum_{j} f_{\mathrm{sd}}^{l(s\to j)} = \rho_{\mathrm{sd}} \quad \forall s,d \tag{3}$$

$$\sum_{j} f_{\mathrm{sd}}^{l(i\to j)} - \sum_{j} f_{\mathrm{sd}}^{l(j\to i)} = 0 \quad \forall i, i\neq s, i\neq d \tag{4}$$

$$\sum_{i} f_{\mathrm{sd}}^{l(i\to d)} = \rho_{\mathrm{sd}} \quad \forall s,d \tag{5}$$

$$\sum_{\mathrm{sd}} f_{\mathrm{sd}}^{l(i\to j)} < 1 \quad \forall l \tag{6}$$

$$f_{\mathrm{sd}}^{l(i\to j)} \geq 0 \quad \forall l,s,d. \tag{7}$$

Equations (3)–(5) represent the classic flow conservation equations; (6) forces all the flows routed over a link to be admissible, and finally (7) forces all the flows to be nonnegative. By selecting an appropriate utility function, the above expressions can be mapped into an optimization problem that defines the *optimal* routing under the chosen utility function. This class of routings belongs to the static class [6]–[8], and it assumes the knowledge of the traffic matrix $\boldsymbol{\rho}$.

Given an admissible traffic pattern, a system of queues is *stable* if all average virtual backlogs are bounded. i.e.:

Definition 2: A network of queues is *stable* if

$$\limsup_{n\to\infty} E\left[||\mathbf{X}(n)||\right] < \infty$$

where $||\cdot||$ is any norm function.

Now we introduce the important definition of traffic pattern sustainability. Sustainability depends on the scheduling policy, i.e., on the way in which the different flows queued at the same queue are served. More formally:

Definition 3: A traffic pattern $\boldsymbol{\rho}$ is *sustainable* if there exist a static routing algorithm $\hat{\mathbf{R}}$ and a scheduling policy such that the network of queues is stable.

The main difference between admissibility and sustainability is the fact that the former only guarantees that no network elements are overloaded, while the latter implies that all the traffic is effectively transferred through the network with a finite average delay.

It is immediate to verify that admissibility is a necessary condition for a traffic pattern to be sustainable; the reverse statement is, instead, less evident. It was indeed proved that in a network of queues under work-conserving scheduling policies instabilities may arise even in presence of admissible traffic. See, for example, [21].

We will show in the next sections that, under our assumptions, and with scheduling policies related to link loads, the two definitions of admissibility and sustainability are equivalent.

A. Lyapunov Stability Criterion

The stochastic Lyapunov function is a powerful tool to prove stability (i.e., positive recurrency) of Markovian systems. In this subsection we briefly report one of the main results related to the Lyapunov function methodology, which will be used in the remainder of this paper; we refer the interested reader to [15] for more details.

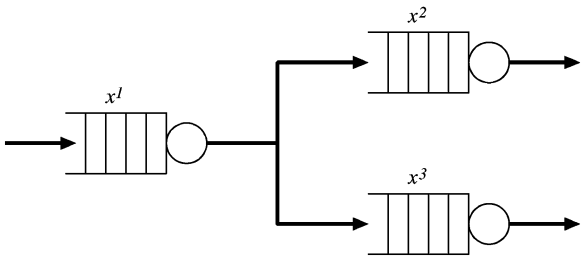

Fig. 1. Simple network topology in which some links cannot be fully loaded.

Theorem 1: Let $\mathbf{X}(n)$ be a $|E|$-dimensional Markov chain, whose elements $x^l(n)$, $l = 1,2,\ldots,|E|$ are nonnegative integers, i.e., $\mathbf{X}(n) \in \mathbb{N}_+^{|E|}$. If there exists a nonnegative valued function $\{\mathcal{L} : \mathbb{N}_+^{|E|} \to \mathbb{R}_+\}$ such that:

$$E\left[\mathcal{L}\left(\mathbf{X}(n+1)\right) - \mathcal{L}\left(\mathbf{X}(n)\right)|\mathbf{X}(n)\right] < \infty \tag{8}$$

$$\limsup_{||\mathbf{X}(n)||\to\infty} \frac{E\left[\mathcal{L}\left(\mathbf{X}(n+1)\right) - \mathcal{L}\left(\mathbf{X}(n)\right)|\mathbf{X}(n)\right]}{||\mathbf{X}(n)||} < -\epsilon \tag{9}$$

for some $\epsilon > 0$, then $\mathbf{X}(n)$ is positive recurrent (i.e., stable) and

$$\limsup_{n\to\infty} E\left[||\mathbf{X}(n)||\right] < \infty.$$

Inequality (8) imposes that the increments of the Lyapunov function $\mathcal{L}(\mathbf{X})$ be finite. It is immediate to verify that this constraint can be met in general if all the moments of $\mathbf{X}(n+1) - \mathbf{X}(n)$ are finite; in particular, for quadratic Lyapunov functions, it is sufficient that the second moment of $\mathbf{X}(n+1) - \mathbf{X}(n)$ is finite. The second inequality, (9), often termed as the Lyapunov function drift, requires that, for large values of $||\mathbf{X}||$, the relative average increment in the Lyapunov function from time n to time $n+1$ be negative. The intuition behind this result is that the system must be such that a negative feedback exists, which is able to pull it toward the empty state, thus making it ergodic.

IV. Results for an Optimistic Scenario

The results presented in this section are preliminary, since they refer to an ideal, optimistic scenario, in which we assume that the capacity of every link $l \in E$ is fully utilized, as long as the link virtual backlog $x^l(n)$ is non-null.

This is not true in general, as it can be understood by considering the simple example shown in Fig. 1. Remember that we are considering a flow-level model, and that the link virtual backlog $x^l(n)$ corresponds to the total amount of information associated with active flows already scheduled for link l. Assume that the three links in the figure have the same capacity and that the virtual backlogs are non-null. We immediately understand that link 1 is the system bottleneck: if link 1 works at its capacity, it cannot feed enough data into links 2 and 3 to keep them fully utilized.

Nevertheless, with our optimistic assumption (which will be corrected in the next section) the dynamics of the virtual backlog on link l is given by the following equation:

$$x^l(n+1) = \max\left(x^l(n) - 1, 0\right) + \sum_{\text{sd}} r^l_{\text{sd}}(n) a_{\text{sd}}(n)$$

where we assume a dynamic routing algorithm.

Using a vectorial notation we can write

$$\mathbf{X}(n+1) = \max\left(\mathbf{X}(n) - \mathbb{1}, \mathbf{0}\right) + \mathbf{R}^T(n)\mathbf{A}(n) \quad (10)$$

where $\mathbf{X}(n)$ is the vector of virtual backlogs; $\mathbb{1}$ is an $|E|$-dimensional vector with all entries equal to 1; $\mathbf{A}(n) \in \mathbb{R}_+^{|V|^2}$ is the vector whose components represent the workloads provided by flows arrived during time slot n; i.e., $\mathbf{A}(n) = [a_{\text{sd}}(n)]$; scalar operators are extended to vectors by applying the operator on all components. For example, given any two vectors $\mathbf{V}_1$ e $\mathbf{V}_2$, the vector $\max(\mathbf{V}_1, \mathbf{V}_2)$ contains components $\max(v_1^l, v_2^l)$.

Equation (10) implies that the system dynamics are described by an irreducible Markov chain whose state descriptor is $\mathbf{X}$.

Under these idealized conditions, it is immediate to realize that the sustainability and admissibility conditions are equivalent. Indeed, any scheduling policy capable of guaranteeing to flows service rates equal to or larger than the average virtual backlog, i.e., to the average amount of traffic routed over the different links, ensures bounded queues for any admissible traffic.

The question we raise now is the following: "Is it possible to devise a dynamic routing algorithm that can guarantee stability for any admissible traffic pattern, e.g., without requiring the knowledge of the traffic matrix $\boldsymbol{\rho}$?"

Our answer is positive; the following theorem defines a simple dynamic routing algorithm that guarantees stability under any admissible traffic pattern. We call this algorithm *min-backlogged-path routing* since newly arrived flows $s \to d$ that arrive during time slot n are routed according to a minimum cost path strategy in which link costs equal the link virtual backlogs. That is, the algorithm selects the path $\pi \in \Pi(sd)$ that minimizes the sum of virtual backlogs, i.e.,

$$\pi = \arg \min_{\pi \in \Pi(sd)} \sum_{l \in L(\pi)} x^l(n).$$

Theorem 2: A dynamic routing algorithm $\mathbf{R}^*(n)$ defined as

$$\mathbf{R}^*(n) = \arg\min_{\mathbf{R}} \mathbf{A}^T(n)\mathbf{R}\mathbf{X}(n) \quad (11)$$

stabilizes the network under any admissible traffic pattern.

Proof: We will prove this result by applying the Lyapunov function methodology. Let us consider the quadratic Lyapunov function

$$\mathcal{L}(\mathbf{X}) = \mathbf{X}^T\mathbf{X}. \quad (12)$$

Recalling that we consider Poisson arrivals of flows, and finite variance flow sizes, Inequality (8) holds. Then, consider the Lyapunov function drift (9)

$$\frac{E\left[\mathcal{L}\left(\mathbf{X}(n+1)\right) - \mathcal{L}\left(\mathbf{X}(n)\right)|\mathbf{X}(n)\right]}{||\mathbf{X}(n)||} = \frac{E\left[\mathbf{X}^T(n+1)\mathbf{X}(n+1) - \mathbf{X}^T(n)\mathbf{X}(n)|\mathbf{X}(n)\right]}{||\mathbf{X}(n)||}$$

$$= \frac{E\left[\max\left(\mathbf{X}(n) - \mathbb{1}, \mathbf{0}\right)^T \max\left(\mathbf{X}(n) - \mathbb{1}, \mathbf{0}\right)\right]}{||\mathbf{X}(n)||} + \frac{2E\left[\mathbf{A}^T(n)\mathbf{R}^*(n)\mathbf{X}(n)\right] + o\left(||\mathbf{X}(n)||\right)}{||\mathbf{X}(n)||} - \frac{E\left[\mathbf{X}^T(n)\mathbf{X}(n)|\mathbf{X}(n)\right]}{||\mathbf{X}(n)||}$$

where $\max(\mathbf{X}(n) - \mathbb{1}, \mathbf{0})$ was approximated with $\mathbf{X}(n) + o(||\mathbf{X}(n)||)$ in the second term, and terms not comprising $\mathbf{X}(n)$ were neglected at the numerator [since we consider large values of $||X(n)||$ in (9)].

Observing that $\max(\mathbf{X} - \mathbb{1}, \mathbf{0}) = \mathbf{X} - \min(\mathbf{X}, \mathbb{1})$

$$\begin{aligned} E\left[\max\left(\mathbf{X}(n) - \mathbb{1}, \mathbf{0}\right)^T \max\left(\mathbf{X}(n) - \mathbb{1}, \mathbf{0}\right)\right] &= \mathbf{X}^T(n)\mathbf{X}(n) - 2\min\left(\mathbf{X}^T(n), \mathbb{1}^T\right)\mathbf{X}(n) \\ &\quad + \min\left(\mathbf{X}(n)^T, \mathbb{1}^T\right)\min\left(\mathbf{X}(n), \mathbb{1}\right) \\ &= \mathbf{X}^T(n)\mathbf{X}(n) - 2\mathbb{1}^T\mathbf{X}(n) + o\left(||\mathbf{X}||\right) \end{aligned}$$

since $\min(\mathbf{X}^T, \mathbb{1}^T)\mathbf{X} = \mathbb{1}^T\mathbf{X} - o(||\mathbf{X}||)$. Finally

$$\frac{E\left[\mathcal{L}\left(\mathbf{X}(n+1)\right) - \mathcal{L}\left(\mathbf{X}(n)\right)|\mathbf{X}(n)\right]}{||\mathbf{X}(n)||} = \frac{2E\left[\mathbf{A}^T(n)\mathbf{R}^*(n)\right]\mathbf{X}(n) - 2\mathbb{1}^T\mathbf{X}(n) + o\left(||\mathbf{X}(n)||\right)}{||\mathbf{X}(n)||}. \quad (13)$$

Stability is guaranteed if there exist $\epsilon > 0$, $B > 0$ such that the Lyapunov drift is strictly negative for $||\mathbf{X}(n)|| > B$

$$\frac{\left(E\left[\mathbf{A}^T(n)\mathbf{R}^*(n)\right] - \mathbb{1}^T\right)\mathbf{X}(n)}{||\mathbf{X}(n)||} < -\epsilon. \quad (14)$$

To prove that the drift is negative, we first define the policy $\mathbf{R}'(\mathbf{X}(n)) = \arg\min_{\mathbf{R}} E[\mathbf{A}^T(n)]\mathbf{R}\mathbf{X}(n)$; note that

$$\mathbf{A}^T(n)\mathbf{R}^*(n)\mathbf{X}(n) = \min_{\mathbf{R}} \mathbf{A}^T(n)\mathbf{R}\mathbf{X}(n) \le \mathbf{A}^T(n)\mathbf{R}'(n)\mathbf{X}(n)$$

thus

$$E\left[\mathbf{A}^T(n)\mathbf{R}^*(n)\right]\mathbf{X}(n) \le E\left[\mathbf{A}^T(n)\right]\mathbf{R}'(n)\mathbf{X}(n).$$

Being the traffic pattern admissible, there exists a static routing $\hat{\mathbf{R}}$ such that $E[\mathbf{A}^T(n)]\hat{\mathbf{R}}\mathbf{X}(n) < \mathbb{1}^T\mathbf{X}(n)$. Finally, since $E[\mathbf{A}^T(n)]\mathbf{R}'(n)\mathbf{X}(n) \le E[\mathbf{A}^T(n)]\hat{\mathbf{R}}\mathbf{X}(n) < \mathbb{1}^T\mathbf{X}(n)$

$$\begin{aligned} E\left[\mathbf{A}^T(n)\mathbf{R}^*(n)\right]\mathbf{X}(n) - \mathbb{1}^T\mathbf{X}(n) &\le E\left[\mathbf{A}^T(n)\right]\mathbf{R}'(n)\mathbf{X}(n) - \mathbb{1}^T\mathbf{X}(n) \\ &= \min_{\mathbf{R}} E\left[\mathbf{A}^T(n)\right]\mathbf{R}\mathbf{X}(n) - \mathbb{1}^T\mathbf{X}(n) \\ &\le E\left[\mathbf{A}^T(n)\right]\hat{\mathbf{R}}\mathbf{X}(n) - \mathbb{1}^T\mathbf{X}(n) < -\epsilon \end{aligned}$$

hence, the nominator in (14) is negative. ■

The proof can be extended to the case in which the routing algorithm operates on a wrong estimate of $\mathbf{X}(n)$, as far as the mismatch $\mathbf{E}(n)$ between $\mathbf{X}(n)$ and its estimate $\tilde{\mathbf{X}}(n)$ is bounded:

Theorem 3: If there exists a finite k such that $||\mathbf{E}(n)|| < k, \forall \mathbf{X}(n)$, where $\mathbf{E}(n) = \tilde{\mathbf{X}}(n) - \mathbf{X}(n)$, then a dynamic routing algorithm $\mathbf{R}^+(n)$ defined as

$$\mathbf{R}^+(n) = \arg\min_{\mathbf{R}} \mathbf{A}^T(n)\mathbf{R}\tilde{\mathbf{X}}(n) \quad (15)$$

stabilizes the network under any admissible traffic pattern.

We call this algorithm *approximate min-backlogged-path routing*.

Proof: We build on the same calculations and on the same Lyapunov function as in the previous proof. Stability is guaranteed if there exist $\epsilon > 0, B > 0$ such that the Lyapunov function drift is strictly negative for $||\mathbf{X}(n)|| > B$

$$\frac{\left(E\left[\mathbf{A}^T(n)\mathbf{R}^+(n)\right] - \mathit{I}\!\mathit{I}^T\right)\mathbf{X}(n)}{||\mathbf{X}(n)||} < -\epsilon.$$

Define $\mathbf{E}(n) = \tilde{\mathbf{X}}(n) - \mathbf{X}(n)$. We note that

$$\begin{aligned}\mathbf{A}^T(n)\mathbf{R}^+(n)\tilde{\mathbf{X}}(n) &= \min_{\mathbf{R}} \mathbf{A}^T(n)\mathbf{R}(n)\tilde{\mathbf{X}}(n) \\ &\le \mathbf{A}^T(n)\mathbf{R}^*(n)\left[\mathbf{X}(n) + \mathbf{E}(n)\right].\end{aligned}$$

Thus

$$\begin{aligned}E\left[\mathbf{A}^T(n)\mathbf{R}^+(n)\tilde{\mathbf{X}}(n)|\mathbf{X}(n)\right] &\le E\left[\mathbf{A}^T(n)\mathbf{R}^*(n)\right]\mathbf{X}(n) \\ &\quad + E\left[\mathbf{A}^T(n)\mathbf{R}^*(n)\mathbf{E}(n)|\mathbf{X}(n)\right]\end{aligned}$$

being the last term bounded in norm, due to the fact that $\mathbf{E}(n)$ is bounded in norm by k.

Similarly

$$\begin{aligned}&E\left[\mathbf{A}^T(n)\mathbf{R}^+(n)\mathbf{X}(n)|\mathbf{X}(n)\right] \\ &\quad = E\left[\mathbf{A}^T(n)\mathbf{R}^+(n)\tilde{\mathbf{X}}(n)|\mathbf{X}(n)\right] \\ &\qquad - E\left[\mathbf{A}^T(n)\mathbf{R}^+(n)\mathbf{E}(n)|\mathbf{X}(n)\right] \\ &\quad = E\left[\mathbf{A}^T(n)\mathbf{R}^+(n)\tilde{\mathbf{X}}(n)|\mathbf{X}(n)\right] + o\left(||\mathbf{X}(n)||\right).\end{aligned}$$

As a consequence

$$\begin{aligned}&\frac{\left(E\left[\mathbf{A}^T(n)\mathbf{R}^+(n)\right] - \mathit{I}\!\mathit{I}^T\right)\mathbf{X}(n)}{||\mathbf{X}(n)||} \\ &\qquad \le \frac{\left(E\left[\mathbf{A}^T(n)\mathbf{R}^*(n)\right] - \mathit{I}\!\mathit{I}^T\right)\mathbf{X}(n)}{||\mathbf{X}(n)||} + o(1)\end{aligned}$$

which concludes the proof, since we have already proved that

$$\limsup_{||X(n)||\to\infty} \frac{\left(E\left[\mathbf{A}^T(n)\mathbf{R}^*(n)\right] - \mathit{I}\!\mathit{I}^T\right)\mathbf{X}(n)}{||\mathbf{X}(n)||} < -\epsilon.$$

■

Repeating the same arguments, stability of approximate min-backlogged-path routing can be proved in the two following more general cases:

1) $E[\mathbf{E}(n)] < \infty]$ and $E[\mathbf{E}^T(n)\mathbf{E}(n)] < \infty]$;
2) $\mathbf{E}(n)$ depends on $\mathbf{X}(n)$ being possibly unbounded when $||\mathbf{X}(n)|| \to \infty$; however $\lim_{||\mathbf{X}(n)||\to\infty}(\sqrt{E[\mathbf{E}^T(n)\mathbf{E}(n)|\mathbf{X}(n)]}/||\mathbf{X}(n)||) = 0$.

In both cases, by Cauchy–Schwarz inequality, it results

$$\begin{aligned}&E\left[\mathbf{A}^T(n)\mathbf{R}(n)\mathbf{E}(n)\right] \\ &\quad \le \sqrt{E\left[\left(\mathbf{A}^T(n)\mathbf{R}\right)\left(\mathbf{R}^T\mathbf{A}(n)\right)\right]E\left[\mathbf{E}^T(n)\mathbf{E}(n)\right]} \\ &\quad = o\left(||\mathbf{X}(n)||\right).\end{aligned}$$

This last result is of particular interest, since it allows us to conclude that the nice properties of the min-backlogged-path routing algorithm are maintained even when there are some mismatches between the current network state $\mathbf{X}(n)$ and the set of costs $\tilde{\mathbf{X}}(n)$ used by the routing algorithm to route new requests. This mismatch can be due for example to a delay in distributing updated link cost vectors, or to errors due to finite precision representation of costs. We can thus state that finite delays in propagating the link state information may affect general performance indexes, such as average file transfer times, but do not reduce the stability region of the routing algorithm.

We will not repeat the proof in case of wrong estimate of $\mathbf{X}(n)$ for the more general cases considered in the next sections, but the same arguments hold also for those cases.

V. Results for Feasible Scheduling Policies

In the previous section we have considered an idealized scenario where links are always fully utilized whenever their virtual backlog is not null. This optimistic situation is not sustained in practice, as already shown in example of Fig. 1.

Thus, (10) must be rewritten as

$$\mathbf{X}(n+1) = \mathbf{X}(n) - \mathbf{W}(n) + \mathbf{R}^T(n)\mathbf{A}(n) \qquad (16)$$

where $\mathbf{W}(n)$ is a vector whose lth element $w^l(n)$ represents the amount of work provided by link (queue) q^l to all the enqueued flows during time slot n, i.e., $\mathbf{W}(n)$ is the result of the *scheduling policy* implemented at the different links.

In general, the maximum amount of work that can be provided at a link l is not fully specified by the virtual backlog x^l, since it is also constrained by the dynamics of the flows that are physically available at the link, and possibly by the blocking properties of node architectures.

In this section, we consider scheduling policies according to which each flow in the network receives the same amount of work by all queues along its path. This assumption assures, indeed, that the sequence of selected $\mathbf{W}(n)$ is physically sustainable. With reference to the example of Fig. 1, the assumption we introduce in this section automatically forces $w^2(n)+w^3(n) = w^1(n) \le 1$. We notice that in packet networks, the presence of buffers at nodes allows to sustain short-term mismatches in the amount of work provided to a flow by the physical queues along its path. Under our assumption, instead backlog build-up is never experienced in the buffer at network nodes. However, note that the presence of end-to-end congestion control mechanisms for elastic traffic (such as the one imposed by TCP, or by explicit rate control mechanisms [17]–[19]) dynamically adapts the transmission rate of the source to the instantaneous available bandwidth along the path, by limiting the in-flight packet number for each flow to few units; it essentially forces the physical queues along the path to provide the same amount of work to in-transit flows, neglecting very short-term effects. Note that we are neglecting propagation delays, and (finite) transients at flow establishment and termination.

Let $w^\pi(n) \ge 0$ be the amount of work provided to the flows routed on path π by all the queues $l \in L(\pi)$ by time n, and $\mathbf{W}^\pi(n)$ the corresponding column vector. Let $x^\pi(n)$ be the virtual backlog at time n due to flows routed on path $\pi \in \Pi(sd)$ on every queue $l \in L(\pi)$ (this is identical for all queues due

to the assumptions of this section). Let $\mathbf{X}^\pi(n)$ the vector of the $x^\pi(n)$.

Definition 4: Vector $\mathbf{W}(n)$ is physically sustainable if

$$w^l(n) = \sum_{\pi \in \Pi(l)} w^\pi(n) \leq 1 \quad \forall l \tag{17}$$

$$w^\pi(n) \leq x^\pi(n). \tag{18}$$

We call *feasible scheduling policy* a policy that at each time slot finds a sustainable $\mathbf{W}(n)$.

The scheduling policy computes either the vector $\mathbf{W}(n)$, i.e., the amount of provided work, or a set of guaranteed service rates, from which the provided work is derived according to some algorithm (e.g., using GPS [20]). We call $b^\pi(n)$ the minimum guaranteed service rate associated with the flow routed on path π at all queues, and $\mathbf{B}^\pi(n)$ the corresponding column vector. The $b^\pi(n)$ are constrained by a relation similar to (17)

$$\sum_{\pi \in \Pi(l)} b^\pi(n) \leq 1 \quad \forall l \tag{19}$$

but do not need to satisfy the equivalent of (18), and do not necessarily depend from $\mathbf{X}(n)$. A simple way of deriving $w^\pi(n)$ from $b^\pi(n)$ is the following:

$$w^\pi(n) = \min\left(x^\pi(n), b^\pi(n)\right). \tag{20}$$

The dynamics of the path virtual backlogs can be written as

$$\mathbf{X}^\pi(n+1) = \mathbf{X}^\pi(n) - \mathbf{W}^\pi(n) + \left(\mathbf{R}^\pi(n)\right)^T \mathbf{A}(n). \tag{21}$$

Since physical backlog at network queues is negligible, the elements $x^\pi(n)$ of $\mathbf{X}^\pi$ acquires the meaning of the amount of information belonging to flows that have been routed on path π still waiting for transmission at the source.

We are now in a position to generalize the results presented in the previous section, proving that, under any admissible traffic pattern, a network adopting a routing similar to the min-backlogged-path routing in conjunction with a generic feasible scheduling policy is stable.

However, we first need to prove that traffic admissibility and traffic sustainability are equivalent also under the assumptions made in this section. The basic statement is that this equivalence holds if the service rates provided by the scheduling policy are matched with the average loads routed through network nodes. A GPS-like scheduler easily achieves this result.

A. Sustainability Versus Admissibility

To prove that sustainability is equivalent to admissibility, we must show that there always exists a static routing function $\hat{\mathbf{R}}$ and a static (time invariant, so that we drop the dependence from n) feasible scheduling policy such that

$$b^\pi > \hat{r}^\pi_{\mathrm{sd}} \rho_{\mathrm{sd}} \quad \forall \pi \tag{22}$$

i.e., the service rate guaranteed to active flows routed on π by queues along π is greater than the average workload arrival rate on path π.

Lemma 1: Any *admissible* average arrival vector $E[\mathbf{A}]$ is *sustainable*.

TABLE I
SUMMARY OF CONSIDERED DYNAMIC ROUTING ALGORITHMS

Routing algorithm	Definition
min-backlogged-path	$\arg\min_{\mathbf{R}} \mathbf{A}^T(n)\mathbf{R}\mathbf{X}(n)$
min-backlogged-flow	$\arg\min_{\mathbf{R}^\pi} \mathbf{A}^T(n)\mathbf{R}^\pi\mathbf{X}^\pi(n)$
min-congested-flow	$\arg\min_{\mathbf{R}^\pi} \mathbf{A}^T(n)\mathbf{R}^\pi\nabla(\mathbf{G}^\pi(\mathbf{X}^\pi(n)))$
min-congested-link	$\arg\min_{\hat{\pi}\in\Pi(sd)} \max_{l\in L(\hat{\pi})} x^l(n)$

TABLE II
SUMMARY OF CONSIDERED SCHEDULING POLICIES

Scheduling policy	Definition
max-scalar path	$\arg\max_{\mathbf{W}} \mathbf{W}^T\mathbf{X}^\pi(n)$
generalized max-scalar path	$\arg\max_{\mathbf{W}} \mathbf{W}^T\nabla(\mathbf{G}^\pi(\mathbf{X}^\pi(n)))$
max-backlog-proportional	$b^\pi(n) = \dfrac{x^\pi(n)}{\max_{l\in L(\pi)} x^l(n)}$

Proof: According to the definition of admissibility, given an admissible traffic pattern, there exists a static routing algorithm $\hat{\mathbf{R}}$, such that, for some $\epsilon > 0$

$$\hat{\mathbf{R}}^T E[\mathbf{A}] \leq \mathbb{1}(1-2\epsilon).$$

Let us assign to each path $\pi \in \Pi(sd)$ a static service rate $b^\pi_{\mathrm{sd}} = (1+\epsilon)\hat{r}^\pi_{\mathrm{sd}}\rho_{\mathrm{sd}}$. Call $b^l = \sum_{\pi\in\Pi(l)} b^\pi_{\mathrm{sd}}$ the link working rates, and $\mathbf{B}$ the corresponding column vector.

By construction, $\mathbf{B} = (1+\epsilon)\hat{\mathbf{R}}^T E[\mathbf{A}] < \mathbb{1}$ defines a set of sustainable link working rates that stabilizes the network. ■

Note that, from the previous Lemma, it immediately follows that, for each admissible average arrival vector $\mathbf{A}$, $\hat{\mathbf{R}}^T E[\mathbf{A}]$ lies in the convex-hull of the sustainable service rates $\mathbf{B}$.

B. First Result

We now extend the result of Theorem 2 to the general case of feasible scheduling policies, i.e., to the case where $\mathbf{W}(n)$ is constrained to be sustainable. Therefore, we need to define a scheduling algorithm that computes $\mathbf{W}(n)$ when the chosen dynamic routing algorithm is adopted.

Definition 5: We define the *max-scalar path scheduling policy* as the feasible scheduling policy computing the provided work $\mathbf{W}^{*\pi}(n)$ that maximizes the scalar product $\mathbf{W}^T\mathbf{X}^\pi(n)$. That is, according to $\mathbf{W}^{*\pi}(n)$, the amount of work provided along paths $w^\pi(n) \geq 0$ is selected for which

$$\mathbf{W}^{*\pi}(n) = \arg\max_{\mathbf{W}} \mathbf{W}^T\mathbf{X}^\pi(n)$$

with $\mathbf{W}$ subject to (17) and (18).

We now define a modification of the min-backlogged-path routing.

Definition 6: A dynamic routing algorithm $\mathbf{R}^{*\pi}(n)$ such that

$$\mathbf{R}^{*\pi}(n) = \arg\min_{\mathbf{R}} \mathbf{A}^T(n)\mathbf{R}\mathbf{X}^\pi(n) \tag{23}$$

is called *min-backlogged-flow routing*.

This routing algorithm selects the path with the minimum virtual backlog, without accounting for the hop count along the path.

To help the reader in recalling all the definitions, Table I summarizes all considered routing algorithms, while Table II summarizes scheduling policies.

We introduce the following a preliminary result:

Lemma 2: Let

$$\mathbf{B}^{*\pi}(n) = \arg\max_{\mathbf{B}} \mathbf{B}^T \mathbf{X}^{\pi}(n)$$

with $\mathbf{B}$ subject to (19). It results

$$\left[\mathbf{B}^{*\pi}(n) - \mathbf{W}^{*\pi}(n)\right]^T \mathbf{X}^{\pi}(n) \leq \left(\mathbf{B}^{*\pi}(n)\right)^T \mathbf{B}^{*\pi}(n)$$

i.e, the difference between $(\mathbf{B}^{*\pi}(n))^T \mathbf{X}^{\pi}(n)$ and $(\mathbf{W}^{*\pi}(n))^T \mathbf{X}^{\pi}(n)$ is bounded by a finite constant.

Proof: Recalling (20), we have

$$\begin{aligned}
&\left[\mathbf{B}^{*\pi}(n) - \mathbf{W}^{*\pi}(n)\right]^T \mathbf{X}^{\pi}(n) \\
&\quad = \left(\mathbf{B}^{*\pi}(n)\right)^T \mathbf{X}^{\pi}(n) - \max_{W} \mathbf{W}^T \mathbf{X}^{\pi}(n) \\
&\quad \leq \left(\mathbf{B}^{*\pi}(n)\right)^T \mathbf{X}^{\pi}(n) - \min\left(\mathbf{X}^{\pi}(n), \mathbf{B}^{*\pi}(n)\right)^T \mathbf{X}^{\pi}(n) \\
&\quad = \left[\mathbf{B}^{*\pi}(n) - \min\left(\mathbf{X}^{\pi}(n), \mathbf{B}^{*\pi}(n)\right)\right]^T \mathbf{X}^{\pi}(n) \\
&\quad = \max\left(\mathbf{B}^{*\pi}(n) - \mathbf{X}^{\pi}(n), 0\right)^T \mathbf{X}^{\pi}(n) \\
&\quad \leq \max\left(\mathbf{B}^{*\pi}(n) - \mathbf{X}^{\pi}(n), 0\right)^T \mathbf{B}^{*\pi}(n) \\
&\quad \leq \left(\mathbf{B}^{*\pi}(n)\right)^T \mathbf{B}^{*\pi}(n).
\end{aligned}$$

Note, indeed, that $\max(b-x,0)x \leq \max(b-x,0)b$ holds for every $x \geq 0, b \geq 0$. ■

We are now ready to introduce our first main result.

Theorem 4: For any sustainable traffic pattern, a network is stable if the min-backlogged-flow routing and the max-scalar path scheduling policy are adopted.

Proof: We prove this result by applying the Lyapunov function methodology, following the same approach used in the proof of Theorem 2. Let us consider the quadratic Lyapunov function, as in (12). Stability is guaranteed if there exist $\epsilon > 0$ and $B > 0$ such that, for $\|\mathbf{X}(n)\| > B$, the Lyapunov function drift is strictly negative

$$\begin{aligned}
&\frac{E\left[\mathcal{L}\left(\mathbf{X}^{\pi}(n+1)\right) - \mathcal{L}\left(\mathbf{X}^{\pi}(n)\right) | \mathbf{X}^{\pi}(n)\right]}{\|\mathbf{X}^{\pi}(n)\|} \\
&\quad = \frac{2\left\{E\left[\mathbf{A}^T(n)\mathbf{R}^{*\pi}(n)\right] - \left(\mathbf{W}^{*\pi}(n)\right)^T\right\}\mathbf{X}^{\pi}(n)}{\|\mathbf{X}^{\pi}(n)\|} + o(1)
\end{aligned}$$

having neglected at the numerator terms not comprising $\mathbf{X}(n)$.

To prove that the drift is negative, we define the routing algorithm $\mathbf{R}'^{\pi}(n) = \arg\min_{\mathbf{R}} E[\mathbf{A}^T(n)]\mathbf{R}\mathbf{X}^{\pi}(n)$; note that

$$\begin{aligned}
\mathbf{A}^T(n)\mathbf{R}^{*\pi}(n)\mathbf{X}^{\pi}(n) &= \min_{\mathbf{R}} \mathbf{A}^T(n)\mathbf{R}\mathbf{X}^{\pi}(n) \\
&\leq \mathbf{A}(n)\mathbf{R}'^{\pi}(n)\mathbf{X}^{\pi}(n)
\end{aligned}$$

thus

$$E\left[\mathbf{A}^T(n)\mathbf{R}^{*\pi}(n)\right]\mathbf{X}^{\pi}(n) \leq E\left[\mathbf{A}(n)\right]\mathbf{R}'^{\pi}(n)\mathbf{X}^{\pi}(n).$$

Finally

$$\begin{aligned}
&E\left[\mathbf{A}(n)\right]^T \mathbf{R}'^{\pi}(n)\mathbf{X}^{\pi}(n) - \left(\mathbf{W}^{*\pi}(n)\right)^T \mathbf{X}^{\pi}(n) \\
&\quad = \min_{\mathbf{R}} E\left[\mathbf{A}^T(n)\right]\mathbf{R}\mathbf{X}^{\pi}(n) - \left(\mathbf{W}^{*\pi}(n)\right)^T \mathbf{X}^{\pi}(n) \\
&\quad \leq E\left[\mathbf{A}^T(n)\right]\hat{\mathbf{R}}^{\pi}\mathbf{X}^{\pi}(n) - \left(\mathbf{W}^{*\pi}(n)\right)^T \mathbf{X}^{\pi}(n) \\
&\quad = E\left[\mathbf{A}^T(n)\right]\hat{\mathbf{R}}^{\pi}\mathbf{X}^{\pi}(n) - \max_{\mathbf{W}} \mathbf{W}^T \mathbf{X}^{\pi}(n) \\
&\quad \leq E\left[\mathbf{A}^T(n)\right]\hat{\mathbf{R}}^{\pi}\mathbf{X}^{\pi}(n) - \max_{\mathbf{B}} \mathbf{B}^T \mathbf{X}^{\pi}(n) \\
&\qquad + o\left(\|\mathbf{X}^{\pi}(n)\|\right) < -\epsilon\|\mathbf{X}^{\pi}(n)\|
\end{aligned}$$

where $\hat{\mathbf{R}}^{\pi}$ is a static routing leading to admissibility. In the second-last equality the provided work $\mathbf{W}^{\pi}$ was approximated with the service rates $\mathbf{B}^{\pi}$ thanks to Lemma 2. Moreover, the last inequality holds since $E[\mathbf{A}^T(n)]\hat{\mathbf{R}}^{\pi}$ is in the convex hull of sustainable service rates $\mathbf{B}^{\pi}$ (due to Lemma 1) if $\mathbf{A}$ is admissible. ■

Note that we do not need that

$$\mathbf{W}^{*\pi}(n) = \arg\max_{\mathbf{W}} \mathbf{W}^T \mathbf{X}^{\pi}(n).$$

We just need

$$\mathbf{W}^{*\pi T}(n)\mathbf{X}^{\pi}(n) > E\left[\mathbf{A}^T(n)\right]\hat{\mathbf{R}}^{\pi}\mathbf{X}^{\pi}(n).$$

C. Considering Generalized Virtual Backlog Costs

A further extension of the previous result can be obtained considering generalized cost functions of the virtual backlog.

Let $g^{\pi}(x)$ be a set $\mathbb{R}^+ \rightarrow \mathbb{R}^+$ functions in $C^{\infty}[0,\infty]$, that are strictly increasing and convex, with $g^{\pi}(0) = 0$, $(dg^{\pi}(x)/dx)|_{x=0} > 0$ and

$$\lim_{x\to\infty} \frac{\frac{dg^{\pi(n+1)}(x)}{dx}}{\frac{dg^{\pi(n)}(x)}{dx}} = 0 \quad \forall n.$$

We use $g^{\pi}(x^{\pi})$ as congestion index related to path π, i.e., we define as link l congestion index $g^l(n) = \sum_{\pi\in\Pi(l)} g^{\pi}(x^{\pi}(n))$. Let us modify the min-backlogged-flow routing. Let $\mathbf{G}^{\pi}(\mathbf{X}^{\pi}(n))$ be the vector with components $g^{\pi}(x^{\pi}(n))$.

Definition 7: A dynamic routing algorithm $\mathbf{R}^{*\pi}(n)$ such that

$$\mathbf{R}^{*\pi}(n) = \arg\min_{\mathbf{R}} \mathbf{A}^T(n)\mathbf{R}\nabla\left(\mathbf{G}^{\pi}\left(\mathbf{X}^{\pi}(n)\right)\right) \tag{24}$$

being $\nabla(\mathbf{G}^{\pi}(\mathbf{X}^{\pi}(n)))$ the vector whose elements are $g^{\pi}(x^{\pi})(\partial g^{\pi}(x^{\pi})/\partial x^{\pi})$, is called *min-congested-flow routing*.

This routing algorithm selects the least congested path among all paths from qs to d according to the congestion definition above.

Definition 8: We define the *generalized max-scalar path scheduling policy* as the feasible scheduling policy computing the provided work $\mathbf{W}^{*\pi}(n)$ that maximizes the scalar product $\mathbf{W}^T\nabla(\mathbf{G}^{\pi}(\mathbf{X}^{\pi}(n)))$.

That is, according to $\mathbf{W}^{*\pi}(n)$, the amount of work provided along paths $w^{\pi}(n) \geq 0$ is selected for which

$$\mathbf{W}^{*\pi}(n) = \arg\max_{\mathbf{W}} \mathbf{W}^T\nabla\left(\mathbf{G}^{\pi}\left(\mathbf{X}^{\pi}(n)\right)\right) \tag{25}$$

subject to (17) and (18).

Theorem 5: For any sustainable traffic pattern, a network is stable if the min-congested-flow routing and the generalized max-scalar path scheduling policy are adopted.

Proof: This proof is a straightforward extension of the proof of Theorem 4.

Consider the Lyapunov function

$$\mathcal{L}\left(\mathbf{X}(n)\right) = \left(\mathbf{G}^{\pi}\left(\mathbf{X}^{\pi}(n)\right)\right)^T \mathbf{G}^{\pi}\left(\mathbf{X}^{\pi}(n)\right).$$

It results

$$E\left[\mathcal{L}\left(\mathbf{X}^{\pi}(n+1)\right)-\mathcal{L}\left(\mathbf{X}^{\pi}(n)\right)|\mathbf{X}^{\pi}(n)\right] = 2E\left[\left(2\mathbf{A}^{T}\mathbf{R}^{*\pi}-\left(\mathbf{W}^{*\pi}\right)^{T}\right)\right]\nabla\left(\mathbf{G}^{\pi}\left(\mathbf{X}^{\pi}(n)\right)\right) + o\left(\left\|\mathbf{G}^{\pi}\left(\mathbf{X}^{\pi}(n)\right)\right\|\right).$$

Proceeding exactly as in the proof of Theorem 4, the result can be obtained. ∎

D. Considering Simpler Scheduling Algorithms

Unfortunately, the implementation of the max-scalar scheduling policy or its generalization can be difficult, since it requires the solution of a difficult optimization problem [as defined in Definition 5, or (25)] which is hard to implement in a distributed scenario.

We thus need to look for other pairs of dynamic routing algorithm and feasible scheduling policy that guarantee maximum throughput, while also being easy to implement in a distributed environment. We define in the following a general framework that allows to characterize pairs of dynamic routing algorithm and feasible scheduling policy that guarantee maximum throughput.

Definition 9: A scheduling policy is (δ,ϵ) efficient if, for every $\epsilon>0$, there exists a $\delta>0$ and a $B>0$ such that, defined

$$J_{\delta}^{*}(n)=\left\{l\in E|\frac{x^{l}(n)}{\max_{l'\in E}x^{l'}(n)}>1-\delta\right\} \tag{26}$$

whenever $\max_{l'\in E}x^{l'}(n)>B$

$$w^{l}(n)>1-\epsilon \quad \forall l\in J_{\delta}^{*}(n).$$

The previous definition states that whenever the network link congestion exceeds B, the amount of work provided by bottleneck links (i.e., links in set J_{δ}^{*}) is almost one.

Theorem 6: If flows from s to d are routed on path π, with

$$\pi=\arg\min_{\hat{\pi}\in\Pi(sd)}\max_{l\in L(\hat{\pi})}x^{l}(n)$$

and the network adopts a (δ,ϵ) efficient scheduling, then the network is stable under any admissible traffic.

In the case of ties, i.e, when two paths π_1 and π_2 exist, such that $\max_{l\in L(\pi_1)}x^{l}(n)=\max_{l\in L(\pi_2)}x^{l}(n)$, the path with the least loaded second highest-load link is selected. In case of further tie, the third highest-load link is considered, and so on. In case links of π_1 present the same congestion on the sequence of highest loaded links of π_2, the shortest path will be selected. This routing algorithm will be called *min-congested-link routing*, since flows $s\rightarrow d$ are routed along the path whose highest-load link is least loaded.

Proof: The proof is reported in Appendix A. ∎

A possible (δ,ϵ)-efficient policy is the allocation of guaranteed service rates according to

$$b^{\pi}(n)=\frac{x^{\pi}(n)}{\max_{l\in L(\pi)}x^{l}(n)}. \tag{27}$$

Using (20)

$$\begin{aligned}w^{\pi}(n)&=\min\left(b^{\pi}(n),x^{\pi}(n)\right)\\&=b^{\pi}(n)\\&=\frac{x^{\pi}(n)}{\max_{l\in E}x^{l}(n)}\qquad\text{for }\max_{l\in E}x^{l}(n)>1.\end{aligned}$$

Summing both sides for $\pi\in\Pi(l)$, and choosing $\delta=\epsilon$, we easily find that this policy is (δ,ϵ)-efficient.

We call this scheduling policy *max-backlog-proportional scheduling*; it provides service to flows proportionally to their virtual backlog with respect to the virtual backlogs of flows sharing the most congested link along the path. Note that it does not guarantee to fully utilize the link capacities, and that it does not require any knowledge of traffic matrix $\boldsymbol{\rho}$.

A generalization of the max-backlog-proportional scheduling is possible. Indeed

$$w^{\pi}(n)\geq\frac{y^{\pi}(n)}{\max_{l\in L(\pi)}x^{l}(n)}\qquad\text{for }\max_{l\in L(\pi)}x^{l}(n)>1 \tag{28}$$

being $y^{\pi}(n)$ any set of nonnegative numbers satisfying the property

$$x^{l}(n)\leq\sum_{\pi\in\Pi(l)}y^{\pi}(n)\leq\max_{l\in L(\pi)}x^{l}(n)\quad\forall l$$

can be easily proved to fall in the class of (δ,ϵ) policy. We call this class of policies *generalized max-backlog-proportional scheduling*.

VI. System State in Terms of Number of Flows

In previous sections the network dynamics was expressed in terms of flows backlogs. This requires that link state metrics account for the backlogs of the different flows, which may be difficult to implement. In this section we instead define the system state in terms of the number of active flows. We start by considering exponentially distributed flow durations.

If flow lengths are exponentially distributed, the network can be modeled by a Markov chain whose state descriptor $\mathbf{Z}^{\pi}(n)$ is an $|E|$-dimensional vector whose πth element $z^{\pi}(n)$ represents the number of flows that are traversing path π at time n.

The dynamics of $z^{\pi}(n)$ are described by

$$z^{\pi}(n+1)=z^{\pi}(n)+\sum_{\mathrm{sd}}\gamma_{\mathrm{sd}}(n)r_{\mathrm{sd}}^{\pi}(n)-d^{\pi}(n)$$

where $\gamma_{\mathrm{sd}}(n)$ represents the number of new flows from node s to node d arrived at the network during time slot n, and $d^{\pi}(n)$ is a random variable that denotes the number of flows that completed the data transfer at time slot n.

Rewriting the previous equation using vectorial notation, we obtain

$$\mathbf{Z}^{\pi}(n+1)=\mathbf{Z}^{\pi}(n)+\mathbf{R}^{\pi}(n)^{T}\boldsymbol{\Gamma}(n)-\mathbf{D}^{\pi}(n) \tag{29}$$

being easy to verify that i) $E[\mathbf{\Gamma}(n)] = \mu E[\mathbf{A}(n)]$, and ii) $E[\mathbf{D}^{\pi}(n)|\mathbf{Z}^{\pi}(n)] \to \mu \mathbf{W}^{\pi}(n)$ for $||\mathbf{Z}^{\pi}(n)|| \to \infty$. Thus, for $||\mathbf{Z}^{\pi}(n)|| \to \infty$

$$E\left[\mathbf{Z}^{\pi}(n+1) - \mathbf{Z}^{\pi}(n)|\mathbf{Z}^{\pi}(n)\right] = \mu \mathbf{R}^{\pi}(n)^{T} E\left[\mathbf{A}(n)\right] - \mu \mathbf{W}^{\pi}(n) + o(1)$$

which highlights the fact that the structure of (29) is asymptotically (for large $||\mathbf{Z}^{\pi}(n)||$) identical to the structure of (21), under the variable substitution $\mathbf{X}^{\pi} \to \mathbf{Z}^{\pi}$. As a consequence, the results obtained in the previous section can be immediately extended for "companion" routing algorithms and scheduling policies that operate on $\mathbf{Z}^{\pi}$ rather than $\mathbf{X}^{\pi}$. A formal proof can be obtained by applying the stochastic Lyapunov functions obtained by substituting $\mathbf{Z}^{\pi}$ to $\mathbf{X}^{\pi}$.

Therefore, the following results can be immediately derived.

We can redefine the dynamic min-backlogged-flow routing algorithm as the routing algorithm $\mathbf{R}^{*}(n)$ such that:

$$\mathbf{R}^{*\pi}(n) = \arg\min_{\mathbf{R}} \mathbf{A}^{T}(n)\mathbf{R}\mathbf{Z}^{\pi}(n).$$

We can redefine the max-scalar path scheduling policy as the policy $\mathbf{W}^{*\pi}(n)$ that maximizes the scalar product $\mathbf{W}^{T}\mathbf{Z}^{\pi}(n)$. That is

$$\mathbf{W}^{*\pi}(n) = \arg\max_{\mathbf{W}} \mathbf{W}^{T}\mathbf{Z}^{\pi}(n)$$

subject to (17) and (18).

Let $z^{l}(n) = \sum_{\pi \in \Pi(l)} z^{\pi}(n)$. We can redefine the max-backlog-proportional scheduling policy or its generalizations to operate on the flow number $\mathbf{Z}^{\pi}$ rather than the path backlog $\mathbf{X}^{\pi}$, also explicitly considering different link capacities.

The max-backlog-proportional scheduling becomes

$$b^{\pi}(n) = \frac{z^{\pi}(n)}{z^{l'}(n)/C^{l'}} \tag{30}$$

being l' the network bottleneck (i.e., $l' = \arg\max_{l \in E}(z^{l}(n)/C^{l})$).

We can finally redefine the min-congested-link routing algorithm as follows:

$$\pi = \arg \min_{\hat{\pi} \in \Pi(sd)} \max_{l \in L(\hat{\pi})} \frac{z^{l}(n)}{C^{l}.}$$

Then, we can state the following two theorems, which are direct consequences of Theorems 4 and 6.

Theorem 7: If flow lengths are exponentially distributed, for any sustainable traffic pattern, a network adopting the min-backlogged-flow routing algorithm and the max-scalar-path scheduling policy is stable.

Theorem 8: If flow lengths are exponentially distributed, for any sustainable traffic pattern, a network adopting the min-congested-link routing algorithm and the max-backlog-proportional scheduling policy is stable.

This latter theorem refers to a routing algorithm that is similar to what has been proposed in actual packet networks, since it corresponds to a minimum distance routing adopting the L_{∞} norm (i.e, where only the bottleneck link is considered) [11].

The assumption of exponential flow sizes may be unrealistic. However, the results just obtained for exponentially distributed flow sizes can be extended to flows with generally distributed size (with mean $1/\mu$ and finite polynomial moments). Indeed, we observe that in our model each link can be described by a single-server processor sharing queue if we assume that the flows uniformly share the available capacity (i.e., a max-backlog-proportional scheduling policy is adopted). Thus, we claim that the following property holds in our system:

$$\lim_{||\mathbf{Z}^{\pi}(n)|| \to \infty} \frac{E\left[||\mu^{*}\mathbf{X}^{\pi}(n) - \mathbf{Z}^{\pi}(n)||\right]}{||\mathbf{Z}^{\pi}(n)||} = 0 \tag{31}$$

being μ^{*} a constant which depends on the flow length distribution. Under the above property the number of flows becomes a good estimator of the residual workload at the queue when the workload in the network becomes arbitrarily large. This property is satisfied by systems subject to the *state space collapse* phenomenon(or multiplicative state space collapse)[5] [22], [23], which has been already proved for several queuing systems, among which the G/G/1 Processor Sharing queue [24].

Under the state space collapse assumption, we can extend the results obtained for exponentially distributed flows also to the generally distributed flow size, since 1) routing algorithms and scheduling policies that operate on $\mathbf{Z}^{\pi}(n)$ and $\mathbf{X}^{\pi}(n)$ become equivalent when $||\mathbf{Z}^{\pi}(n)|| \to \infty$; and 2) policies operating on $\mathbf{X}^{\pi}(n)$ were already proved to maximize network throughput.

VII. Modeling Buffers Along the Path

In this section we explicitly consider the effect of buffers inside the network; buffers allow to sustain a temporary mismatch in the amount of work provided to a flow along its path. In this case, since the physical backlog enqueued in the network is no longer negligible, the virtual backlog of active flows routed on path π may be different at different queues.

We extend to this general case the theoretical framework developed in previous sections. In order to properly describe the virtual backlog dynamics in the network, we have to directly represent the evolution of the backlog at the sources.

Let $x^{l,\pi}(n)$ be the virtual backlog at queue q^{l} due to flows scheduled for path π that at time n must still be forwarded by link l. Let $x^{s,\pi}(n)$, instead the backlog at source s, i.e., the amount of information belonging to flows routed on path π that has still to be transmitted from source s. Let $w^{l,\pi}(n)$ the amount of work provided by link l in time slot n to flows routed through path π. Let $w^{s,\pi}(n)$ be the amount of information injected on path π by the source s.

The dynamical evolution of virtual backlog in the network and at the source nodes is driven by

$$x^{l,\pi}(n+1) = x^{l,\pi}(n) - w^{l,\pi}(n) + \sum_{\mathrm{sd}} r^{\pi}_{\mathrm{sd}}(n) a_{\mathrm{sd}}(n) \tag{32}$$

[5]Space collapse is an asymptotic property of many multidimensional Markovian (possibly over general space states) queueing systems, which appears in heavy-traffic conditions, i.e., under diffusion scaling, when the system load approach 1. If space collapse occurs the queueing system asymptotically lives in a lower dimensional subspace of the original space state, with particular properties.

$$x^{s,\pi}(n+1) = x^{s,\pi}(n) - w^{s,\pi}(n) + \sum_{\text{sd}} r^{\pi}_{\text{sd}}(n) a_{\text{sd}}(n). \quad (33)$$

Let $p^{l,\pi}(n)$ the physical backlog at time n at queue l due to flow routed along path π, i,e., the amount of information (bytes) belonging to flows routed through path π that are physically enqueued at time n in queue q^l.

We remind that, by definition, $x^{l,\pi}(n)$ is the sum of three contributions: 1) the physical backlog contribution, $p^{l,\pi}(n)$, at queue q^l; 2) the physical backlog contribution that is enqueued in upstream queues; and 3) the backlog at source s, $x^{s,\pi}(n)$.

As a consequence, being m the link preceding l along π, we have

$$p^{l,\pi}(n) = \begin{cases} x^{l,\pi}(n) - x^{s,\pi}(n) \geq 0, & \text{if } l \text{ is the first link of } \pi \\ x^{l,\pi}(n) - x^{m,\pi}(n) \geq 0, & \text{otherwise.} \end{cases} \quad (34)$$

Now we generalize the definition of sustainable amount of work provided to flows by links and sources.

Definition 10: The amount of work provided to flows by links $\{w^{l,\pi}(n), w^{s,\pi}(n)\}$ is physically sustainable if

$$w^l(n) = \sum_{\pi \in \Pi(l)} w^{l,\pi}(n) \leq 1 \quad \forall l \quad (35)$$

$$w^{l,\pi}(n) \leq \begin{cases} p^{l,\pi}(n) + w^{s,\pi}(n), & \text{if } l \text{ is the first link of } \pi \\ p^{l,\pi}(n) + w^{m,\pi}(n), & \text{otherwise} \end{cases} \quad (36)$$

$$w^{s,\pi}(n) \leq x^{s,\pi(n)}. \quad (37)$$

We call *feasible scheduling policy* a policy that at each time slot finds a sustainable set $\{w^{l,\pi}(n), w^{s,\pi}(n)\}$. We notice that the class of feasible scheduling policies studied in the previous sections forces $w^{s,\pi}(n) = w^{l,\pi}(n) = w^{\pi}(n)$ for any $l \in L(\pi)$ and, therefore, are a proper subset of sustainable policies considered in this section.

As before, the scheduling policy can compute either directly the amounts of work $\{w^{l,\pi}(n), w^{s,\pi}(n)\}$ to provide along the path, or a set of guaranteed service rates $\{b^{l,\pi}(n), b^{s,\pi}(n)\}$, which are constrained by

$$\sum_{\pi \in \Pi(l)} b^{l,\pi}(n) \leq 1 \quad \forall l. \quad (38)$$

In this case, the relationships between $\{b^{l,\pi}(n), b^{s,\pi}(n)\}$ and $\{w^{l,\pi}(n), w^{s,\pi}(n)\}$ are more complex

$$w^{s,\pi}(n) = \min\left(b^{s,\pi}(n), x^{s,\pi}(n)\right)$$

and (39), shown at the bottom of the page.

It is worth to note that, if $p^{l,\pi}(n) = 0$, then $w^{l,\pi}(n) = w^{s,\pi}(n) = w^{\pi}(n)$ and $b^{l,\pi}(n) = b^{\pi}(n)$ for every l along π, (39) becomes equal to (20).

Under this general framework it is possible to extend the results found in previous sections.

Theorem 9: If flows from s to d are routed on path π, with

$$\pi = \arg \min_{\hat{\pi} \in \Pi(sd)} \max_{l \in L(\hat{\pi})} \sum_{\pi' \in \Pi(l)} x^{l,\pi'}(n)$$
$$= \arg \min_{\hat{\pi} \in \Pi(sd)} \max_{l \in L(\hat{\pi})} x^l(n)$$

and the network adopts any (δ, ϵ) efficient scheduling policy, then the network is stable under any admissible traffic.

Proof: The proof of this result is similar to the proof of Theorem 6 and is given in Appendix B. ■

In particular, we can consider a scheduling policy that allocates bandwidths at links according to the following rule:

$$b^{l,\pi}(n) = \frac{x^{l,\pi}(n)}{x^l(n)}. \quad (40)$$

It is easy to verify that (40) defines a (δ, ϵ) efficient scheduling policy, which relies only on information available at each link, i.e., on local information.

VIII. Conclusion

In this paper, we considered packet networks loaded by admissible traffic patterns, i.e., traffic patterns that, if optimally routed, do not overload network resources. We proved that several combinations of simple distributed dynamic routing algorithms and scheduling policies based upon link state information can achieve the same network throughput as optimal centralized routing and scheduling algorithms with complete traffic information.

Our proofs are based on abstract *flow-level* models of the network, consider *elastic* traffic, and exploit the Lyapunov function methodology.

We showed that maximum throughput is achieved even in case of temporary mismatches between the actual link costs and those used by the routing algorithm. This implies that our proofs hold even for distributed implementations, which lead to errors in the estimation of routing metrics.

Special attention was given to the case of exponentially distributed flow sizes, which permit particularly simple routing and scheduling algorithms.

Although the contribution of this paper is mostly theoretical, we believe that the implementation of our algorithms is not unrealistic in the framework of currently considered QoS approaches for the Internet. A discussion on the implementability of our schemes can be found in [25].

Appendix

A. Proof of Theorem 6

Let $x^l(n) = \sum_{\pi \in \Pi(l)} x^{\pi}(n)$. Given the state of the system $x^{\pi}(n)$, there exists a sufficiently large constant m_0 such that

$$w^{l,\pi}(n) = \begin{cases} \min\left(b^{l,\pi}(n), p^{l,\pi}(n) + w^{s,\pi}(n)\right), & \text{if } l \text{ is the first link along } \pi \\ \min\left(b^{l,\pi}(n), p^{l,\pi}(n) + w^{m,\pi}(n)\right), & \text{otherwise} \end{cases} \quad (39)$$

the min-congested-link routing algorithm can be defined, for $m > m_0$, as

$$\mathbf{R}^*(n) = \arg\min_{\mathbf{R}} \sum_{\mathrm{sd}} a_{\mathrm{sd}}(n) \sum_{\pi\in\Pi(sd)} r^{\pi}_{\mathrm{sd}} \sum_{l\in L(\pi)} \left(x^l(n)\right)^m. \tag{41}$$

We have replaced the maximum operator by a sufficiently large power. Note that this requires the ties resolution approach described for the theorem. From (41), clearly

$$\sum_{\mathrm{sd}} a_{\mathrm{sd}}(n) \sum_{\pi\in\Pi(sd)} r^{*\pi}_{\mathrm{sd}}(n) \sum_{l\in L(\pi)} \left(x^l(n)\right)^m \le \sum_{\mathrm{sd}} a_{\mathrm{sd}}(n) \sum_{\pi\in\Pi(sd)} \hat{r}^{\pi}_{\mathrm{sd}} \sum_{l\in L(\pi)} \left(x^l(n)\right)^m$$

where $\hat{r}^{\pi}_{\mathrm{sd}}$ defines a static routing algorithm that stabilizes the network. Note that, for any routing algorithm $\mathbf{R}$

$$\sum_{\mathrm{sd}} a_{\mathrm{sd}}(n) \sum_{\pi\in\Pi(sd)} r^{\pi}_{\mathrm{sd}}(n) \sum_{l\in L(\pi)} \left(x^l(n)\right)^m = \sum_{l} \sum_{\pi\in\Pi(l)} \sum_{\mathrm{sd}} a_{\mathrm{sd}}(n) r^{\pi}_{\mathrm{sd}} \left(x^l(n)\right)^m.$$

Thus

$$\sum_{l} \sum_{\pi\in\Pi(l)} \sum_{\mathrm{sd}} a_{\mathrm{sd}}(n) r^{*\pi}_{\mathrm{sd}} \left(x^l(n)\right)^m \le \sum_{l} \sum_{\pi\in\Pi(l)} \sum_{\mathrm{sd}} a_{\mathrm{sd}}(n) \hat{r}^{\pi}_{\mathrm{sd}} \left(x^l(n)\right)^m.$$

Since the network traffic $\boldsymbol{\rho}$ is admissible, under $\hat{\mathbf{R}}$ the average load on every link must be strictly less then 1. Let u_M be the maximum link utilization under $\hat{\mathbf{R}}$. Thus

$$\sum_{\pi\in\Pi(l)} \sum_{\mathrm{sd}} \rho_{\mathrm{sd}} \hat{r}^{\pi}_{\mathrm{sd}} \le u_M < 1 \quad \forall l \in E. \tag{42}$$

We prove the stability of the min-congested-link algorithm defined by (41) by using the following Lyapunov function for an appropriate large m:

$$\mathcal{L}(\mathbf{X}) = \sum_{l} \frac{(x^l)^{m+1}}{m+1} = \sum_{l} \frac{(x^l)^m}{m+1} \sum_{\pi\in\Pi(l)} x^{\pi}.$$

Since all the polynomial moments of flow sizes are finite, inequality (8) holds. As a consequence, we have just to show that the Lyapunov drift is negative when the queue size becomes large, i.e., that (9) also holds. Let us now compute the Lyapunov function drift. We define $\Delta x^l(n) = x^l(n+1) - x^l(n)$

$$\begin{aligned}
&\mathcal{L}(\mathbf{X}(n+1)) - \mathcal{L}(\mathbf{X}(n)) \\
&= \sum_{l} \frac{\left(x^l(n+1)\right)^{m+1}}{m+1} - \sum_{l} \frac{\left(x^l(n)\right)^{m+1}}{m+1} \\
&= \sum_{l} \frac{\left(x^l(n+1)\right)^{m+1} - \left(x^l(n)\right)^{m+1}}{m+1} \\
&= \sum_{l} \frac{\left(x^l(n) + \Delta x^l(n)\right)^{m+1} - \left(x^l(n)\right)^{m+1}}{m+1} \\
&= \sum_{l} \left(x^l(n)\right)^m \Delta x^l(n) + o\left(\|\mathbf{X}(n)\|^m\right).
\end{aligned}$$

Noticing that $\Delta x^l(n) = x^{\pi}(n+1) - x^{\pi}(n) = \sum_{\mathrm{sd}} a_{\mathrm{sd}}(n) r^{*\pi}_{\mathrm{sd}}(n) - w^{\pi}(n)$, we get

$$\begin{aligned}
&E\left[\mathcal{L}(\mathbf{X}(n+1)) - \mathcal{L}(\mathbf{X}(n)) \,|\mathbf{X}(n)\right] \\
&= \sum_{l} \sum_{\pi\in\Pi(l)} \left[\sum_{\mathrm{sd}} \rho_{\mathrm{sd}} r^{*\pi}_{\mathrm{sd}}(n) - w^{\pi}(n)\right] \left(x^l(n)\right)^m \\
&\quad + o\left(\|\mathbf{X}(n)\|^m\right).
\end{aligned}$$

When $\|\mathbf{X}(n)\| > B$, since the scheduling policy is (δ, ϵ) efficient, for each $\epsilon > 0$ there exists a $\delta > 0$ such that

$$w^l(n) = \sum_{\pi\in\Pi(l)} w^{\pi}(n) > 1 - \epsilon \quad \forall l \in J^*_{\delta}(n) \tag{43}$$

Moreover, there exists a m such that:

$$\left(\frac{x^l}{\max_{l'} x^{l'}}\right)^m < \frac{1}{B} \quad \forall l \notin J^*_{\delta}(n). \tag{44}$$

Thus

$$\begin{aligned}
&\sum_{l} \sum_{\pi\in\Pi(l)} \left[\sum_{\mathrm{sd}} \rho_{\mathrm{sd}} r^{*\pi}_{\mathrm{sd}}(n) - w^{\pi}(n)\right] \left(x^l(n)\right)^m \\
&= \sum_{l\in J^*_{\delta}} \sum_{\pi\in\Pi(l)} \left[\sum_{\mathrm{sd}} \rho_{\mathrm{sd}} r^{*\pi}_{\mathrm{sd}}(n) - w^{\pi}(n)\right] \left(x^l(n)\right)^m \\
&\quad + \sum_{l\notin J^*_{\delta}} \sum_{\pi\in\Pi(l)} \left[\sum_{\mathrm{sd}} \rho_{\mathrm{sd}} r^{*\pi}_{\mathrm{sd}}(n) - w^{\pi}(n)\right] \left(x^l(n)\right)^m \\
&< \sum_{l\in J^*_{\delta}} \left[\sum_{\pi\in\Pi(l)} \sum_{\mathrm{sd}} \rho_{\mathrm{sd}} \hat{r}^{\pi}_{\mathrm{sd}} - (1-\epsilon)\right] \left(x^l(n)\right)^m \\
&\quad + \sum_{l\notin J^*_{\delta}} \sum_{\pi\in\Pi(l)} \left[\sum_{\mathrm{sd}} \rho_{\mathrm{sd}} \hat{r}^{\pi}_{\mathrm{sd}} - w^{\pi}(n)\right] \left(x^l(n)\right)^m \\
&\le \sum_{l\in J^*_{\delta}} (u_M - 1 + \epsilon) \left(x^l(n)\right)^m \\
&\quad + \sum_{l\notin J^*_{\delta}} \left(u_M - w^l(n)\right) \left(x^l(n)\right)^m \\
&< \sum_{l\in J^*_{\delta}} (u_M - 1 + \epsilon) \left(x^l(n)\right)^m + u_M \sum_{l\notin J^*_{\delta}} \left(x^l(n)\right)^m \\
&< -\sum_{l\in J^*_{\delta}} 2\epsilon \left(x^l(n)\right)^m + u_M \sum_{l\notin J^*_{\delta}} \left(x^l(n)\right)^m
\end{aligned}$$

where u_M was defined in (42), and the last inequality holds for any choice of ϵ such that $3\epsilon < 1 - u_M$. We note that the second term can be made strictly smaller in module than the first term for sufficiently large B in (44). Indeed, by choosing $B > (|E|/\epsilon)$, it results

$$\sum_{l\notin J^*_{\delta}} \left(x^l(n)\right)^m < \epsilon \max_{l} \left(x^l(n)\right)^m < \epsilon \sum_{l\in J^*_{\delta}} \left(x^l(n)\right)^m.$$

■

B. Proof of Theorem 9

This proof is very similar to the proof of Theorem 6.

Let $x^l(n) = \sum_{\pi \in \Pi(l)} x^{l,\pi}(n)$. Also in this case, there exists a sufficiently large constant m_0 such that the min-congested-link routing algorithm can be defined, for $m > m_0$, as

$$\mathbf{R}^*(n) = \arg\min_{\mathbf{R}} \sum_{\mathrm{sd}} a_{\mathrm{sd}}(n) \sum_{\pi \in \Pi(sd)} r^{\pi}_{\mathrm{sd}} \sum_{l \in L(\pi)} \left(x^l(n)\right)^m \tag{45}$$

from which, clearly

$$\sum_{\mathrm{sd}} a_{\mathrm{sd}}(n) \sum_{\pi \in \Pi(sd)} r^{*\pi}_{\mathrm{sd}}(n) \sum_{l \in L(\pi)} \left(x^l(n)\right)^m \leq \sum_{\mathrm{sd}} a_{\mathrm{sd}}(n) \sum_{\pi \in \Pi(sd)} \hat{r}^{\pi}_{\mathrm{sd}} \sum_{l \in L(\pi)} \left(x^{l,\pi}(n)\right)^m$$

where $\hat{r}^{pi}_{\mathrm{sd}}$ defines a static routing algorithm that stabilizes the network. In addition

$$\sum_{l} \sum_{\pi \in \Pi(l)} \sum_{\mathrm{sd}} a_{\mathrm{sd}}(n) r^{*\pi}_{\mathrm{sd}} \left(x^l(n)\right)^m \leq \sum_{l} \sum_{\pi \in \Pi(l)} \sum_{\mathrm{sd}} a_{\mathrm{sd}}(n) \hat{r}^{\pi}_{\mathrm{sd}} \left(x^l(n)\right)^m .$$

Considering the Lyapunov function already applied in the proof of Theorem 6

$$\mathcal{L}(\mathbf{X}) = \sum_{l} \frac{(x^l)^{m+1}}{m+1} = \sum_{l} \sum_{\pi \in \Pi(l)} x^{l,\pi} \frac{(x^l)^m}{m+1}$$

we get

$$E\left[\mathcal{L}\left(\mathbf{X}(n+1)\right) - \mathcal{L}\left(\mathbf{X}(n)\right) | \mathbf{X}(n)\right] = \sum_{l} \sum_{\pi \in \Pi(l)} \left[\sum_{\mathrm{sd}} \rho_{\mathrm{sd}} r^{*\pi}_{\mathrm{sd}}(n) - w^{l,\pi}(n)\right] \left(x^l(n)\right)^m + o\left(\|\mathbf{X}(n)\|^m\right).$$

Now, since the scheduling policy is (δ, ϵ) efficient, for each $\epsilon > 0$ there exists a $\delta > 0$ such that

$$w^l(n) = \sum_{\pi \in \Pi(l)} w^{l,\pi}(n) > 1 - \epsilon \quad \forall l \in J^*_{\delta}(n). \tag{46}$$

In addition, for any large $B > 0$, there exists an m such that

$$\left(\frac{x^l}{\max_{l'} x^{l'}}\right)^m < \frac{1}{B} \quad \forall l \notin J^*_{\delta}. \tag{47}$$

Thus, following the same scheme of Theorem 6, being u_M the maximum link utilization under $\hat{\mathbf{R}}$, we obtain

$$E\left[\mathcal{L}\left(\mathbf{X}(n+1)\right) - \mathcal{L}\left(\mathbf{X}(n)\right) | \mathbf{X}(n)\right] < -\sum_{l \in J^*_{\delta}} 2\epsilon \left(x^l(n)\right)^m + u_M \sum_{l \notin J^*_{\delta}} \left(x^l(n)\right)^m$$

where the last inequality holds for any choice of ϵ such that $3\epsilon < 1 - u_M$. From which, proceeding as in the proof of Theorem 6, we obtain the result. ■

REFERENCES

[1] J. L. Sobrinho, "Algebra and algorithms for QoS path computation and hop-by-hop routing in the internet," *IEEE/ACM Trans. Netw.*, vol. 10, no. 4, pp. 541–550, Aug. 2002.

[2] J. L. Sobrinho, "Network routing with path vector protocols: Theory and applications," in *Proc. ACM SIGCOMM 2003*, Karlsruhe, Germany, Aug. 2003, pp. 40–60.

[3] G. Apostolopoulos, R. Guérin, S. Kamat, and S. K. Tripathi, "Quality of service based routing: A performance perspective," in *Proc. ACM SIGCOMM 1998*, Vancouver, Canada, Sep. 1998, pp. 17–28.

[4] A. Shaikh, J. Rexford, and K. Shin, "Load-sensitive routing of long-lived IP flows," in *Proc. ACM SIGCOMM 1999*, Cambridge, MA, Aug. 1999, pp. 215–226.

[5] G. Apostolopoulos, D. Williams, S. Kamat, R. Guerin, A. Orda, and T. Przygienda, "QoS routing mechanisms and OSPF extensions," RFC 2676, Aug. 1999.

[6] L. Fratta, M. Gerla, and L. Kleinrock, "The flow deviation method—An approach to the store-and-forward communication network design," *Networks*, vol. 3, pp. 97–133, 1973.

[7] Z. Wang, Y. Wang, and L. Zhang, "Internet traffic engineering without full mesh overlaying," in *Proc. IEEE INFOCOM 2001*, Anchorage, AK, Apr. 2001, vol. 1, pp. 565–571.

[8] A. Sridharan, R. Guérin, and C. Diot, "Achieving near-optimal traffic engineering solutions for current OSPF/IS-IS networks," *IEEE/ACM Trans. Netw.*, vol. 13, no. 2, pp. 234–247, Apr. 2005.

[9] A. Medina, N. Taft, K. Salamatian, S. Bhattacharya, and C. Diot, "Traffic matrix estimation: Existing techniques and new directions," in *Proc. ACM SIGCOMM*, Pittsburgh, PA, Aug. 2002, pp. 161–174.

[10] Z. Wang and J. Crowcroft, "QoS routing for supporting multimedia applications," *IEEE J. Sel. Areas Commun.*, vol. 14, no. 7, pp. 1228–1234, Sep. 1996.

[11] Q. Ma, P. Steenkiste, and H. Zhang, "Routing high-bandwidth traffic in max-min fair share networks," in *Proc. ACM SIGCOMM 1996*, Stanford, CA, Aug. 1996, pp. 206–217.

[12] C. Casetti, R. Lo Cigno, M. Mellia, M. Munafò, and Z. Zsoka, "A new class of QoS routing strategies based on network graph reduction," *Comput. Netw.*, vol. 41, no. 4, pp. 475–487, Feb. 2003.

[13] A. Basu, A. Lin, and S. Ramanathan, "Routing using potentials: A dynamic traffic-aware routing algorithm," in *Proc. ACM SIGCOMM 2003*, Karlsruhe, Germany, Aug. 2003, pp. 37–48.

[14] E. Rosen, A. Viswanathan, and R. Callon, "Multi-protocol label switching architecture," RFC 3031, Jan. 2001.

[15] S. P. Meyn and R. Tweedie, *Markov Chain and Stochastic Stability*. New York: Springer-Verlag, 1993.

[16] M. J. Neely, E. Modiano, and C. E. Rohrs, "Dynamic power allocation and routing for time varying wireless networks," in *Proc. IEEE INFOCOM 2003*, San Francisco, CA, Apr. 2003, pp. 745–755.

[17] S. Kunniyur and R. Srikant, "End-to-end congestion control schemes: Utility functions, random losses and ECN marks," *IEEE/ACM Trans. Netw.*, vol. 11, no. 5, pp. 689–702, Oct. 2003.

[18] D. Katabi, M. Handley, and C. E. Rohrs, "Congestion control for high bandwidth-delay product networks," in *Proc. ACM SIGCOMM 2002*, Pittsburgh, PA, Aug. 2002, pp. 92–102.

[19] J. Mo and J. Walrand, "Fair end-to-end window-based congestion control," *IEEE/ACM Trans. Netw.*, vol. 8, no. 5, pp. 556–567, Oct. 2000.

[20] A. K. Parekh and R. G. Gallager, "A generalized processor sharing approach to flow control in integrated services networks—The single node case," *IEEE/ACM Trans. Netw.*, vol. 13, pp. 344–357, Jun. 1993.

[21] M. Bramson, "Instability of FIFO queueing networks," *Ann. Appl. Probab.*, vol. 4, pp. 414–431, 1994.

[22] R. J. Williams, "An invariance principle for semimartingale reflecting brownian motions in an orthant," *Queueing Syst.: Theory Appl.*, vol. 30, pp. 5–25, 1998.

[23] M. Bramson, "State space collapse with application to heavy traffic limits for multiclass queueing networks," *Queueing Syst.: Theory Appl.*, vol. 30, pp. 26–44, 1998.

[24] H. C. Gromoll, "Diffusion approximation for a processor sharing queue in heavy traffic," *Ann. Appl. Prob.*, vol. 14, pp. 555–611, 2004.

[25] E. Leonardi, M. Mellia, M. Ajmone Marsan, and F. Neri, "Joint optimal scheduling and routing for maximum network throughput," in *Proc. IEEE INFOCOM 2005*, Miami, FL, Jun. 2005, pp. 819–830.

Emilio Leonardi (M'99) was born in Cosenza, Italy, in 1967. He received the Dr.Ing degree in electronics engineering and the Ph.D. degree in telecommunications engineering from the Politecnico di Torino, Torino, Italy, in 1991 and 1995, respectively.

He is currently an Associate Professor in the Dipartimento di Elettronica, Politecnico di Torino. In 1995, he visited the Computer Science Department of the University of California, Los Angeles (UCLA); in summer 1999 he joined the High Speed Networks Research Group, Bell Laboratories/Lucent Technologies, Holmdel, NJ; in summer 2001, he joined the Electrical Engineering Department of the Stanford University and, finally in summer 2003, he joined the IP Group at Sprint, Advanced Technologies Laboratories, Burlingame, CA. His research interests are in the fields of high-speed switching architectures, performance evaluation, and QoS routing algorithms.

Marco Mellia (M'97) was born in Torino, Italy, in 1971. He received the degree in electronic engineering and the Ph.D degree in telecommunications engineering from the Politecnico di Torino, Torino, Italy, in 1997 and 2001, respectively.

From March to October 1999 he was with the Computer Science Department, Carnegie Mellon University, Pittsburgh, PA, as a Visiting Scholar. Since April 2001, he is with the Electronics Department, Politecnico di Torino, as an Assistant Professor. His research interests are in the fields of all-optical networks, traffic measurement and modeling, and QoS routing algorithms.

Dr. Mellia has participated in the program committees of several conferences including IEEE INFOCOM, IEEE GLOBECOM, and IEEE ICC.

Marco Ajmone Marsan (F'99) received the Laurea degree in electronic engineering from the Politecnico di Torino, Torino, Italy, in 1974 and the M.S.E.E. from the University of California, Los Angeles (UCLA), in 1978.

He is a Full Professor at the Electronics Department, Politecnico di Torino, Torino, Italy. He is the Director of the Institute for Electronics, Information and Telecommunications Engineering of the National Research Council since September 2002. He is the Vice-Rector for Research, Innovation, and Technology Transfer at Politecnico di Torino since November 2005. He was at Politecnico di Torino Electronic Department from November 1975 to October 1987—first as a Researcher and then as an Associate Professor. He was a Full Professor at the Computer Science Department, University of Milan, Milan, Italy, from November 1987 to October 1990. During the summers of 1980 and 1981, he was with the Research in Distributed Processing Group, Computer Science Department, UCLA. During the summer of 1998, he was an Erskine Fellow at the Computer Science Department, University of Canterbury, New Zealand.

Dr. Marsan received the Best Paper Award at the Third International Conference on Distributed Computing Systems in Miami, Florida, in 1982. In 2002, he was awarded a Honoris Causa Degree in Telecommunications Networks from the Budapest University of Technology and Economics. He is a corresponding member of the Academy of Sciences of Torino. He participates in a number of Editorial Boards of international journals, including the IEEE/ACM TRANSACTIONS ON NETWORKING. He is listed by ISI among the highly cited researchers in Computer Science.

Fabio Neri (M'98) is a Full Professor at the Electronics Department, Politecnico di Torino, Torino, Italy.

He leads a research group on optical networks at Politecnico di Torino. He has recently been involved in several European projects on WDM networks. He is currently the coordinator of the FP6 Network of Excellence e-Photon/ONe on optical networks, which involves 40 European institutions. He coordinated the participation of his research group to several national Italian research projects. His research interests are in the fields of performance evaluation of communication networks, high-speed and all-optical networks, packet switching architectures, discrete event simulation, and queuing theory.

Dr. Neri has served on several IEEE conferences and journals. He has participated on the technical program committees of several conferences, including IEEE INFOCOM and IEEE Globecom. He serves on the Editorial Board of IEEE/ACM TRANSACTIONS ON NETWORKING, and is Co-Editor-in-Chief of the *Elsevier Optical Switching and Networking Journal.*