EUROPEAN TRANSACTIONS ON TELECOMMUNICATIONS
Eur. Trans. Telecomms. 2008; **19**:421–442
Published online 10 April 2008 in Wiley InterScience
(www.interscience.wiley.com) DOI: 10.1002/ett.1285

Stochastic network utility maximisation – a tribute to Kelly's paper published in this journal a decade ago

Yung Yi and Mung Chiang*

Department of Electrical Engineering, Princeton University, NJ 08544, USA

SUMMARY

Since the seminal work by Kelly on distributed network resource allocation using the language of network utility maximisation (NUM) a decade ago, there have been extensive research efforts generalising and applying NUM to model, analyse and design various network protocols and architectures. Some of these works combine the distributed optimisation approach with stochastic network models to study NUM under network dynamics occurring at the session, packet and constraint levels. We survey these works by presenting the key questions, results and methodologies in this emerging theory of stochastic network utility maximisation, followed by discussion on related work and future research challenges. Copyright © 2008 John Wiley & Sons, Ltd.

1. INTRODUCTION

In the 1997 paper in this journal [1] and the 1998 paper [2], Kelly *et al.* presented an innovative idea on network resource allocation that has led to many research activities since. An optimisation problem is formulated where the variables are the source rates constrained by link capacities and the objective function captures design goals:

$$\begin{array}{ll} \text{Maximise} & \sum_i U_i(x_i) \\ \text{Subject to} & \boldsymbol{Rx} \leqslant \boldsymbol{c} \end{array} \qquad (1)$$

where the source rate vector $\boldsymbol{x}$ is the set of optimisation variables, one for each of the sources indexed by i, the $\{0, 1\}$ routing matrix $\boldsymbol{R}$ and link capacity vector $\boldsymbol{c}$ are constants, and $U_i(\cdot)$ is the utility function of source i. Because the above network utility maximisation (NUM) problem can be readily decomposed, distributed algorithms are developed where each of the links and sources controls its local variable, such as link price or source rate, based on local observables, such as link load or path price. By techniques such as Lyapunov function or the descent lemma, global or local asymptotic convergence towards the optimum can be proved for these distributed algorithms, for cases with or without propagation delay. A key insight is that the effects of network protocols can be understood as the trajectories of a controlled dynamic system.

After a decade of work by many researchers, there is now a substantial set of theory, algorithms, applications and even commercialisation based on the NUM model of networks.

First, NUM and its extensions can be used to model various resource allocation problems and network protocols. Utility functions may depend on rate, latency, jitter, energy, distortion, etc., may be coupled across users, and may be any non-decreasing function, although most papers assume smooth and concave utility functions. They can be constructed based on user behaviour model, operator cost model or traffic elasticity model. They can also shape the fairness of resource allocation: a maximiser of the α-fair utility functions satisfies the definition of α-fair allocation. Here α-fair utility function refers to a family of functions parameterised by $\alpha \geqslant 0$: $U^{\alpha}(\cdot) = (\cdot)^{1-\alpha}/(1-\alpha)$ for $\alpha \geqslant 0$, $\alpha \neq 1$ and $\log(\cdot)$, for $\alpha = 1$ [3], and a feasible allocation $\boldsymbol{x}$ is called α-fair if, for any other feasible allocation $\boldsymbol{y}$, $\sum_s (y_s - x_s)/x_s^{\alpha} \leqslant 0$. The notion of α-fairness includes maxmin fairness, proportional fairness, throughput maximisation as

* Correspondence to: Mung Chiang, Department of Electrical Engineering, Princeton University, NJ 08544, USA. E-mail: chiangm@princeton.edu

Accepted 22 January 2008

special cases, and it is often believed that larger α means more fairness.

Sometimes, network protocols are modelled by NUM *via* 'reverse-engineering': a given protocol, originally designed based on engineering intuitions, is shown to be implicitly solving an underlying optimisation problem. Insights thus obtained on why the protocol works well, or not so well, can then guide a systematic design of future versions of the protocol. Indeed, one of the first applications of NUM was to show that Internet congestion control in Transmission Control Protocol (TCP) implicitly solves a NUM problem where the variant of TCP dictates the exact shape of the utility function.

Second, the basic NUM formulation can be generalised to incorporate other degrees of freedom, e.g. routes, schedules, transmit powers, contention probabilities, channel codes and modulations, with constraint set and objective function modified accordingly. In particular, the constraint set can reflect physical constraints, technological and economic constraints and inelastic quality of service (QoS) constraints by individual users. These generalised NUM formulations provide a starting point to generate distributed resource allocation algorithms for wireline and wireless networks alike.

Furthermore, cross-layer interactions may be characterised and layered protocol stacks designed by viewing the process of 'layering', i.e. the modularisation and distribution of network functionalities into layers or network elements, as decomposition of a given NUM problem into many subproblems. These subproblems are 'glued together' by certain functions of the primal and dual variables. This framework of 'layering as optimisation decomposition' [4] has been developed by many researchers over the last several years for joint control of congestion, routing, scheduling, random access, transmit power, code and modulation, etc. It has also amplified the benefits of mechanisms such as back-pressure scheduling and network coding in the entire protocol stack. Alternatives of decomposing the same NUM formulation in different ways further lead to the opportunities of enumerating and comparing alternative protocol stacks. The theory of decomposition of NUM thus becomes a foundation to understand, in a conceptually simple way, the complexities of network architectures: 'who does what' and 'how to connect them'.

Despite the range of research activities above, an important aspect of NUM models remain challenging. In the basic NUM (1) and the associated solutions, it is often assumed that the user population remains static, with each user carrying an infinite backlog of packets that can be treated as fluid, injected into a network with static connectivity and time-invariant channels. While these assumptions make the resulting model tractable, they also compromise the accuracy of the model and limit the applicability of the results. In many applications, users arrive with a finite workload and depart after finishing it. Each packet generated by users goes through a sequence of events such as random arrival and probabilistic dropping and the network consists of time-varying channels with dynamic topology and uncontrolled traffic. Will the results of NUM theory remain valid and the conclusions maintain predictive power under these stochastic dynamics? Can new questions arising out of stochastic models also be answered? This paper surveys the results over the last 8 years on these two questions.

An analogy can be drawn with Shannon's seminal work in 1948 [5]. By turning the focus from the design of finite-block length codes to the regime of infinite-block length codes, thus enabling the Law of Large Numbers to take effect, Shannon's work provided architectural principles (e.g. source-channel-separation) and established fundamental limits (e.g. channel capacity) in his mathematical theory of communication. Since then, the complicating issues associated with the design of practical codes and with finite block length have been brought back into information theory. Some of the principles and design guidelines from the 1948 papers remain unchanged while other have been modified.

A similar development has happened in networking over the past decade. By turning the focus from coupled queuing dynamics to an optimisation problem based on deterministic fluid models, thus enabling the views of decomposition and distributed control, Kelly's work [1,2] leads to our current study on architectural principles in networks and a mathematical foundation for network protocol design. However, the complicating issues associated with stochastic network dynamics need to be brought back into such study, possibly modifying some of the principles and design guidelines based on NUM.

Incorporation of stochastic network dynamics into the generalised NUM formulation often leads to challenging models for those working in either stochastic network theory or distributed optimisation algorithms. As an example to be further explained later, in establishing stability of NUM under session arrivals and departures, service rates of queues are determined by a distributed solution to NUM, while the parameters of NUM formulations are in turn stochastically time-varying according to the states of the queues.

Just like control theory that has been connected with NUM *via* feedback system models, or economics theory that has been connected with NUM *via* pricing-based supply-demand models, stochastic network theory is being connected with NUM *via* the problems summarised in this survey.

The variety of these research problems will be grouped into three major categories: session level, packet level and constraint level. After a brief overview in this section, each category will be discussed in a separate section, before related work and open problems are presented in the remaining part of the paper.

Session level. Sessions[†] dynamically arrive with finite workload and depart after finishing the workload, rather than holding infinite backlog and staying in the network forever. In such a session-level dynamic model, researchers have studied the stochastic stability region achieved by allocating resources through NUM. Among the results surveyed in Section 2, a typical one is that the stability region of α-fair allocation, for any α within a certain range, is equivalent to the constraint set if it is convex and time-invariant, which is also the largest possible stability region that can be obtained by any resource allocation. This means that satisfying the constraints in a deterministic formulation is both necessary and sufficient for session-level stability. However, researchers have also found that different models of arrival process, utility functions, constraint sets and timescale separation might lead to a dependence of stability region on α, and a gap between achievable stability region by NUM and the maximum stability region.

Necessary and sufficient conditions for stability of NUM under the most general assumptions remain an open problem, as are most of the questions on the distribution of queue length and end-to-end delay induced by session arrivals and NUM resource allocation.

Packet level. Packets typically arrive in bursts, rather than in a smooth fluid, due to packet-level protocols that determine the exact pattern of packet generation. Upon arrival, packets go through a sequence of events including probabilistic marking and dropping. Queues in downstream nodes are shaped by the queuing actions in upstream nodes. In addition, flows obeying congestion control interact with uncontrolled flows, such as user datagram protocol (UDP)-based multimedia traffic and short-lived 'mice' traffic, which are often modelled as stochastic processes independent of the NUM variables.

We group the above issues into one large category of study referred to as stochastic dynamics at packet level. Among the associated studies on the impact of microscopic dynamics on macroscopic properties include the following topics: (i) validity of deterministic-fluid-based NUM under packet-level dynamics, (ii) the effects of randomness due to uncontrolled flows on performance of flows controlled by NUM (iii) translation of application-layer characteristics in hypertext transfer protocol (HTTP), Peer-To-Peer (P2P), and Internet-Protocol-TV (IPTV) and (iv) impact of stochastic noisy feedback on distributed solutions of NUM.

Constraint level. In both wireline and wireless networks, the constraint set of NUM may be time-varying for a variety of reasons, e.g. fading in wireless channel, failure of links, and mobility, sleeping mode or battery depletion of nodes. Variations in the constraint set offer both the challenge to prove stability and optimality of the existing algorithms, and the opportunity to gain benefits by opportunistic transmission and scheduling. We focus on the first set of issues in this survey.

A different and major class of problems with constraint-level dynamics is to maximise utility subject to stability, where the constraint set is a general convex set representing the stability region.

Obvious from the brief introduction above, the range of questions raised under the general term of 'stochastic network utility maximisation' is in fact very broad. There are also many situations where dynamics at more than one level must be modelled in the same problem formulation. We will discuss these combinations after presenting each levels of dynamics individually.

Timescale separation. There are at least four timescales involved in the formulation of stochastic network utility maximisation. Understanding the combination of timescales is often important. Let T_s represent the timescale of session-level dynamics, e.g. the inter-arrival time of sessions. Let T_p represent that of packet-level dynamics, e.g. user think-time of a web-browsing model. Let T_c represent that of constraint-level dynamics, e.g. coherence time of wireless channel. Finally, let T_r represent that of resource allocation algorithm, often iterative and distributed, solving a NUM problem, e.g. round-trip time divided by the rate of convergence of the algorithm. By saying a timescale is 0, we mean it is infinitesimally small: the variations of the network dynamics smooth out or convergence of the algorithm instantaneously achieved. Each combination of timescale assumptions can be represented by a string of inequalities, e.g. $0 = T_p \approx T_r << T_s \approx T_c$ means that the packet level details can be ignored, the algorithm converges instantaneously fast compared to either the session arrivals or the constraint variations, which takes place at similar timescales and cannot be ignored. In

[†] Sometimes also called users, flows or connections.

each of the problem formulations to be surveyed, there is an underlying assumption, or a set of assumptions all reasonable to the application, of timescale separation as represented by similar string of inequalities.

Timescale separation often leads to more tractable model and stronger analysis results, but we also have to be aware of the danger of imposing unreasonable timescale separations that are too inaccurate for the resulting conclusions to be useful.

Notation. We use standard $\mathbb{R}^N$ and $\mathbb{R}^N_+$ for the N-dimensional real and non-negative Euclidean spaces, respectively. Generally, we use the calligraphic font $\mathcal{S}$ to refer to a set, and the bold-face fonts $\boldsymbol{x}$ and $\boldsymbol{X}$ to refer to a vector and a matrix, respectively.

2. SESSION-LEVEL DYNAMICS

2.1. *Basic framework of session-level stability*

Traffic demands. Consider a network where sessions are randomly generated by users and cease upon completion. Sessions are classified according to the set of resources required to transfer the corresponding packets. For example, in wired networks with fixed routing, the class of a session is defined by the set of links that the session traverses from the source to the destination, i.e. session's path. We have a finite set $\mathcal{S}$ of S classes of sessions.

Suppose that sessions of class s are generated according to a Poisson process of intensity λ_s sessions per second. The file sizes of class-s sessions are i.i.d. exponentially distributed with mean size $1/\mu_s$ bits. We will discuss the cases for general arrival and file-size distributions in Subsection 2.2.4. Denote the traffic intensity of sessions of class s by $\rho_s = \lambda_s/\mu_s$ bit/s.

Network state. At time t, the network state is denoted by $\boldsymbol{N}(t) = (N_1(t), \ldots, N_S(t))$, where $N_s(t)$ is the number of active class-s sessions. $\{\boldsymbol{N}(t)\}_{t=0}^{\infty}$ is a stochastic process governed by the random arrivals and departures of sessions.

Rate region. The constraint set $\mathcal{R}$ is the set of achievable resource vectors $\boldsymbol{\phi} = (\phi_1, \ldots, \phi_S)$ where ϕ_s is the total rate allocated to class-s sessions. Since rate allocation was the first application of NUM, sometimes people refer to the constraint set of a NUM problem as the rate region, even when the problem involves other degrees of freedom beyond rates. We assume here that the rate region does not depend on the network state $\boldsymbol{N}$. The form of the rate regions is diverse and can be either fixed or time-varying, depending on the system model and resource allocation algorithms. In this subsection, we focus on the fixed rate regions of various shapes. We will discuss time-varying rate regions later in Section 5.

Resource allocation by NUM. Resource allocation algorithms allocate network resources to different session classes according to the current network state $\boldsymbol{N}(t)$, the utility function and the rate region. Resource allocation based on NUM is the solution of the following optimisation problem:

$$\begin{array}{ll} \text{maximise} & \sum_s N_s(t) U_s(\phi_s/N_s(t)) \\ \text{subject to} & \boldsymbol{\phi} \in \mathcal{R} \end{array} \tag{2}$$

where the utility functions U_s are assumed, throughout this paper, as twice differentiable and concave, although non-smooth and non-concave utility functions [6, 7] have also been studied for deterministic NUM. Sometimes researchers further assume that all session classes share the same utility function, i.e. $U_s = U$ for all s. Figure 1 depicts the framework of session-level dynamics research.

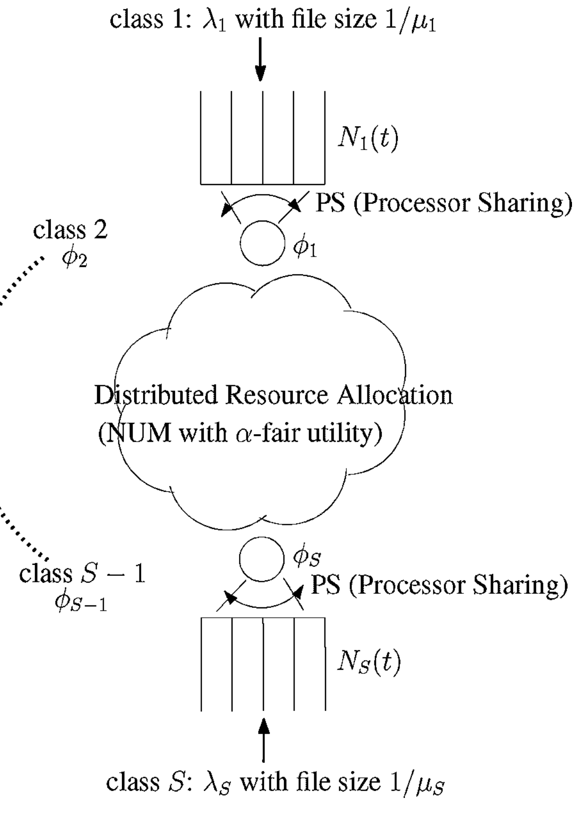
class 1: λ_1 with file size $1/\mu_1$
$N_1(t)$
PS (Processor Sharing)
ϕ_1
class 2
ϕ_2
Distributed Resource Allocation
(NUM with α-fair utility)
class $S-1$
ϕ_{S-1}
ϕ_S
PS (Processor Sharing)
$N_S(t)$
class S: λ_S with file size $1/\mu_S$

Figure 1. Distributed resource allocation by solving NUM under a dynamic user population with session arrivals. Sessions departure time is shaped by file size as well as service rate, i.e. the resource allocation vector $\{\phi_s\}$ obtained by solving NUM, where each session in class s receives resource $\phi_s/N_s(t)$.

Two different timescale assumptions are used: either with timescale separation: $0 = T_{\mathrm{r}} << T_{\mathrm{s}}$, or without timescale separation: $T_{\mathrm{r}} \approx T_{\mathrm{s}}$.

Session level stability. A main research focus about session-level dynamics is to prove necessary and sufficient conditions for session-level stability. There are many notions of stability, two of which will be used in this section.

Definition 2.1. (Queue stability). *A queueing network under a resource allocation algorithm is said to be stable-in-the-mean* (*or, queue stable, or, simply, stable*) [8], *if*

$$\limsup_{t\to\infty} \frac{1}{t} \int_{\tau=0}^{t} E\left[\sum_{s\in\mathcal{S}} N_s(\tau)\right] < \infty$$

Intuitively, queue stability means that the number of active sessions in the system remains finite almost surely. From the Little's law, the mean duration time of a session also remains finite almost surely.

A weaker notion of stability is sometimes used.

Definition 2.2. (Rate stability). *A queueing network under a resource allocation policy is said to be rate stable if, for every class s,* $\lim_{t\to\infty} D_s(t)/t = \lambda_s$, *where* $D_s(t)$ *is the number of class s sessions that finished the service.*

In a general network topology, it is challenging to prove the queue or rate stability of NUM-based resource allocation. It is a multi-class queueing network with service rates dependent on the solution to NUM, which is in turn dependent on the number of active flows. A technique often used is due to Dai [9], where a suitable scaling of the original system, called fluid-limit scaling, can provide a way of proving such stability results. We will later discuss fluid-limit and describe how it helps with the proof of session-level stability.

Definition 2.3. (Stability region).

(i) *The stability region of a resource allocation algorithm is defined as the set of traffic intensity vectors* $\boldsymbol{\rho} = (\rho_1, \dots, \rho_S)$ *for which session-level stability is achieved under the resource allocation algorithm.*

(ii) *The maximum stability region is defined by the union of the stability regions achieved by all possible resource allocations.*

As an 'outer bound' or 'converse result', the maximum stability states that for any traffic intensity vector outside this set, there exists no resource allocation algorithm that can stabilise the network at session-level. Note that a resource allocation achieving the maximum stability region may not be utility-maximisation-based or implementable in a distributed fashion.

If we assume Poisson arrival and exponential file size (i.e. workload), the system can be modelled by a Markov chain, whose network dynamics evolve in the following way: for each class s,

$$N_s(t) \rightarrow N_s(t) + 1, \quad \text{with rate } \lambda_s$$
$$N_s(t) \rightarrow N_s(t) - 1, \quad \text{with rate } \mu_s \phi_s(\boldsymbol{N}(t))$$

Then, mathematically, stability means that the process $\{\boldsymbol{N}(t)\}_{t=0}^{\infty}$ is ergodic. The session-level stability is now equivalent to the positive recurrence of the Markov process $\boldsymbol{N}(t)$ under simple technical conditions on aperiodicity and irreducibility.

2.2. Session-level stability results

The existing research results on session-level stability are summarised in Table 1, with different network topologies,

Table 1. Summary of main results on session-level stability.

Work	Arrival, file size dist.	Topology	Rate regions	U_i	U shape (i.e. α)
[10]	Poisson, exponential	General	Convex	Same	$\alpha = 1, \alpha \to \infty$
[11]	Poisson, exponential	General	Convex	Diff.	General
[12, 13]	Poisson, exponential	General	Convex	Same	$\alpha \geqslant 1$
(Fast timescale)					
[14]	General, exponential	General	Convex	Diff.	General
[15]	General, general	General	Convex	Same	$\alpha \to \infty$
[16]	General, phase type	2×2 grid	Convex	Same	$\alpha = 1$
[17]	General, phase type	General	Convex	Same	$\alpha = 1$
[18]	General, general	Tree	Convex	Same	General
[19]	General, general	General	Convex	Diff.	$\alpha \to 0^+$
(Rate stability)					
[20]	Poisson, exponential	General	Non-convex	Diff.	General
			Time-varying convex	Diff.	General
Open problem	General, general Fast timescale	General	Convex, non-convex Time-varying	Diff.	General, non-concave

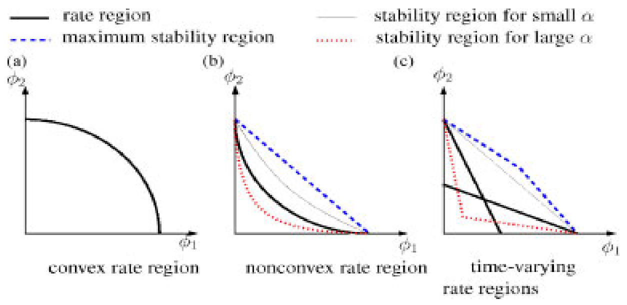

Figure 2. Illustration of session-level stability region for different shapes of rate region. In the convex case, all four curves coincide in many models of file arrivals and utility functions, which means that stability region for NUM is independent of α and is the maximum one that can be achieved by any resource allocation. This is no longer the case for non-convex rate region (Subsection 2.2.3) or time-varying rate regions (Subsection 5.3). (a) Convex rate region, (b) non-convex rate region, (c) time-varying rate-regions.

shapes of rate region and utility function and arrival process and file size distributions. They are illustrated in Figure 2. Results for α-fair utility can be readily extended to general concave utility functions whose ratio of the first and second derivatives follow a similar order of growth in x as the α-fair utility. The case for time-varying rate regions will be discussed in Section 5.

2.2.1. *Polytope and general convex rate region*

The first analysis of session-level stability focuses on wired networks with fixed routing, supporting data traffic only [10, 11]. For such networks, the rate region is a polytope formed by the intersection of a finite number of linear capacity constraints, i.e.

$$\mathcal{R} = \{\boldsymbol{\rho} | \boldsymbol{R}\boldsymbol{\rho} \leqslant \boldsymbol{c}\}$$

For this rate region, assuming timescale separation, it is shown that all α-fair allocations with $\alpha > 0$ provide session-level stability if and only if the vector representing the traffic intensities of session classes lie in the rate region. In other words, the rate region in the α-fair utility maximisation problem is also the stability region under session-level stochastic dynamics. This stability region is also the maximum stability region.

A key proof technique is to use the fluid-limit, defined by the following:

$$\boldsymbol{n}(t) = \lim_{\omega \to \infty} \frac{\boldsymbol{N}(\omega t)}{\omega}$$

Then, the system dynamics at the fluid-limit regime is provably represented by

$$\frac{\mathrm{d}}{\mathrm{d}t} n_s(t) = \lambda_s - \mu_s \phi(\boldsymbol{n}(t))$$

This fluid-limit scaling of the original system makes the limiting system deterministic rather than random, which facilitates the proof of stability, guided by Theorems 2.1 and 2.2.

Now we describe how the fluid-limit scaling can be used to prove session-level stability in Definitions 2.1 and 2.2.

Theorem 2.1. *The system is queue-stable if, for $\boldsymbol{n}(t)$ with $\|\boldsymbol{n}(0)\| \leqslant 1$, there exists a $T < \infty$, such that $\forall t \geqslant T$, $\boldsymbol{n}(t) = \boldsymbol{0}$.*

Theorem 2.2. *The system is rate-stable if, for $\boldsymbol{n}(t)$ with $\boldsymbol{n}(0) = 0$, we have $\boldsymbol{n}(t) = \boldsymbol{0}$, $\forall t \geqslant 0$.*

Note the difference between queue and rate stability. Queue stability requires that for non-negative initial value, the number of active sessions in the fluid-limit should become empty within a finite time and remain empty. Rate stability requires that for zero initial value, the number of active session continue to be zero.

Combining with the above technique, the stability result of α-fair allocations can be proved by choosing a suitable Lyapunov function $V(\boldsymbol{n}(t))$, for example,

$$V(\boldsymbol{n}(t)) = \sum_{s \in \mathcal{S}} \mu_s^{-1} \rho_s^{-\alpha} \frac{(n_s(t))^{\alpha+1}}{\alpha + 1}$$

An immediate extension is to allow a general convex rate region. An example is the rate region in wireless system with the TDMA scheduling, where the rate region is the convex hull of the possible rate point that can be allocated instantaneously. For a convex rate region $\mathcal{R}$, it is proved in Reference [21] that the rate region is also the stability region for α-fair allocations, $\alpha > 0$.

2.2.2. *No time-scale separation assumption*

In some networks, session level stochastic may operate on a faster timescale, with the arrive-and-depart process of sessions varying constantly. Hence, the assumption on instantaneous convergence of the resource allocation algorithm, with respect to session arrivals, may not hold. Session-level stability without the timescale separation assumption is studied in References [12, 13].

The work in Reference [12] considers a dual congestion controller used at the sources, which forms part of a distributed solution of NUM. Time is divided into discrete slots of length T, indexed by k. Then, the system dynamics with respect to a class s are given by

$$\phi_s(t) = \phi_s(kT) = \min \left\{ \Big(\sum_{l \in L} R_{ls} q^l(kT) \Big)^{\frac{-1}{\alpha}}, M_s \right\} \quad (3)$$

where

$$q^l((k+1)T) = \left[q^l(kT) + \epsilon_l \left(\sum_{s \in \mathcal{S}} R_{ls} \int_{kT}^{(k+1)T} N_s(t)\phi_s(kT)\mathrm{d}t - Tc_l \right) \right] \quad (4)$$

where R_{ls} is the (l, s)th entry of the routing matrix of $\boldsymbol{R}$, ϵ_l is the step-size used at the link l, and M_s is the upper-bound of transmission rate of class s. Note that the session-level dynamics induces change of $N_s(t)$ in Equation (4), which is in turn reflected in the congestion control in Equation (3). It is proved in Reference [12] that, for $\alpha \geqslant 1$, sufficiently small step-size ϵ_l leads to the maximum stability region of α-fair allocations. The case for $\alpha < 1$ is unclear.

We briefly summarise a key methodology in the proof. Recall that for the case for timescale separation earlier, the fluid-limit of the original system is derived, and then it is showed that such a fluid-limit is stable, leading to stability of the original system. However, without timescale separation, the fluid-limit of the original system represented by the dynamics of Equations (3) and (4) cannot be derived for technical reasons. Instead, the authors directly find a Lyapunov function and to use the Foster's criterion in the proof of stability. The difficulty of finding an appropriate fluid-limit of the original system is tackled differently in Reference [13] where an upper-bound on the quantity $N_s(t)\phi_s(t)$ is assumed. Then, the fluid-limit can be established, but at the expense of assuming such a bound.

2.2.3. *General non-convex rate region*

There are also many practical scenarios in which the rate regions are non-convex, usually due to the limited capability of the underlying resource allocation algorithm. For example, random access may lead to continuous but non-convex rate region, and quantized levels of parameter control lead to discrete, thus non-convex, rate region.

For non-convex rate regions, it is generally impossible to derive an explicit and exact stability condition. This is mainly due to the fact that the stability condition depends on detailed statistical characteristics of the session arrival processes, as well as the session departure processes determined by the solutions of a non-convex optimisation problem. Some papers provide bounds on the stability region for specific topologies under particular allocations, see e.g. [22, 23]. Some other papers aim at providing exact stability conditions: in [24–26], a recursive (with respect to the number of session classes) stability condition is given for a particular class of networks, including those studied in References [22, 23].

In networks with only two classes of sessions and a discrete rate region, the stability condition of a large class of allocations, including α-fair allocation, is characterised in Reference [27]. The stability region of α-fair allocation, for $\alpha > 0$ is the smallest coordinate convex set [‡] containing the contour of $\mathcal{R}^\alpha$, where $\mathcal{R}^\alpha$ is the set of rate vectors that could be chosen by the α-fair allocation. In other words, $\mathcal{R}^\alpha$ corresponds to the set of rate vectors, say $\boldsymbol{r}$, which is a subset of the system rate region $\mathcal{R}$, such that there exists a state $\boldsymbol{N}$ with $\boldsymbol{\phi}(\boldsymbol{N}) = \boldsymbol{r}$.

Recently, the authors in Reference [20] characterise sufficient and necessary conditions for session-level stability for $\alpha > 0$ in networks with an arbitrary number of classes over a discrete rate region. In this case, there exists a gap between the necessary and sufficient conditions for stability. However, they show that these conditions coincide when the set of allocated rate vectors are continuous, leading to an explicit stability condition for network with continuous non-convex rate regions, summarised as follows. For continuous $\mathcal{R}^\alpha$, the stability region of α-fair allocation is the smallest coordinate-convex set containing $c(\mathcal{R}^\alpha)$, where $c(\mathcal{Y})$ denotes the smallest closed set containing $\mathcal{Y}$. Note that now the stability region varies for different values of α.

We conclude this section by introducing a resource allocation algorithm that achieves the maximum stability region in an arbitrary fixed rate region. This policy is called max-projection (MP) allocation [28]. For a given network state $\boldsymbol{N}$, the MP allocation allocates rates that solves the following optimisation problem:

$$\begin{aligned} &\text{Maximise} \quad \textstyle\sum_s N_s \phi_s \\ &\text{Subject to} \quad \boldsymbol{\phi} \in \mathcal{R} \end{aligned} \quad (5)$$

However, this allocation is not utility-based, thus it does not guarantee fairness of resource allocation. There also may not be a distributed implementation of this kind of allocation.

[‡] A set $\mathcal{Y} \subset \mathbb{R}^n_+$ is said to be coordinate-convex when the following is true: if $\boldsymbol{b} \in \mathcal{Y}$, then for all $\boldsymbol{a} : 0 \leqslant \boldsymbol{a} \leqslant \boldsymbol{b}, \boldsymbol{a} \in \mathcal{Y}$.

2.2.4. *General arrival and general file size distribution*

Another important and challenging extension of the basic stability result is to remove the unrealistic assumption on Poisson arrivals and exponential file size distribution, since these characterisations have been proved false in many network measurement studies. For general arrivals, proving stability conditions becomes much more difficult than the case of assuming exponential file size distribution. Reduction to positive recurrence of Markov chain no longer works, and keeping track of the residual file size is not a scalable proof technique. New fluid limit needs to be established and new Lyapunov functions constructed.

Using the technique in References [9, 14] relaxes the assumption of Poisson arrivals, by studying a general stationary and a bursty network model. In Reference [29], a fluid model is formulated for exponentially distributed workload to study the 'invariant states' as an intermediate step for obtaining diffusion approximation for all $\alpha \in (0, \infty)$. In Reference [18], the fluid model is established for α-fair rate allocation, $\alpha \in (0, \infty)$, under general distributional condition on arrival process and service distribution. Using this fluid model, they have obtained characterisation of 'invariant states', which led to stability of network under α-fair allocation, $\alpha \in (0, \infty)$, when the network topology is a tree.

For general network topologies, three recent works have tackled this difficult problem of stochastic stability under general file size distribution, for different special cases of utility functions; Reference [15] establishes stability for max–min fair (corresponding to $\alpha \to \infty$) rate allocation, and Reference [17] establishes stability for proportional fair (corresponding to $\alpha = 1$) rate allocation for Poisson arrival and phase-type distribution. Using the fluid model in Reference [18] but under a different scaling, Reference [19] establishes the rate stability of α-fair allocation for general file size distribution for a continuum of α: α sufficiently close to (but strictly larger than) 0, and a partial stability results for any $\alpha > 0$ fair allocation policy. It is also proved that α-fair allocation is rate stable over convex rate region scaled down by $1/(1 + \alpha)$, for all $\alpha > 0$, general topology and possibly different utility functions for different users. The general problem of session-level stability remains open.

3. PACKET-LEVEL DYNAMICS

Packet-level actions take place in a network by various means: (i) uncontrolled flows such as real-time traffic and short-lived 'mice' traffic, (ii) burstiness of packet arrivals due to application protocols, (iii) probabilistic packet marking and dropping in active queue management (AQM) algorithms and (iv) other randomness residing in the protocol implementation and feedback signals. In studying the effect of these packet-level random factors on the network, most of the research investigate the system in the asymptote of large-scale network.

3.1. ***NUM with packet-level randomness***

In Reference [30], the authors prove that stochastic delay-difference equations, which model primal-based congestion control algorithm under stochastic noise, converge to a deterministic functional differential equation. In other words, the trajectory of the average rate (over flows) at the router converges to that of the deterministic model with the noise replaced by its mean. Mathematically, this is given by

$$\dot{x}(t) = \kappa(\omega - x(t-d)p(x(t-d) + a))$$

where $x(t)$ is the average transmission rates, a is the mean of stochastic noise, d is the round-trip delay, κ and ω are the constants of a primal algorithm achieving proportional-fair controller [2] and $p(\cdot)$ is the marking function at the router.

To enable tractable analysis and scalable simulation, many of the works on NUM use deterministic fluid approximations. The result in Reference [30] provides an evidence of confidence that such a deterministic approximation can be valid, at least for log utility functions over a single-link with homogeneous delay. The convergence to a deterministic fluid differential equation occurs over a finite time horizon, and convergence for the infinite time horizon is also proved under additional technical conditions.

The work in Reference [30] is extended to 'TCP-like' controllers in Reference [31]. The TCP-like controller corresponds to a rate-based controller that resembles the congestion avoidance phase of TCP. The authors in Reference [31] still assume a single link with homogeneous delay, but their results are weaker than those in Reference [30] in the sense that the convergence occurs only 'asymptotically' in the number of flows and in time, as opposed to the path-wise convergence in Reference [30]. The asymptotic convergence means that the equilibrium transmission rate of each flow in the scaling system converges to the fixed point solution of the deterministic differential equation. The authors in Reference [31] show that the global stability criterion for the deterministic system based on a differential equation is also a global stability condition for the stochastic system with multiple flows. This leads to implication that the parameter design (for stability) can be studied based on the deterministic version of the model.

Other works also provide justifications for the use of a deterministic feedback system model. Both Reference [32] and [33] use stochastic models in their papers to capture the randomness by mice flows, probabilistic marking in AQM and marking quantization errors.

In Reference [33], the authors characterise queue fluctuations using Central Limit Theorem based approximation to include the variance properties of the router queues in the analysis. This paper proves that, as the number of flows becomes large, the AQM queue dynamics can be accurately approximated by a sum of a deterministic process and a stochastic process. Further, the recursion of the deterministic process depends only on the capacity of the bottleneck link and the expected traffic arrival, which help justify the use of a deterministic feedback system model to study the expected queue behaviour.

On the other hand, in Reference [32], a detailed stochastic model is presented for N TCP Reno sources sharing a single bottleneck link with capacity Nc implementing random early detection (RED). They show that as the number of sources and the link capacity both increase linearly, the queue process converges to a deterministic process described by differential equations as usually assumed in the congestion control literature. Even though these results are proved only for a single bottleneck node, they provide some justification for the popular deterministic fluid model by suggesting that the deterministic process is the limit of a scaled stochastic process as the number of flows and link capacities scale to infinity.

One of the sources of randomness in the Internet is the set of real-time flows that require a certain QoS, e.g. packet loss probability. The authors in Reference [34] examine the effect of congestion control mechanism on the QoS of real-time flows. In particular, they study the relation between 'aggressiveness' of marking functions at the intermediate router and the achieved QoS for real-time flows. The aggressiveness, parameterised by some constant, is captured in the name of elasticity of marking functions, which intuitively corresponds to the slope of marking function at the equilibrium point. The following two tradeoffs are studied:

(1) Stability-elasticity tradeoff. The more elastic the marking function is, the better is the QoS guarantee of real-time flows. However, this also leads to a smaller maximum allowable end-to-end delay that guarantees stability [35].
(2) Scheduling-elasticity tradeoff. Given the equilibrium point of controlled flows and the QoS for real-time flows, two queueing mechanisms are considered: first-in-first-out (FIFO) and priority. In priority queueing, there are two separate queues for controlled and real-time flows, and the absolute priority is given to real-time flows, as long as there are packets to serve in that queue. The authors show that one can achieve the same QoS performance for the real-time flows with the simpler queueing of FIFO as that with more complex queueing of priority, by employing more aggressive marking functions.

In the above analysis, there are two different timescales to consider: one at the routers and another at the sources. The first option is to view the system at the timescale of sources. This timescale is N times 'slower' than that of the router dynamics, since for every packet transmitted by an end-user, there are N packets that arrive at the intermediate router. Over any fixed time interval (i.e. time is not scaled), one can view the queue as a multiplexer over a large number of random, open-loop flows (e.g. real-time flows), leading to a deterministic limit in the large number of flows regime. The research in References [30, 31, 34] mainly considers this end-system time-scale. In References [30, 31], the authors study the fluid-limit of the system from the viewpoint of the controlled flows, and the open-loop flows appear as a constant rate process, with the rate equal to the expected value of the uncontrolled arrival process. On the other hand, the work in Reference [34] tries to model the behaviour of an open-loop end-user, at the timescale of the end-system. It considers the tail probability for open-loop flows as the performance metric of a real-time flow using large deviation techniques.

The other option is to look at the system behaviour from the viewpoint of the router. As the number of flows and router capacity increase, the timescale of queue becomes faster. In this regime, by appropriately 'slowing-down' time at the router, it is shown that the uncontrolled arrival processes can be modelled by a Poisson process [36], and there is much work [37–39] in understanding the queueing dynamics at this timescale in routers. Understanding the behaviour of router queues is also crucial to buffer-sizing questions, i.e. what amount of buffer space is needed to achieve sufficient multiplexing gain. We will discuss the buffer-sizing issues later in Subsection 6.3.

3.2. Application-layer burstiness

Another aspect of packet-level stochastic models is to understand the effect of application-layer burstiness on NUM-based congestion control at the transport layer. For example, in Reference [40], the authors consider a single link with capacity Nc (bits) shared by N HTTP flows consisting of J classes, where the number of class j connections is $\theta_j N$, $\theta_1 + \cdots + \theta_J = 1$. Flow class j alternates between think times and transfer times with different mean think time, $\bar{I}_j$, and average amount of data, $\bar{B}_j$, during transfer times. During the period of a think time, a flow does not require any bandwidth from the link. Immediately after a period of think time, the source starts to transmit a random amount of data by a TCP connection. The transfer time depends on the amount of download data and the bandwidth allocation to this flow by TCP.

Let $Q_j(t)$ and $X_j(t)$ be the number of class j connections and the amount of bandwidth received by a class j connection during transfer times, respectively. Then, $X_j(t)$ is allocated by solving the following optimisation problem

$$\begin{aligned} &\text{maximise} && \sum\nolimits_{j=1}^{J} Q_j(t) U_j(X_j(t)) \\ &\text{subject to} && \sum\nolimits_{j=1}^{J} Q_j(t) X_j(t) = Nc \end{aligned} \quad (6)$$

The number of active flows is random, but at any time, the active flows share the link capacity according to a transport layer resource allocation algorithm as in Equation (6). Let

$$\rho_j = \frac{\bar{B}_j}{\bar{I}_j}, \quad Y_j(t) = \frac{Q_j(t) X_j(t)}{N\theta_j}$$

i.e. ρ_j is the load of connection j, and $Y_j(t)$ is the average bandwidth received by class j connections at time t. Then,

$$\lim_{N\to\infty} Y_j(t) = y_j^*, \quad \text{a.s.}$$

where y_j^* is the solution of

$$\begin{aligned} &\text{maximise} && \sum\nolimits_{j=1}^{J} \theta_j \hat{U}_j(y_j) \\ &\text{subject to} && \sum\nolimits_{j=1}^{J} \theta_j y_j = c \end{aligned}$$

where

$$\hat{U}_j(y_j) = \frac{\rho_j - y_j}{\rho_j} U_j\left(\frac{y_j \rho_j}{\rho_j - y_j}\right)$$

Furthermore,

$$\lim_{N\to\infty} X_j(t) = x_j^*, \quad \text{a.s.}$$

where

$$x_j^* = \frac{\rho_j y_j^*}{\rho_j - y_j^*}$$

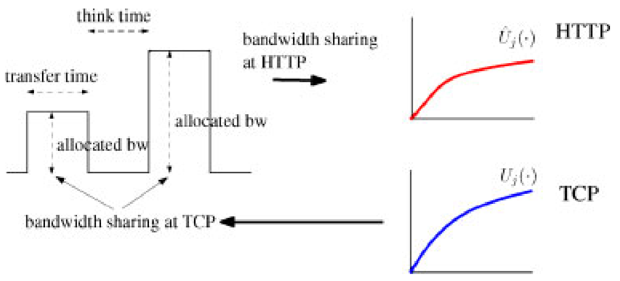

Figure 3. Characterising bandwidth sharing in HTTP [40], with a combination of transfer time and think time, through utility function $\hat{U}$, which is induced by TCP rate allocation through NUM with utility function U.

The above results state that the average throughput y_j^*, i.e. the throughput aggregated over active flows of each type normalised by the total number of flows of that type, also solve a utility maximisation problem with an induced utility functions $\hat{U}_j(\cdot)$ at the transport layer (see Figure 3).

3.3. *Stochastic noisy feedback*

In the research on distributed implementation of the NUM problem, mostly feedbacks are assumed to be perfectly communicated among the network elements. However, in practice, perfect feedback is impossible, mainly due to probabilistic marking and dropping, contention-induced loss of packets, limited size of messages, etc. In Reference [41], the authors study the impact of stochastic noisy feedback on distributed algorithms, where they first consider a primal-dual algorithm in the following general form

$$\begin{aligned} x_s(t+1) &= \left[x_s(t) + \epsilon(t)(L_{x_s}(\boldsymbol{x}(t), \boldsymbol{\lambda}(t)))\right]_{\mathcal{D}} \\ \lambda_l(t+1) &= \left[\lambda_l(t) + \epsilon(t)(L_{\lambda_l}(\boldsymbol{x}(t), \boldsymbol{\lambda}(t)))\right]_0^{\infty} \end{aligned}$$

where x_s is the source transmission rate of session s, λ_l is the shadow price, L is gradient update, ϵ is step size and $[\cdot]_{\mathcal{D}}$ is the projection onto the feasible set $\mathcal{D}$.

Noisy feedback adds noise to the gradients L_{x_s} and L_{λ_l}, which become stochastic. Two cases are considered for analysis:

(1) Unbiased feedback noise. When the gradient estimator is unbiased, it is established, *via* a combination of the stochastic Lyapunov Stability Theorem and local analysis, that the iterates generated by distributed NUM algorithms converge with probability one to an optimum, under some standard technical conditions. In

general, the limit process of the interpolated process based on the normalised iterate sequence is a stationary reflected linear diffusion process, not necessarily a Gaussian diffusion process.

(2) Biased feedback noise. In contrast, when the gradient estimator is biased, the iterates converge to a contraction region around the optimal point, provided that the biased terms are asymptotically bounded by a scaled version of the true gradients. These results confirm those derived based on deterministic models of feedback with errors [42, 43].

The primal-dual algorithm has only a single timescale in terms of source and link updates. However, there exist various decomposition methods, leading to different distributed implementations and also different timescales of updates by various network elements. Also in Reference [41], the authors study a primal-decomposition based algorithm with multiple timescale algorithms. For two-timescale algorithm, with faster timescale for source rate update and shadow price update and slower timescale with link contention probabilities, convergence under noisy feedback is established. In the case when the gradient estimator at the faster time scale is unbiased and at the slower time scale is biased, the algorithm converges to a contraction region. When comparing alternative decompositions with different timescales, robustness to stochastic dynamics can be used as one of the metrics.

4. CONSTRAINT-LEVEL DYNAMICS

4.1. Overview

Consider a constrained queueing system with exogenous arrivals that are not infinitely backlogged but follows a certain stochastic process with finite mean. In Reference [44], Tassiulas and Ephremides developed a control policy, called throughput-optimal policy, that maximises the arrival rates subject to queue stability.

Now consider the case where the arrivals are random but outside the stability region, and a certain amount of utility is defined for transmission of a unit volume of data. Combining the NUM framework with stability requirement is generally non-trivial, due to the fact that rate regions are difficult to characterise by distributed controllers, and can be time-varying in some networks. This section discusses this research topic that is often referred to as 'utility maximisation subject to stability'.

The general problem formulation in this topic is given by the following optimisation problem

$$\begin{aligned} \text{Maximise} \quad & \sum\nolimits_s U_s(\bar{x}_s) \\ \text{Subject to} \quad & \bar{\boldsymbol{x}} \in \Lambda \end{aligned} \tag{7}$$

where $\bar{\boldsymbol{x}}$ is the long-term rate vector, averaged over instantaneous rate $x(\tau)$ at time τ, i.e.

$$\bar{x}_s = \lim_{t\to\infty} \frac{1}{t} \int_{\tau=0}^{t} x(\tau)$$

and Λ is the maximum stability region. It is the set of arrival rates for which there exists a control policy to stabilise the system, and may not be known to the distributed controllers.

The problem formulation is similar to Equation (2). But the formulation represented by Equation (7) assumes random traffic arrivals. Furthermore, here Λ is either the 'average' of the instantaneous rate regions for time-varying systems or the convex hull of discrete rate sets. In such a case, it is required to develop a control mechanism that is unaware of Λ, the statistics of the arrivals, or the time-variations of the instantaneous rate region.

If the stochastic arrivals are inside the stability region, it suffices to stabilise the system, since the policy stabilising the system also achieves the maximum utility under a reasonable assumption on monotonicity of utility functions. Therefore, the problems considered in the rest of this section assumes that the arrivals are outside of the stability region. Later, in Subsection 4.4, we will mention a different model, where even for the arrivals inside the stability region, stabilising the system is not equal to maximising the achieved utility.

4.2. Wireless cellular networks

Utility maximisation under random channel fluctuations is first studied for cellular networks [45–48], with an objective of achieving proportional fairness. This is generalised in Reference [49] by to other forms of utility functions.

Consider a discrete, slotted-time model in a wireless cellular system with multiple users, where the channel conditions are time-varying, following a certain stochastic process $(\boldsymbol{S}(t))_{t=0}^{\infty}$. Note that $\boldsymbol{S}(t)$ at time t is a vector of the channel state seen by each mobile user, and assumed to hold over one timeslot. This means that the timescale of channel variations is equivalent to that of packet-level resource allocation policy. If this time-varying channel process has a stationary distribution π, then the steady-state average rate region is given by

$$\Lambda \triangleq \sum_{\text{channel state } i} \pi_i \mathcal{R}_i \tag{8}$$

where π_i is the stationary probability that the instantaneous rate region $\mathcal{R}(t)$ is the ith rate region $\mathcal{R}_i$.

The optimisation problem (7) can be solved using the standard stochastic gradient algorithm, from which the algorithm can be described by choosing the rate allocation vector $\boldsymbol{x}(t)$, such that

$$\boldsymbol{x}(t) \in \arg \max_{\boldsymbol{r} \in \mathcal{R}(t)} \nabla U(\hat{\boldsymbol{x}}(t))^T \boldsymbol{r} \tag{9}$$

Here, $\hat{\boldsymbol{x}}(t)$ is the moving averaged service rates, updated as follows:

$$\hat{\boldsymbol{x}}(t+1) = \hat{\boldsymbol{x}}(t) + \epsilon(t)(\boldsymbol{x}(t) - \hat{\boldsymbol{x}}(t))$$

where $\epsilon(t)$ is the step size at time t.

It is proved that this algorithm is asymptotically optimal, i.e. when $\epsilon(t) \to 0$, the algorithm in Equation (9) converges to an optimum of (7). In spite of the asymptotic optimality with respect to the utility, there are situations where the algorithm in Equation (9) cannot stabilise the system for any constant step-size $\epsilon(t) = \epsilon$, for random arrival inputs that are within the maximum stability region [50].

This case of maximising the long-term aggregate utility over time-varying channels shows that maximising utility may not guarantee system stability on its own. We need a different control policy to achieve utility maximisation subject to system stability, which we summarise in the next subsection in a more general system model.

4.3. *Wireless ad hoc networks*

In this subsection, we survey the resource allocation research [51–56] that maximises aggregate utility subject to stability through joint congestion control, scheduling and routing in wireless *ad hoc* networks.

Similar to wireless cellular systems in the previous section, the channel is again assumed to be fixed within a discrete timeslot but changes randomly and independently across slots. We denote by $\mathcal{R}(t)$ as the link-level rate region at t. The end-to-end throughput-region Λ (i.e. the constraint set in Equation (7)) is given by the set of end-to-end transmission rates $\boldsymbol{x}$, such that at every node, flow conservation is satisfied with the link capacities in the average link rate region $\sum_{\text{channel state i}} \pi_i R_i$. To avoid notational confusion, we remark that Λ in Equation (8)

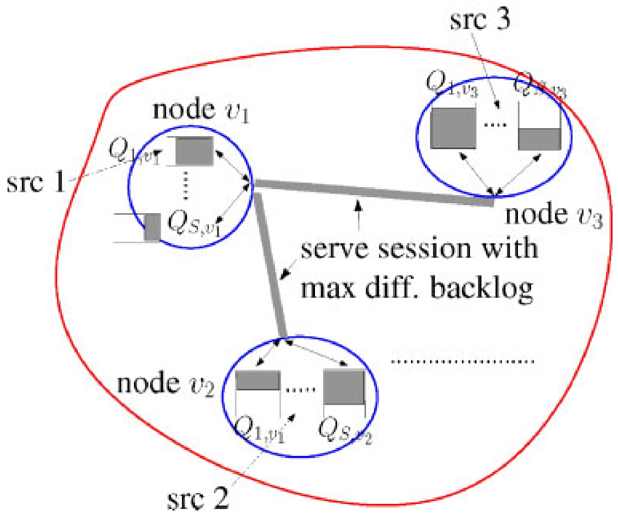

Figure 4. Illustration of joint congestion control, routing and scheduling, through a back-pressure based algorithm that solves the NUM in node-based representation.

corresponds to both the end-to-end throughput-region and the average link rate region, since it is a single-hop cellular system.

There are two types of formulations, and the associated distributed algorithms, for utility maximisation subject to stability: (i) node-based [51, 53, 55, 56] and (ii) link-based ones [52]. In a node-based algorithms, which we focus here, each node installs separate queues for each session, and sources adapt their transmission rates based on the congestion price at each source node itself. As will be described shortly, this price essentially corresponds to the queue length for the source node. Node-based algorithms can operate based on only the price at each source node because of the 'back-pressure' mechanism: the congestion prices at an intermediate nodes are indirectly transferred to the source in an hop-by-hop manner. In contrast, in a link-based algorithm, each source controls its rates based on the aggregate sum of prices over its path. Technically, link-based algorithm works only when routing is fixed. In the problem of joint congestion control, scheduling, and routing, node-based algorithms are more suitable, but at the expense of maintaining per-session queues at each node. Another advantage of node-based algorithms is that it does not need to assume that the arrival rates are instantaneously available at each nodes, unlike the link-based algorithms.

As illustrated in Figure 4, the node-based back-pressure algorithms have three different components: (i) source congestion control, (ii) routing and (iii) scheduling. Scheduling determines the rate schedule assigned to be links, and routing deals with how to share the scheduled rate over the sessions on each link. We first describe routing and scheduling common to all the algorithms, and then elaborate on the slight differences on source congestion control.

- **Routing:** At timeslot t, on each link l, serve the session that has the maximum differential backlog, i.e.

$$s_l^*(t) = \arg\max_{s\in S}(Q_{s,\mathrm{tx}(l)}(t) - Q_{s,\mathrm{rx}(l)}(t)) \tag{10}$$

  where tx(l) and rx(l) are the transmitter and the receiver of a link l, respectively, and $Q_{s,v}(t)$ denotes the length of the session s queue at node v at slot t. Let $Q_l(t)$ be the differential backlog of session $s_l^*(t)$, i.e.

$$Q_l(t) = \max_{s\in S}(Q_{s,\mathrm{tx}(l)} - Q_{s,\mathrm{rx}(l)})$$

- **Scheduling:** At timeslot t, choose the link rate schedule $\boldsymbol{r}(t)$ that maximises the aggregate weight, i.e.

$$\boldsymbol{r}(t) = \max_{\boldsymbol{r}\in\mathcal{R}(t)} \sum_{l\in L} Q_l(t) r_l \tag{11}$$

  Then, the service rate $r_l(t)$ is applied to serve the session s_l^* over link l.

It is easy to solve the problem in Equation (10) using only local information. However, the problem in Equation (11) is generally hard to implement, and may not be solvable in a distributed manner with reasonably low complexity, unless special structures of the rate region $\mathcal{R}(t)$ is specified and exploited. We will further discuss low complexity, distributed implementation of link scheduling in Subsection 6.4.

For source rate control, there are dual-controller [51, 53, 55] and primal-dual controller [54, 56, 57]. These two types of congestion controllers are first studied in the wired Internet congestion control (see [58–60] for details). Briefly speaking, a dual controller consists of a gradient-type algorithm for shadow price updates and a static source rate algorithm, i.e. two different timescales for shadow price updates and source rate updates. In the primal-dual algorithm, the source rates and the shadow prices are updated at the same timescale.

4.3.1. Dual source controller

There are two forms of dual controller algorithms, which turn out to be essentially the same: 1) using queue lengths directly, and 2) using shadow prices.

(1) *Using queue length directly:* In Reference [53], it has been proposed that each session s performs the following:

$$x_s(t) = \arg\max_{y_s}(VU_s(y_s) - y_s Q_s(t)) \tag{12}$$

where V is a constant parameter, and $Q_s(\cdot)$ is the queue length of the source node of session s. Differentiating Equation (12), the source rate for session s are determined by

$$x_s(t) = U_s'^{-1}(Q_s(t)/V) \tag{13}$$

(2) *Using price:* In Reference [55], the source controller adjusts its transmission rate by computing

$$x_s(t) = U_s'^{-1}(\lambda_s(t)) \tag{14}$$

where $\lambda_s(t)$ is the aggregated shadow congestion price value over the link paths of the session s, and obtained by dual decomposition of the original problem in Equation (7).

In Reference [53], the timeslot length (i.e. timescale of source-rate update) is fixed, and the system is parameterised by V. On the other hand, in Reference [55], the timescale of source-rate update is scaled by the step-size.

By choosing sufficiently large V (or sufficiently small step-size), the achieved utility, denoted by $\bar{U}$, can be made arbitrarily close to the optimal utility U^*. The scaling law on the difference between $\bar{U}$ and U^* is linear [53]

$$U^* - \bar{U} \leqslant O(1/V)$$

4.3.2. Primal-dual source controller

In this type of controller, the sources do not compute their transmission rate in one shot, instead, the transmission rates are also updated by the following equation [56]

$$x_s(t+1) = \left[x_s(t) + \gamma(KU_s'(x_s(t)) - Q_s(t))\right]_m^M$$

where γ is the parameter which determines sensitivity to queue length changes (i.e. rate of convergence), and K is the parameter which determines the distance between the optimal utility point and the achieved utility, similar to the parameter V in Reference [53]. Here, $[x]_m^M$ refers to the projection of x over the interval $[m, M]$.

The authors in References [54, 57] also study a similar primal-dual type algorithm, called greedy primal-dual (GPD), under a very general system model, motivated by the stochastic gradient algorithm in Subsection 4.2. For simplicity, by assuming separability of the utility function w.r.t. the source rate vector, at timeslot t, the source of session s chooses the transmission rates $x_s(t)$, such that

$$x_s(t) = \arg \max_{y_s \in \mathcal{Y}_s(t)} \left(U_s'(\hat{x}_s(t)) - \beta Q_s(t) \right) \times y_s$$

where the running average $\hat{x}_s(t)$ of $x_s(t)$ is updated by

$$\hat{x}_s(t+1) = (1-\beta)\hat{x}_s(t) + \beta x_s(t), 0 < \beta < 1$$

The set $\mathcal{Y}_s(t)$, which is time-varying, refers to the set of possible transmission rates of the session s at time t. The basic idea is to greedily maximise the first order increment of

$$\sum_s U_s(x_s) - \frac{1}{2}\beta \sum_s Q_s^2$$

For the algorithms introduced so far, stability and optimality are proved, where optimality can be adjusted by choosing the parameter appropriately, i.e. a parameter which determines the distance between the optimal utility point and the achieved utility in the algorithm mentioned so far (i.e. V in Reference [53], K in Reference [56] and the step-size in Reference [55]). The stability proof is based on the Markovian structure of the system and an appropriate Lyapunov function, e.g. $L(\boldsymbol{Q}(t)) = \sum_{s,v} Q_{s,v}^2(t)$ or $L(\boldsymbol{Q}(t)) = \sum_l Q_l^2(t)$.

However, there may be some cost for large K and V, such as delay [53, 61]. This tradeoff can be represented by

$$\text{average queue lengths} \leqslant O(f(V))$$
$$\text{achieved utility} \geqslant U^\star - O(1/V)$$

[61] shows that $f(V)$ can also be a logarithmic function, and this logarithmic tradeoff is the best utility-delay tradeoff with respect to the parameter V, under some additional technical assumptions.

4.4. *Other models*

In the last part of this section, we discuss a related research with a slightly different problem formulation, where even for arrivals inside the throughput-region, stabilising the system is not equivalent to optimising the utility.

In Reference [62], the authors consider a constrained queueing system with random arrivals. Consider a schedule set $\mathcal{S}$, where a link schedule $\boldsymbol{S} \in \mathcal{S}$ is a set of links that can be scheduled simultaneously. Each session is assumed to traverse a single-hop link, and there is no time-varying channel variations.

Denote by $\boldsymbol{S} \in \{0,1\}^L$ as a binary vector where $S_l = 1$ if link l is scheduled. We use the notation $l \in \boldsymbol{S}$, if $S_l = 1$. Consider an arrival rate vector inside the convex hull of $\mathcal{S}$, the achieved long-term utility is defined by

$$\liminf_{t\to\infty} \frac{1}{t} \sum_{\tau=1}^{t} \sum_{l \in L} U_l(\boldsymbol{S}(\tau)) S_l(\tau) \tag{15}$$

Different from Equation (7), in Equation (15), the total utility is the long-term average of the instantaneously achieved utility rather than the utility of the long-term average service rate. These two quantities are not equivalent unless the utility function is linear. Also, in Equation (15), the utility achieved by the link l is a function of the entire schedule. Due to these differences, even for the arrivals inside the throughput-region, control decision at each timeslot becomes different in order to maximise utility subject to stability; the controller chooses the schedule $\boldsymbol{S}^*$, such that

$$\boldsymbol{S}^* = \arg \max_{\boldsymbol{S} \in \mathcal{S}} \sum_{l \in \boldsymbol{S}} \left(Q_l(t) - V(U_l^{\max} - U_l(\boldsymbol{S})) \right) S_l \tag{16}$$

where V is a constant parameter, and $U_l^{\max}$ is a given constant and, the maximum possible utility of link l that can be achieved by any schedule.

5. COMBINATIONS OF MULTIPLE DYNAMICS

So far, we have surveyed the wide range of questions raised in stochastic network utility maximisation, under the three headings of session, packet and constraint-level models. Combinations of more than one type of stochastic models also naturally arise. We focus on the question of session-level stability of NUM with a time-varying constraint set in this section.

5.1. *Timescale possibilities*

Recall the notation for timescales in Section 1:

(1) T_s: session arrivals,
(2) T_c: constraint set variations,
(3) T_r: convergence of resource allocation algorithm.

We first assume a timescale separation between convergence in resource allocation algorithm and session-level dynamics, i.e. $T_\mathrm{r} << T_\mathrm{s}$, as assumed in most of Section 2. Then there are three possibilities of timescale separation across T_s, T_c and T_r, depending on the timescale of T_c.

(1) **Fast regime ($T_\mathrm{c} << T_\mathrm{r} << T_\mathrm{s}$).** This corresponds to fast variations of rate regions due to fast channel fluctuations. In this case, even resource allocation algorithm does not track resource variations, but only sees the

average of time-varying resource. The stochastic nature of resource variations are masked and invisible to resource allocation algorithms and session-level dynamics. Therefore, we will not discuss this case further.

(2) **Intermediate regime ($T_r \approx T_c << T_s$).** When the timescales of rate region variations and the resource allocation algorithms are similar, the resource allocation algorithms can harness the rate region variations, by allocating resource opportunistically. A typical example of such systems is channel-aware scheduling in cellular networks [47, 63], where fading variations of the channels are exploited to achieve more throughput.

(3) **Slow regime ($T_r << T_c \approx T_s$).** When the rate region variations are not that fast, and similar to the session-level dynamics, being opportunistic becomes more difficult and these variations can be exploited only at the expense of compromising delay perceived by sources. This regime is encountered due to, for example, slow mobility in wireless networks and link failures in wired networks.

5.2. Intermediate regime

In Reference [64], opportunistic scheduling is employed, i.e. giving higher priority to users experiencing better channel state. Due to this opportunism, total throughput across users also increases, as the number of users increases. Then, assuming that the entire total throughput is evenly distributed to the active users, the system can be modelled by a multi-class processor sharing queue, from which session-level stability and other related performance metrics such as mean duration time are analysed.

We mention the work of Lin and Shraff [12] in this Subsection, it studies the session-level stability at the intermediate regime in wireless multi-hop networks, but without timescale separation assumption between session-level dynamics and the resource allocation algorithms, i.e. $T_r \approx T_c \approx T_s$. It is proved that the maximum stability of α-fair allocation still holds for this case for $\alpha \geqslant 1$. The intuition and methods are similar to that discussed in Subsubsection 2.2.2.

5.3. Slow regime

Recently, [20] studies the session-level dynamics for slow regime ($T_c \approx T_s$), where time-varying rate regions $\{\mathcal{R}(t)\}_{t=0}^{\infty}$ are modelled by a stationary and ergodic process. Then, the entire system can be modelled by an augmented Markov chain with states $(\boldsymbol{N}(t), \mathcal{R}(t))$. We denote by $\boldsymbol{\pi}$ the stationary distribution of $\{\mathcal{R}(t)\}$, i.e. $\Pr\{\mathcal{R}(t) = \mathcal{R}_i\} = \pi_i$, $i \in \mathcal{I}$, for some index set $\mathcal{I}$.

They first characterise the maximum stability region, given by

$$\overline{\mathcal{R}} = \sum_{i \in \mathcal{I}} \pi_i \mathcal{R}_i \tag{17}$$

It is proved that this maximum stability region is achieved by the max-projection policy described in Equation (5). In summary, the max-projection policy achieves the maximum stability region, irrespective of the shape and time-variation of rate regions.

Next, to study the stability region of α-fair allocation, they develop a fluid-limit model of the augmented Markovian system, and characterise the stability region of α-fair allocations by this fluid-limit model.

The evolution of the system fluid limit is given by

$$\frac{\mathrm{d}n_s}{\mathrm{d}t} = \lambda_s - \mu_s \sum_{i \in \mathcal{I}} \pi_i \phi_s^{(i)}(\boldsymbol{n}), \quad \forall s \in \mathcal{S} \tag{18}$$

In contrast to the fluid model for a fixed rate region, the possible service rate for an α-fair allocation in the fluid limit is the average of the allocated rate vectors in the various rate regions. It is natural to examine the following set:

$$\partial\overline{\mathcal{R}_\alpha} = \left\{ \boldsymbol{\phi} : \exists \boldsymbol{n} \in \mathbb{R}_+^S, \boldsymbol{\phi} = \sum_i \pi_i \times \boldsymbol{\phi}^{(i)}(\boldsymbol{n}) \right\}$$

This is the set of all possible service rate vectors in the fluid limit. Define the average rate region in the fluid limit for the α-fair allocation as the smallest coordinate-convex set containing $\partial\overline{\mathcal{R}_\alpha}$, i.e.

$$\overline{\mathcal{R}_\alpha} = \{\boldsymbol{y} : \exists \boldsymbol{x} \in \partial\overline{\mathcal{R}_\alpha} \quad \text{s.t.} \quad 0 \leqslant \boldsymbol{y} \leqslant \boldsymbol{x}\} \tag{19}$$

The stability region for α-fair allocations is given by the following result: for all $\alpha > 0$, the stability region of the α-fair allocation for time-varying rate region is $\overline{\mathcal{R}_\alpha}$.

Similar to the case for non-convex rate regions, it is proved that stability region depends on α. The proof technique resembles that in non-convex rate regions, where the authors first use necessary and sufficient conditions for discrete time-varying rate regions, and prove that they coincide for continuous time-varying rate regions.

Even though the fluid model based stability region is not in closed form, it helps with deriving various properties and relations of stability regions in α-fair allocations. In particular, for two-class network, the authors characterise the pattern of dependence of stability region on α, by proving that there exists a tradeoff between fairness and

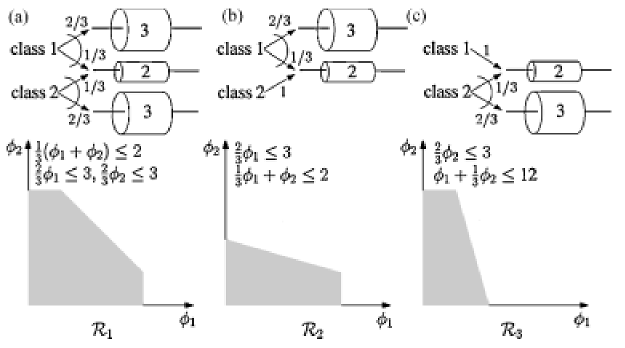
(a)
(b)
(c)
class 1
class 2
2/3
1/3
1
3
2
ϕ_2
ϕ_1
$\frac{1}{3}(\phi_1+\phi_2) \le 2$
$\frac{2}{3}\phi_1 \le 3, \frac{2}{3}\phi_2 \le 3$
$\frac{2}{3}\phi_1 \le 3$
$\frac{1}{3}\phi_1 + \phi_2 \le 2$
$\frac{2}{3}\phi_2 \le 3$
$\phi_1 + \frac{1}{3}\phi_2 \le 12$
$\mathcal{R}_1$
$\mathcal{R}_2$
$\mathcal{R}_3$

Figure 5. A wired network with link failures, allowing multi-path routing with flow-splitting.

stability, i.e. fairness can be enhanced at the expense of reduced network stability. Let $\overline{\mathcal{R}_\alpha}$ denote the stability region for α-fair allocation. Then, the stability region $\overline{\mathcal{R}_\alpha}$ decreases as α increases, i.e. $\overline{\mathcal{R}_{\alpha_1}} \subset \overline{\mathcal{R}_{\alpha_2}}$ if $\alpha_1 > \alpha_2$. This is in sharp contrast to the case of fixed and convex rate regions, where fairness has no impact on stability.

As an example, consider a wired network with time-varying rate regions due to link failures. Different sets of broken links generate various link failure states, which in turn defines time-varying rate regions. We consider a network that allows the multi-path routing with flow splitting. The time-varying rate regions induced by link failures are illustrated in Figure 5, where the probability of each rate region is same, i.e. $\pi_1 = \pi_2 = \pi_3 = 1/3$. Figure 6 shows the change of stability regions for different values of α, where we observe the dependence of the size and even convexity of the stability region on the choice of α.

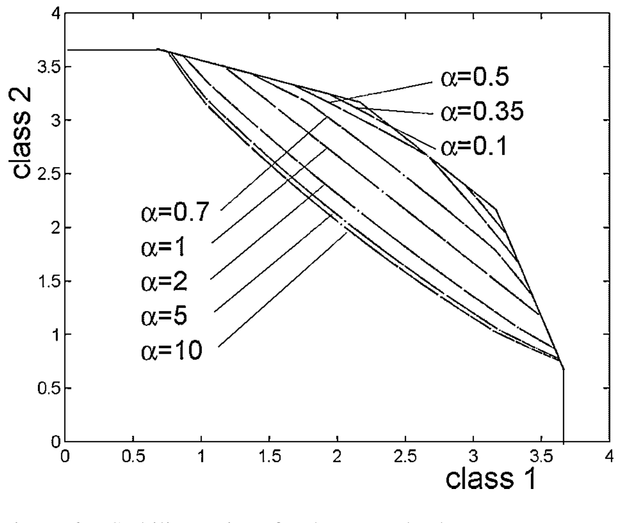

Figure 6. Stability regions for the example above.

6. RELATED WORK

There are several important topics of study that are highly related to the motivations and methodologies in stochastic network utility maximisation. For example, sometimes deterministic optimisation formulations can be derived from the limiting regime of stochastic network optimisation, as in the case of social welfare maximisation for loss networks in Reference [65, 66]. This section discusses four areas of related work.

6.1. Integrated network with elastic and real-time traffic

There are increasingly more real-time streaming flows on the Internet. Recall that the elastic flows are characterised by its file size. In contrast, streaming flows are typically characterised by its holding time (i.e. flow duration). The file size received within the required delay bound can be different.

To consider both elastic and streaming flows in one framework, an extension of the system model in earlier sections becomes necessary. First, we denote by $\lambda_s^{(e)}$ the intensity of Poisson arrival process of session-s elastic flows, and use $1/\mu_s$ as the mean of its exponential file size. The streaming flows of session s arrives again according to a Poisson process with intensity $\lambda_s^{(s)}$, and has its flow duration exponentially distributed with mean $1/h_s$. We denote by $\boldsymbol{M}(t)$ the vector of the numbers of streaming flow in the system, and continue to use $\boldsymbol{N}(t)$ for elastic flows.

Then, the system can be modelled by a Markov chain with states $(\boldsymbol{N}(t), \boldsymbol{M}(t))$, where the transition rates are

$$\begin{aligned}
(N_s(t), M_s(t)) &\to (N_s(t)+1, M_s(t)), \text{ with rate} \lambda_s^{(e)} \\
(N_s(t), M_s(t)) &\to (N_s(t)-1, M_s(t)), \text{ with rate} \\
&\qquad \mu_s N_s(t)\phi_s(\boldsymbol{N}(t)+\boldsymbol{M}(t)) \\
(N_s(t), M_s(t)) &\to (N_s(t), M_s(t)+1), \text{ with rate} \lambda_s^{(s)} \\
(N_s(t), M_s(t)) &\to (N_s(t), M_s(t)-1), \text{ with rate} \\
&\qquad h_s M_s(t)
\end{aligned}$$

Under this model, [67, 68] study sufficient conditions for session-level stability, and investigates the effect of each traffic type on the other. The elastic traffic places a load on each link, such that the remaining capacity is fairly shared by the streaming flows, whereas the streaming traffic is seen as a change of capacity constraints by the elastic flows.

6.2. *Balanced fairness and insensitive bandwidth sharing*

One of the main advantages of α-fair allocation by NUM is its amenability to distributed implementation. However, the existing result on session level issues for α-fair allocation is mainly on stability only. In other applications such as capacity dimensioning or buffer sizing, one may need to know performance metrics such as mean flow duration. The steady state distribution of the system under α-fair allocation is challenging to characterise and depends on the traffic characteristics such as arrival process and file size distribution. This motivates the research on insensitive bandwidth sharing, where the performance metrics depends only on the traffic intensity, denoted by $\boldsymbol{\rho}$ in Section 2.

The session-level model introduced in Section 2 can be thought of as a network of S processor-sharing, where S is the number of classes, and each queue corresponds to a flow-class. An α-fair allocation corresponds to one way of bandwidth sharing. The authors in References [21, 69] study a different bandwidth sharing, called 'balanced fairness', which is insensitive to traffic characteristics except for traffic intensity. Through this bandwidth sharing policy, we can easily compute the steady state distribution of the number of flows, leading to explicit computation of the expected flow duration time based on Little's formula.

Definition 6.1. *An allocation ϕ is said to be balanced if for all pairs of classes i, j and all states $\boldsymbol{x}$, with $x_i > 0$ and $x_j > 0$, we have*

$$\phi_i(\boldsymbol{x})\phi_j(\boldsymbol{x} - \boldsymbol{e}_i) = \phi_i(\boldsymbol{x} - \boldsymbol{e}_j)\phi_j(\boldsymbol{x})$$

where $\boldsymbol{e}_i = (0, \ldots, 1, \ldots, 0)$ a S-dimensional $\{0, 1\}$ vector with only ith element being 1.

Then, the system can be described as a Whittle network, and the distribution of the number of flows in each class is given by

$$\pi(\boldsymbol{x}) = \pi(0)\Phi(\boldsymbol{x})\rho_1^{x_1} \cdots \rho_S^{x_S}$$

where Φ is called the balance function [21, 69].

The steady state distribution is now just a function of traffic intensities, which helps us compute the mean number of flows in the system. Furthermore, it is proved that, for a resource allocation algorithm that does not satisfy this balanced property, the stationary distribution should be sensitive to all traffic characteristics. However, this bandwidth sharing policy is not known to allow distributed implementation.

Balanced fairness also does not lose stability regions. As in α-fair allocation, for any convex rate region, balanced fairness achieves the maximum stability region. The results in References [21, 69] are extended to the case of light and heavy loads to approximate the performance metrics in the whole load range in Reference [70].

6.3. *Buffer sizing and speed of variation in network randomness*

As discussed in Subsection 3.1, examining router timescale is important to understand the required buffer size at the routers. The buffer sizing question has recently gone through much debate. Different limiting models can be derived, depending on the different assumptions on the speed of variations in network randomness. Again, we use N to refer to the system scale.

Let B denotes the buffer size at the bottleneck router. Buffer size scaling is studied based on the following three regimes [39]:

(1) Small (or constant) regime: $B = \Theta(1)$, i.e. independent of the system scale size N,
(2) Intermediate regime: $B = \Theta(N^\beta)$, $0 < \alpha < 1$,
(3) Large regime: $B = \Theta(N)$.

Traditionally, backbone router design operates in the large buffer regime [71], mainly due to the window-based congestion control algorithm in TCP and its self-clocking feature. However, recent experimental studies on the buffer size scaling show that we can achieve enough statistical multiplexing gain and high network utilisation in the intermediate buffer regime [72] with $\beta = 1/2$, based on the idea that a large number of independent TCP flows leads to the normal distribution of the aggregate TCP window size. In Reference [73], an even small buffer is enough in the core routers, provided TCP is modified to reduce burstiness. The main assumption in Reference [72] is that all N flows go through a linear behaviour of TCP (i.e. congestion avoidance), and the required minimum buffer is calculated with an objective of full link utilisation. The small and intermediate buffer regime is based on the intuition that, with a large number of flows multiplexed at a large capacity router, randomness help de-synchronise flows, thus enabling full link utilisation without large buffer size.

The authors in Reference [74] raise some concerns about reducing the buffer size from the large buffer regime. They point out that the goal of the research in Reference [72] is primarily full link utilisation. However, extensive simulations show that loss rates range up to $5 - 15\%$, which harms certain real-time applications. They argue that the buffer sizing question should be first formulated with multiple objectives such as loss rate and delay bound, which may specialise into different regimes of the required buffer size.

Regarding the speed of randomness in network, the authors in References [37, 39] assume that the timescales of randomness in both TCP and unresponsive flows are in the same order. Thus, aggregate TCP and unresponsive flows form a Poisson process with parameter $(\lambda + x)$, where λ and x are the mean arrival rates of unresponsive and TCP flows, respectively. Therefore, the limiting system is approximated by an M/D/1 system with capacity c and Poisson input with parameter $\lambda + x$. On the other hand, Reference [38] assumes that the randomness of TCP flows happens at much slower timescale than that of unresponsive flows, i.e. over a small interval of time, the mean arrival rate of TCP flows looks constant. This assumption leads to an M/D/1 system, but with capacity $c - x$ and Poisson input with parameter λ. The assumptions and models in References [37, 39] can be viewed as a 'conservative' interpretation of the Internet. In contrast, the models in Reference [38] analyse the Internet on a 'optimistic' assumption, i.e. the randomness of TCP flows due to inter-packet jitter over a short time-interval can be ignored, and the randomness of unresponsive flows is dominant.

6.4. Distributed scheduling in wireless ad hoc networks

Distributed scheduling for medium access control is a crucial part as a resource allocation mechanism in wireless *ad hoc* networks. As an example, performance of a scheduling algorithm affects the shape of time-varying or fixed rate regions, which may lead to different stability regions for NUM. As another example, in the research of utility maximisation with system stability, scheduling forms part of the joint congestion control, routing and scheduling policy. It is also the bottleneck of distributed solution to this joint optimisation. In this subsection, we briefly survey research efforts on distributed implementation of scheduling algorithms.

Scheduling in constrained queueing system with the objective of throughput-guarantee dates back to the seminal work by Tassiulas and Ephremides [44], where an algorithm stabilising the system, whenever possible, is proposed in the name of 'max-weight' algorithm. In the max-weight algorithm, over each-timeslot, the scheduler chooses a schedule (i.e. a set of activated links) that maximises the sum of queue lengths in the scheduled links, referred to as weight. However, the 'max-weight' scheduling requires exponential computational complexity and centralised computation. In fact, max-weight scheduling can be reduced to a WMIS (Weighted Maximum Independent Set) problem that is NP-hard.

To overcome the high complexity, Reference [75] proposes a randomised algorithm, which we call pick-and-compare approach in this paper, that still achieves the maximum throughput-region, but only requires linear complexity. This is based on the idea that finding an optimal schedule (i.e. max-weight schedule) is not necessary at each timeslot. For stability guarantee, it is enough to find a reasonably good one with probabilistic guarantee of finding an optimal schedule. These two seminal algorithms are centralised, and has motivated many other subsequent work on distributed scheduling algorithms.

We categorise distributed scheduling into three types: (i) weight approximation, (ii) queue-length based random access and (iii) infrequent computation of max-weight schedule.

(i) Weight approximation. Maximal/greedy scheduling [52, 76, 77–79] belongs to this category. Here, at each timeslot the algorithms achieve the suboptimal weight, and their throughput-region is provably γ fraction of the maximum throughput-region. As an example, greedy scheduling achieves the half of the throughput-region under one-hop interference model [§]. The authors in Reference [80] have proved that maximal scheduling is a weight-approximating algorithm of 'max-interference-weight' scheduling that is defined slightly different from max-weight scheduling, with a different notion of weight.

(ii) Queue-length message based random access. One of the drawbacks of even weight-approximating algorithms above is that their complexity still grows with the given network size. To overcome the limitation and growing complexity with network size, queue-length based random-access algorithms [81, 82] have been proposed, requiring only constant complexity and achieves performance close to maximal scheduling.

(iii) Infrequent computation of optimal schedules. Throughput-region, measured by stability region of arrivals, is an asymptotic concept only. Intuitively, it may be true that the same throughput-region can be guaranteed by computing max-weight schedules infrequently, not on every timeslot. For the bounded number of arrivals over a timeslot, the queue-lengths are Lipschitz-continuous in timeslots. Then, it is guaranteed that the weight has only additive

[§] In the M-hop interference model, the links within M hops interfere with each other.

suboptimal terms with respect to the weight of a max-schedule, which does not affect the throughput. Quantifying such intuition, the pick-and-compare algorithm in Reference [75] reduces complexity without affecting throughput, where frequency of computation of optimal schedules is random. Although throughput is not affected by this frequency reduction, there may be some cost, such as delay, that has to be paid. The work in Reference [80] studies the three-way tradeoff between throughput, delay, and complexity, through a parameterised general framework that includes many of the scheduling algorithms discussed in this subsection.

In recent studies in References [83, 84], the authors develop families of scheduling algorithms that achieves arbitrary throughput-region, and discuss the complexity-efficiency tradeoff. In Reference [84], the authors propose a family of distributed scheduling algorithm parameterised by k and show that the algorithm k achieves $k/(k+2)$ of the maximum throughput-region, under one-hop interference model. The k-algorithm requires $4k+2$ rounds as control overhead, where one round is measured by the time for sending a message and receiving an ACK to and from neighbouring nodes. The work in Reference [84] has been extended to the case for the general M-hop interference model in Reference [85]. The authors in Reference [83] divide an entire graph into a disjoint subsets of links, such that each subset does not interfere with other subsets. With this 'graph partitioning' technique, the schedule computation inside each component can occur in a parallelised manner, leading to a reduction of control overhead complexity. The similar idea of graph-partitioning has been used in Reference [86]. The distributed throughput-optimal algorithms assuming local knowledge of arrival rates have also been proposed for various interference models in References [87, 88]. In Reference [80] mentioned earlier, the authors proposed a framework to study the wide range of scheduling algorithms in the research literature and characterized the achieved tradeoffs in stability, delay, and complexity. These characterizations reveal interesting properties hidden in the study of any one or two dimensions in isolation. For example, decreasing complexity from exponential to polynomial while keeping stability region the same, generally comes at the expense of exponential growth of delays. Investigating trade-offs in the 3-dimensional space allows a designer to fix one dimension and vary the other two jointly.

7. OPEN PROBLEMS

While there have been substantial progress in formulating and answering important questions in stochastic network utility maximisation, there are still many open problems and under-explored topics in this dynamic subject of study. We outline some of these challenges in this section.

7.1. Session-level dynamics

First, as we can see from the summary in Table 1, it still remains to prove session-level stability in the most general model in terms of file size distribution, topology and rate region. Further, it remains to study stability when the utility functions are non-concave and model inelastic traffic.

Second, stability is an asymptotic concept only. Deriving distributions on performance metrics, e.g. moments of mean duration times or queue sizes, are more important in practice and more challenging in theory.

Third, we mentioned that a recent result by Reference [20] shows that stability region in α-fair allocation is sensitive to the choice of α for non-convex and time-varying rate regions. More interestingly, the authors observe a tradeoff between fairness and stability region, but only prove the tradeoff for two classes of sessions. It remains to answer the question whether this tradeoff can be proved for arbitrary number of classes. Further, recall the max-projection (MP) allocation that achieves the maximum stability region regardless of the shape of rate regions. It is an open problem to find ways of implementing the MP allocation in a distributed manner.

Fourth, there is a mathematical need to develop a rigorous fluid limit for the difference equations in discrete time-slotted system model. This is important since, for example, in the research without timescale separation [12] that uses a discrete time model, the lack of fluid limit becomes an obstacle in proving session-level stability results for $\alpha < 1$.

7.2. Packet-level dynamics

Packet-level stochastic dynamics are generally too complicated to model in a tractable way, which motivate the research on validation of the applicability of deterministic fluid models, some of which we summarised in Section 3. However, most of the results are based on the study in a large-scale regime, e.g. the number of flows and router capacity go to infinity. But how many flows are 'many enough' for such approximations to be useful in a finite system? To answer this practically important question, we first have to further quantify the accuracy of deterministic approximation schemes, and to consider how aggregate traffic flows behave in different places of the network with different multiplexing scale, e.g. core, metro or access network. Answers to this question could in turn redefine the boundary between access networks and core networks in terms of the degree of multiplexing effect.

It would also be interesting to see if there exists a tractable yet accurate enough model between packet-level description and fluid approximation model. A related issue is raised in the research on utility maximisation subject to stability, where distributed optimisation and stochastic theory are brought together. For example, the dual subgradient update equation does not accurately describe the queuing dynamics under session- and packet-level stochastic dynamics, since there may not be packets to be sent out of a node even when it is allowed by the subgradient update equation. In general, shaping of packet departure time by upstream routers obviously impacts the arrival process in downstream nodes.

A different topic is to study the interactions between application layer protocols and NUM models for network resource allocation. For HTTP, it would be useful to extend [40] to the case with a mixture of long-lived and short-lived flows over general topology. Moreover, as shown by recent measurement studies such as [89], the dominant traffic in the current Internet is P2P traffic and video traffic. It is important to study how these application-layer changes interact with and affect the basic NUM model.

7.3. Constraint-level dynamics

One of the major sources that lead to disruptive dynamics at constraint-level is variation in network topology. For example, topology of wireless *ad hoc* networks can change due to mobility of nodes, sleep mode and battery power depletion. Solving generalised NUM problems over networks with randomly varying topologies remains an under-explored area, with little known results on models or methodologies. In some cases, topology-level dynamics can be assumed to be exogenous factors (similar to channel-level dynamics) that is independent of the NUM solutions. The problem becomes even more challenging when there is a coupling between NUM and topology-level stochastic. For example, topology may be varying because of battery depletion, which depend on the pattern of battery usage, which is in turn determined by the solution of the NUM problem itself.

8. CONCLUSION

To understand utility maximisation in a stochastic network requires the formulations and solutions of a wide range of new questions. These range from proving session-level stability of NUM to maximising utility subject to stability, from validating deterministic fluid models to exploiting network randomness, and from translating application layer burstiness to NUM models to using robustness against stochastic noise as a metric for comparison of alternative timescales. Interesting progress has been made on these issues since the introduction of NUM by Kelly as a fresh view on thinking about networking 10 years ago, and much more remains to be studied. Hopefully, in the future, tractability of the models and applicability of the subsequent results can be maintained simultaneously in a union between stochastic network theory and distributed optimisation theory.

ACKNOWLEDGEMENTS

This paper benefits a lot from our discussion with R. Srikant on the subject. We are also grateful for related discussions with N. Bambos, A. R. Calderbank, J. C. Doyle, J. Huang, F. P. Kelly, X. Lin, J. Liu, S. Low, R. Mazumdar, E. Modiano, M. Neely, H. V. Poor, A. Proutiere, D. Shah, S. Shakkottai, N. Shroff, A. Stolyar, A. Tang and J. Zhang. Research conducted at Princeton University and surveyed in this paper are in part supported by NSF CAREER grant CCF-0448012, NSF FIND grant CNS-0720570 and ONR Young Investigator Program grant N00014-07-1-0864.

REFERENCES

1. Kelly FP. Charging and rate control for elastic traffic. *European Transactions on Telecommunications* 1997; **8**:33–37.
2. Kelly FP, Maulloo A, Tan D. Rate control in communication networks: shadow prices, proportional fairness and stability. *Journal of the Operational Research Society* 1998; **49**:237–252.
3. Mo J, Walrand J. Fair end-to-end window-based congestion control. *IEEE/ACM Transactions on Networking* 2000; **8**(5):556–567.
4. Chiang M, Low SH, Calderbank AR, Doyle JC. Layering as optimization decomposition. *Proceedings of the IEEE* 2007; **95**(1):255–312.
5. Shannon CE. A mathematical theory of communication. *Bell Systems Technical Journal* 1948; **27**:379–423.
6. Lee JW, Mazumdar R, Shroff N. Non-convex optimization and rate control for multi-class services in the Internet. *IEEE/ACM Transactions on Networking* 2005; **13**(4):827–840.
7. Hande P, Zhang S, Chiang M. Distributed rate allocation for inelastic flows. *IEEE/ACM Transactions on Networking* 2007; **15**(6):1240–1253.
8. Kumar PR, Meyn SP. Stability of queueing networks and scheduling policies. *IEEE Transactions on Automatic Control* 1995; **14**(2):1055–1083.

9. Dai JG. On positive harris recurrence of multiclass queueing networks: a unified approach via fluid limit models. *Annals of Applied Probability* 1995; **5**:49–77.
10. de Veciana G, Lee T, Konstantopoulos T. Stability and performance analysis of networks supporting elastic services. *IEEE/ACM Transactions on Networking* 2001; **1**:2–14.
11. Bonald T, Massoulie L. Impact of fairness on internet performance. In *Proceedings of ACM Sigmetrics*, 2001.
12. Lin X, Shroff NB. On the stability region of congestion control. In *Proceedings of the 42nd Annual Allerton Conference on Communication, Control and Computing*, 2004.
13. Srikant R. On the positive recurrence of a markov chain describing file arrivals and departures in a congestion-controlled network. In *IEEE Computer Communications Workshop*, 2005.
14. Ye H, Ou J, Yuan X. Stability of data networks: stationary and bursty models. *Operations Research* 2005; **53**:107–125.
15. Bramson M. Stability of networks for max–min fair routing. *Presentation at INFORMS Applied Probability Conference*, 2005.
16. Lakshmikantha A, Beck CL, Srikant R. Connection level stability analysis of the internet using the sum of squares (SOS) techniques. In *Proceeding of the 38th Conference on Information Sciences and Systems*, 2004.
17. Massoulie L. Structural properties of proportional fairness: stability and insensitivity. *Annals of Applied Probability* 2007; **17**(3):809–839.
18. Gromoll HC, Williams R. Fluid limit of a network with fair bandwidth sharing and general document size distribution. *Annals of Applied Probability* 2007, in press.
19. Chiang M, Shah D, Tang A. Stochastic stability of network utility maximization: general file size distribution. In *Proceedings of Allerton Conference*, 2006.
20. Liu J, Proutiere A, Yi Y, Chiang M, Poor VH. Flow-level stability of data networks with non-convex and time-varying rate regions. In *Proceedings of ACM Sigmetrics*, 2007.
21. Bonald T, Massoulie L, Proutiere A, Virtamo J. A queueing analysis of max–min fairness, proportional fairness and balanced fairness. *Queueing Systems* 2006; **53**(1–2):65–84.
22. Bonald T, Borst S, Hegde N, Proutière A. Wireless data performance in multicell scenarios. In *Proceedings of ACM Sigmetrics*, 2004.
23. Luo W, Ephremides A. Stability of n interacting queues in random-access systems. In *IEEE Transactions on Information Theory* 1999; **45**(5):1579–1587.
24. Borst S, Leskela L, Jonckheere M. Stability of parallel queueing systems with coupled rates. *Discrete Event Dynamic Systems* 2008, DOI: 10.1007/S10626-007-0021-4.
25. Jonckheere M, Borst S. Stability of multi-class queueing systems with state-dependent service rates. In *Proceedings of IEEE Value Tools*, 2006.
26. Szpankowski W. Stability conditions for some multi-queue distributed systems: buffered random access systems. *Annals of Applied Probability* 1994; **26**:498–515.
27. Bonald T, Proutière A. Flow-level stability of utility-based allocations for non-convex rate regions. In *Proceedings of the 40th Conference on Information Sciences and Systems*, 2006.
28. Armony M, Bambos N. Queueing dynamics and maximal throughput scheduling in switched processing systems. *Queueing Systems* 2003; **44**(3):209–252.
29. Kelly FP, Williams RJ. Fluid model for a network operating under a fair bandwidth-sharing policy. *Annals of Applied Probability* 2004; **14**:1055–1083.
30. Shakkottai S, Srikant R. Mean FDE models for Internet congestion control under a many-flows regime. *IEEE Transactions on Information Theory* 2004; **50**(6):1050–1072.
31. Deb S, Shakkottai S, Srikant R. Asymptotic behavior of internet congestion controllers in a many-flows regime. *Mathematics of Operation Research* 2005; **30**(2):420–440.
32. Baccelli F, McDonald DR, Reynier J. A mean-field model for multiple TCP connections through a buffer implementing RED. *Performance Evaluation* 2002; **49**(1–4):77–97.
33. Tinnakornsrisuphap P, La RJ. Characterization of queue fluctuations in probabilistic AQM mechanisms. In *Proceedings of ACM Sigmetrics*, 2004.
34. Yi Y, Shakkottai S. On the elasticity of marking functions in an integraded network. *IEEE Transactions on Automatic Control* 2008; in press.
35. Johari R, Tan D. End-to-end congestion control for the Internet: delays and stability. *IEEE/ACM Transactions on Networking* 2001; **9**(6):818–832.
36. Cao J, Ramanan K. A poisson limit for buffer overflow probabilities. In *Proceedings of IEEE Infocom*, New York, NY, 2002.
37. Deb S, Srikant R. Rate-based versus Queue-based models of congestion control. In *Proceedings of ACM Sigmetrics*, 2004.
38. Yi Y, Deb S, Shakkottai S. Time-scale decomposition and rate-based marking. *IEEE/ACM Trasactions on Networking* 2006; **14**(5):938–950.
39. Raina G, Wischik D. Buffer sizes for large multiplexers: TCP queueing theory and instability analysis. In *Proceedings of Next Generation Internet Networks*, 2005.
40. Chang CS, Liu Z. A bandwidth sharing theory for a large number of http-like connections. *IEEE/ACM Transactions on Networking* 2004; **12**(5):952–962.
41. Zhang J, Zheng D, Chiang M. The impact of stochastic noisy feedback on distributed network utility maximization. *IEEE Transactions on Information Theory* 2008; **54**(2):645–665.
42. Chiang M. Balancing transport and physical layers in wireless multihop networks: jointly optimal congestion control and power control. *IEEE Journal on Selected Areas in Communications* 2005; **23**(1):104–116.
43. Mehyar M, Spanos D, Low SH. Optimization flow control with estimation error. In *Proceedings of IEEE Infocom*, 2004.
44. Tassiulas L, Ephremides A. Stability properties of constrained queueing systems and scheduling for maximum throughput in multihop radio networks. *IEEE Transactions on Automatic Control* 1992; **37**(12):1936–1949.
45. Agrawal R, Subramanian V. Optimality of certain channel aware scheduling policies. In *Proceedings of the 40th Annual Allerton Conference on Communication, Control and Computing*, 2002.
46. Kushner HJ, Whiting PA. Convergence of proportional-fair sharing algorithms under general conditions. *IEEE Transactions on Wireless Communications* 2004; **3**(4):1250–1259.
47. Liu X, Chong EKP, Shroff NB. Opportunistic transmission scheduling with resource-sharing constraints in wireless networks. *IEEE Journal on Selected Areas in Communications* 2001; **19**(10):2053–2064.
48. Viswanath P, Tse D, Laroia R. Opportunistic beamforming using dub antennas. *IEEE Transactions on Information Theory* 2002; **48**(6):49–77.
49. Stolyar AL. On the asymptotic optimality of the gradient scheduling algorithm for multi-user throughput allocation. *Operations Research* 2005; **53**(1):12–25.
50. Andrews DM. Instability of the proportional fair scheduling algorithm for HDR. *IEEE Transactions on Wireless Communications* 2004; **3**(5):1422–1426.
51. Eryilmaz A, Srikant R. Fair resource allocation in wireless networks using queue-length-based scheduling and congestion control. In *Proceedings of IEEE Infocom*, 2005.
52. Lin X, Shroff NB. The impact of imperfect scheduling on cross-layer rate control in wireless networks. In *Proceedings of IEEE Infocom*, 2005.
53. Neely MJ, Modiano E, Li C. Fairness and optimal stochastic control for heterogeneous networks. In *Proceedings of IEEE Infocom*, 2005.

54. Stolyar AL. Maximizing queueing network utility subject to statbility: greedy primal-dual algorithm. *Queueing Systems* 2005; **50**(4):401–457.
55. Chen L, Low SH, Chiang M, Doyle JC. Joint optimal congestion control, routing, and scheduling in wireless ad hoc networks. In *Proceeding of IEEE Infocom*, 2006.
56. Eryilmaz A, Srikant R. Joint congestion control, routing, and mac for stability and fairness in wireless networks. *IEEE Journal on Selected Areas in Communications* 2006; **24**(8):1514–1524.
57. Stolyar A. Greedy primal-dual algorithm for dynamic resource allocation in complex networks. *Queueing Systems* 2006; **54**:203–220.
58. Low SH. A duality model of TCP and queue management algorithms. *IEEE/ACM Transactions on Networking* 2003; **11**(4):525–536.
59. Low S, Srikant R. A mathematical framework for designing a low-loss low-delay internet. *Network and Spatial Economics, special issue on Crossovers between transportation planning and telecommunications* 2003; **4**:75–101.
60. Srikant R. *The mathematics of Internet Congestion Control*. Birkhauser, 2004.
61. Neely MJ. Super-fast delay tradeoffs for utility optimal fair scheduling in wireless networks. *IEEE Journal on Selected Areas in Communications* 2006; **24**(8):1489–1501.
62. Chaporkar P, Sarkar S. Stable scheduling policies for maximizing throughput in generalized constrained queueing. In *Proceedings of IEEE Infocom*, 2006.
63. Bender P, Black P, Grob M, Padovani R, Sindhushayana N, Viterbi A. CDMA/HDR: a bandwidth-efficient high-speed wireless data service for nomadic users. *IEEE Communications Magazine* 2000; **38**(4):70–77.
64. Borst S. User-level performance of channel-aware scheduling algorithms in wireless data networks. In *Proceedings of IEEE Infocom*, 2003.
65. Paschalidis IC, Liu Y. Pricing in multiservice loss networks: static pricing, asymptotic optimality, and demand substitution effects. *IEEE/ACM Transactions on Networking* 2002; **10**(3):425–438.
66. Paschalidis IC, Tsitsiklis JN. Congestion-dependent pricing of network services. *IEEE/ACM Transactions on Networking* 2000; **8**(2):171–184.
67. Key P, Massoulie L. Fluid models of integrated traffic and multipath routing. *Queueing Systems* 2006; **53**(1–2):85–98.
68. Key P, Massoulie L, Bain A, Kelly F. Fair internet traffic integration: network flow models and analysis. *Annales des Telecommunications* 2004; **59**:1338–1352.
69. Bonald T, Proutiere A. Insensitive bandwidth sharing in data networks. *Queueing Systems* 2003; **44**(1):69–100.
70. Bonald T, Penttinen A, Virtamo J. On light and heavy traffic approximations of balanced fairness. In *Proceedings of ACM Sigmetrics*, 2006.
71. Villamizar C, Song C. High performance TCP in ANSNET. *ACM/SIGCOMM Computer Communications Review* 1994; **24**(5):45–60.
72. Appenzeller G, Keslassy I, McKeown N. Sizing router buffers. In *Proceedings of ACM Sigcomm*, 2004.
73. Enachescu M, Ganjali Y, Goel A, McKeown N, Roughgarden T. Part iii: Routers with very small buffers. *ACM/SIGCOMM Computer Communication Review* 2005; **35**(3):83–90.
74. Dhamdhere A, Dovrolis C. Open issues in router buffer sizing. *ACM/SIGCOMM Computer Communications Review* 2006; **36**(1):87–92.
75. Tassiulas L. Linear complexity algorithms for maximum throughput in radionetworks and input queued switches. In *Proceedings of IEEE Infocom*, 1998.
76. Chaporkar P, Kar K, Sarkar S. Throughput guarantees through maximal scheduling in wireless networks. *Proceedings of the 43rd Annual Allerton Conference on Communication, Control and Computing*, 2005.
77. Sharma G, Mazumdar RR, Shroff NB. On the complexity of scheduling in wireless networks. In *Proceedings of ACM Mobicom*, 2006.
78. Sharma G, Shroff NB, Mazumdar RR. Joint congestion control and distributed scheduling for throughput guarantees in wireless networks. In *Proceedings of IEEE Infocom*, 2007.
79. Wu X, Srikant R. Bounds on the capacity region of multi-hop wireless networks under distributed greedy scheduling. In *Proceedings of IEEE Infocom*, 2006.
80. Yi Y, Proutiere A, Chiang M. Complexity in wireless scheduling: impact and tradeoffs. In *Proceedings of ACM Mobihoc*, 2008.
81. Joo C, Shroff NB. Performance of random access scheduling schemes in multi-hop wireless networks. In *Proceedings of IEEE Infocom*, 2007.
82. Lin X, Rasool S. Constant-time distributed scheduling policies for ad hoc wireless networks. In *Proceedings of IEEE CDC*, 2006.
83. Ray S, Sarkar S. Arbitrary throughput versus complexity tradeoffs in wireless networks using graph partitioning. *Proceedings of Information Theory and Applications Second Workshop*, 2007.
84. Sanghavi S, Bui L, Srikant R. Distributed link scheduling with constant overhead. In *Proceedings of ACM Sigmetrics*, 2007.
85. Yi Y, Chiang M. Wireless scheduling with o(1) complexity for m-hop interference model. In *Proceeding of ICC*, 2008.
86. Jung K, Shah D. Low delay scheduling in wireless network. In *Proceeding of ISIT*, 2007.
87. Yi Y, de Veciana G, Shakkottai S. Learning contention patterns and adapting to load/topology changesin in a MAC scheduling algorithm. In *Proceedings of IEEE WiMesh*, 2007.
88. Yi Y, de Veciana G, Shakkottai S. On optimal MAC scheduling under a physical interference model. In *Proceedings of IEEE Infocom*, 2006.
89. CAIDA. Http://www.caida.org