Foundations and Trends® in
Networking
Vol. 1, No 1 (2006) 1–144

now
the essence of knowledge

Resource Allocation and Cross-Layer Control in Wireless Networks

Leonidas Georgiadis[1], Michael J. Neely[2] and Leandros Tassiulas[3]

[1] *Aristotle University of Thessaloniki, Thessaloniki 54124, Greece, leonid@auth.gr*
[2] *University of Southern California, Los Angeles, CA 90089, USA, mjneely@usc.edu*
[3] *University of Thessaly, Volos, Greece, leandros@uth.gr*

Abstract

Information flow in a telecommunication network is accomplished through the interaction of mechanisms at various design layers with the end goal of supporting the information exchange needs of the applications. In wireless networks in particular, the different layers interact in a nontrivial manner in order to support information transfer. In this text we will present abstract models that capture the cross-layer interaction from the physical to transport layer in wireless network architectures including cellular, ad-hoc and sensor networks as well as hybrid wireless-wireline. The model allows for arbitrary network topologies as well as traffic forwarding modes, including datagrams and virtual circuits. Furthermore the time varying nature of a wireless network, due either to fading channels or to changing connectivity due to mobility, is adequately captured in our model to allow for state dependent network control policies. Quantitative performance measures that capture the quality of service requirements in these systems depending on the supported applications are discussed, including throughput maximization,

energy consumption minimization, rate utility function maximization as well as general performance functionals. Cross-layer control algorithms with optimal or suboptimal performance with respect to the above measures are presented and analyzed. A detailed exposition of the related analysis and design techniques is provided.

1

Introduction

In cross-layer designs of wireless networks, a number of physical and access layer parameters are jointly controlled and in synergy with higher layer functions like transport and routing. Furthermore, state information associated with a specific layer becomes available across layers as certain functions might benefit from that information. Typical physical and access layer functions include power control and channel allocation, where the latter corresponds to carrier and frequency selection in OFDM, spreading code and rate adjustment in spread spectrum, as well as time slot allocation in TDMA systems. Additional choices in certain wireless network designs may include the selection of the modulation constellation or the coding rate, both based on the channel quality and the desired rates [55, 156]. Due to the interference properties of wireless communication, the communication links between pairs of nodes in a multinode wireless environment cannot be viewed independently but rather as interacting entities where the bit rate of one is a function of choices for the physical and access layer parameters of the others. Our cross-layer model in this text captures the interaction of these mechanisms, where all the physical and access layer parameters are collectively represented through a control vector $I(t)$.

Another intricacy of a wireless mobile communication network is the fact that the channel and the network topology might be changing in time due to environmental factors and user mobility respectively. That variation might be happening at various time scales from milliseconds in the case of fast fading to several seconds for connectivity variations when two nodes get in and out of coverage of each other as they move. Actions at different layers need to be taken depending on the nature of the variability in order for the network to compensate in an optimal manner. All the relevant parameters of the environment that affect the communication are represented in our model by the topology state variable $S(t)$. The topology state might not be fully available to the access controller, which may observe only a sufficient statistic of that. The collection of bit rates of all communicating pairs of nodes at each time, i.e. the communication topology, is represented by a function $C(t) = C(I(t), S(t))$. Note that the function $C(.,.)$ incorporates among others the dependence of the link rate on the Signal-to-Interference plus Noise Ratio (SINR) through the capacity function of the link. Over the virtual communication topology defined by $C(t)$, the traffic flows from the origin to the destination according to the network and transport layer protocols. Packets may be generated at any network node having as final destination any other network node, potentially several hops away. Furthermore, the traffic forwarding might be either datagram or based on virtual circuits, while multicast traffic may be incorporated as well. The above model captures characteristics and slightly generalizes systems that have been proposed and studied in several papers including [108, 111, 115, 135, 136, 143, 144, 147, 149]. That model is developed in detail in Section 2 while representative examples of typical wireless models and architectures that fit within its scope are discussed there.

The network control mechanism determines the access control vector and the traffic forwarding decisions in order to accomplish certain objectives. The quantitative performance objectives should reflect the requirements posed by the applications. Various objectives have been considered and studied in various papers including the overall throughput, power optimization, utility optimization of the allocated rates as well as optimization of general objective functions of throughput and/or

power. In the current text we present control strategies for achieving these objectives.

The first performance attribute considered is the capacity region of the network defined as the set of all end-to-end traffic load matrices that can be supported under the appropriate selection of the network control policy. That region is characterized in two stages. First the ensemble of all feasible long-term average communication topologies is characterized. The capacity region includes all traffic load matrices such that there is a communication topology from the ensemble for which there is a flow that can carry the traffic load and be feasible for the particular communication topology. Section 3 is devoted to the characterization of the capacity region outlined above.

The capacity region of the network should be distinguished from the capacity region of a specific policy. The latter being the collection of all traffic load matrices that are sustainable by the specific policy. Clearly the capacity region of the network is the union of the individual policy capacity regions, taken over all possible control policies. One way to characterize the performance of a policy is by its capacity region itself. The larger the capacity region the better the performance will be since the network will be stable for a wider range of traffic loads and therefore more robust to traffic fluctuations. Such a performance criterion makes even more sense in the context of wireless ad-hoc networks where both the traffic load as well as the network capacity may vary unpredictably. A policy A is termed "better" than B with respect to their capacity regions, if the capacity region of A is a superset of the capacity region of B. A control policy that is optimal in the sense of having a capacity region that coincides with the network capacity region and is therefore a superset of the capacity region of any other policy was introduced in [143, 147]. That policy, the max weight adaptive back-pressure policy, was generalized later in several ways [111, 115, 135, 149] and it is an essential component of policies that optimize other performance objectives. It is presented in Section 4. The selection of the various control parameters, from the physical to transport layer, is done in two stages in the max weight adaptive back pressure policy. In the first stage all the parameters that affect the transmission rates of the wireless links are selected, i.e. the function $C(I(t), S(t))$ is determined. In the

second stage routing and flow control decisions to control multihop traffic forwarding are made. The back pressure policy consists in giving priority in forwarding through a link to traffic classes that have higher backlog differentials. Furthermore the transmission rate of a link that leads to highly congested regions of the network is throttled down. In that manner the congestion notification travels backwards all the way to the source and flow control is performed. Proofs of the results based on Lyapunov stability analysis are presented also in Section 4.

The stochastic optimal control problem where the objective is the optimization of a performance functional of the system is considered in Sections 5 and 6. The development of optimal policies for these cases relies on a number of advances including extensions of Lyapunov techniques to enable simultaneous treatment of stability and performance optimization, introduction of virtual cost queues to transform performance constraints into queueing stability problems and introduction of performance state queues to facilitate optimization of time averages. These techniques have been developed in [46, 108, 115, 116, 136, 137] for various performance objectives. More specifically in Section 5 the problem of optimizing a sum of utility functions of the rates allocated to the different traffic flows is considered. That formulation includes the case of the traffic load in the system being out of the capacity region, which case some kind of flow control at the edges of the network needs to be employed. That is done implicitly through the use of performance state queues, allowing adjustment of the optimization accuracy through a parameter. The approach combines techniques similar to those used for optimization of rate utility functions in window flow controlled sessions in wireline networks, with max weight scheduling for dealing with the wireless scheduling. In Section 6 generalization of these techniques for optimization functionals that combine utilities with other objectives like energy expenditure are given and approaches relying on virtual cost queues are developed.

Most of the results presented in the text are robust on the statistics of the temporal model both of the arrivals as well as the topology variation process. The traffic generation processes might be Markov modulated or belong to a sample path ensemble that complies with certain burstiness constraints [35, 148]. Similarly the variability of the

topology might be modeled by a hidden Markov process. These models are adequate to cover most of the interesting cases that might arise in real networks. The proofs in the text are provided for a traffic generation model that covers all the above cases and it was considered in [115]. The definition of stability that was used implies bounded average backlogs. The emphasis in the presentation is on describing the models and the algorithms with application examples that illustrate the range of possible applications. Representative cases are analyzed in full detail to illustrate the applicability of the analysis techniques, while in other cases the results are described without proofs and references to the literature are provided.

2

The Network Model and Operational Assumptions

Consider a general network with a set $\mathcal{N}$ of nodes and a set $\mathcal{L}$ of transmission links. We denote by N and L respectively the number of nodes and links in the network. Each link represents a communication channel for direct transmission from a given node a to another node b, and is labeled by its corresponding ordered node pair (a,b) (where $a,b \in \mathcal{N}$). Note that link (a,b) is distinct from link (b,a). In a wireless network, direct transmission between two nodes may or may not be possible and this capability, as well as the transmission rate, may change over time due to weather conditions, mobility or node interference. Hence in the most general case one can consider that $\mathcal{L}$ consists of all ordered pairs of nodes, where the transmission rate of link (a,b) is zero if direct communication is impossible. However, in cases where direct communication between some nodes is never possible, it is helpful to consider that $\mathcal{L}$ is a strict subset of the set of all ordered pairs of nodes.

The network is assumed to operate in slotted time with slots normalized to integral units, so that slot boundaries occur at times $t \in \{0,1,2,\ldots\}$. Hence, slot t refers to the time interval $[t,t+1)$. Let $\boldsymbol{\mu}(t) = (\mu_{ab}(t))$ represent the matrix of transmission rates offered over

each link (a,b) during slot t (in units of bits/slot).[1] By convention, we define $\mu_{ab}(t) = 0$ for all time t whenever a physical link (a,b) does not exist in the network. The link transmission rates are determined by a *link transmission rate function* $\boldsymbol{C}(I,S)$, so that:

$$\boldsymbol{\mu}(t) = \boldsymbol{C}(I(t), S(t)),$$

where $S(t)$ represents the network *topology state* during slot t, and $I(t)$ represents a *link control action* taken by the network during slot t.

The topology state process $S(t)$ represents all uncontrollable properties of the network that influence the set of feasible transmission rates. For example, the network channel conditions and interference properties might change from time to time due to user mobility, wireless fading, changing weather locations, or other external environmental factors. In such cases, the topology state $S(t)$ might represent the current set of node locations and the current attenuation coefficients between each node pair. While this topology state $S(t)$ can contain a large amount of information, for simplicity of the mathematical model we assume that $S(t)$ takes values in a finite (but arbitrarily large) state space $\mathcal{S}$. We assume that the network topology state $S(t)$ is constant for the duration of a timeslot, but potentially changes on slot boundaries.

The link control input $I(t)$ takes values in a general state space $\mathcal{I}_{S(t)}$, which represents all of the possible resource allocation options available under a given topology state $S(t)$. For example, in a wireless network where certain groups of links cannot be activated simultaneously, the control input $I(t)$ might specify the particular set of links chosen for activation during slot t, and the set $\mathcal{I}_{S(t)}$ could represent the collection of all feasible link activation sets under topology state $S(t)$. In a power constrained network, the control input $I(t)$ might represent the matrix of power values allocated for transmission over each data link. Likewise, the transmission control input $I(t)$ might include bandwidth allocation decisions for every data link.

[1] Transmission rates can take units other than bits/slot whenever appropriate. For example, in cases when all data arrives as fixed length packets and transmission rates are constrained to integral multiples of the packet size, then it is often simpler to let $\boldsymbol{\mu}(t)$ takes units of packets/slot.

Every timeslot the network controller observes the current topology state $S(t)$ and chooses a transmission control input $I(t) \in \mathcal{I}_{S(t)}$, according to some *transmission control policy*. This enables a transmission rate matrix of $\boldsymbol{\mu}(t) = \boldsymbol{C}(I(t), S(t))$. The function $\boldsymbol{C}(I, S)$ is matrix valued and is composed of individual $C_{ab}(I, S)$ functions that specify the individual transmission rates on each link (a, b), so that $\mu_{ab}(t) = C_{ab}(I(t), S(t))$. In general, the rate function for a single link can depend on the full control input $I(t)$ and the full topology state $S(t)$ and hence distributed implementation may be difficult. This is often facilitated when rate functions for individual links depend only on the local control actions and the local topology state information associated with those links. These issues will be discussed in more detail in later sections.

2.1 Link rate function examples for different networks

In this section we consider different types of networks and their corresponding link rate functions $\boldsymbol{C}(I(t), S(t))$. Our examples include static wireline networks, rate adaptive wireless networks, and ad-hoc mobile networks.

Example 2.1. *A static wireline network with fixed link capacities.* Consider the six node network of Fig. 2.1a. The network is connected via wired data links, where each link (a, b) offers a fixed transmission rate C_{ab} for all time. In this case, there is no notion of a time varying topology state $S(t)$ or a control input $I(t)$, and so the transmission rate function for each link (a, b) is given by $C_{ab}(I(t), S(t)) = C_{ab}$ (where $C_{ab} = 0$ if there is no link from node a to node b). The network is thus fully described by a constant matrix (C_{ab}) of link capacities, which is the conventional way to describe a wireline network.

Example 2.2. *A network with time varying link capacities.* Consider the same network as in Example 2.1, but assume now that every timeslot the data links can randomly become active or inactive. In particular, an active link (a, b) can transmit at rate C_{ab} as before, but an inactive link cannot transmit. Let $S_{ab}(t)$ be a link state process taking values in the two-element set $\{\text{ON}, \text{OFF}\}$, where $S_{ab}(t) = \text{ON}$ if link (a, b) is

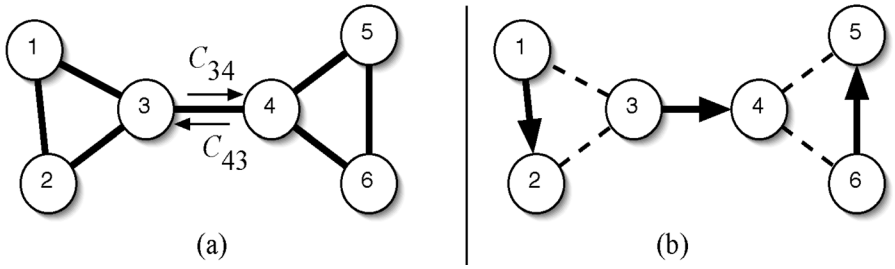

Fig. 2.1 (a) A static network with 6 nodes and constant link capacities C_{ab}. (b) A network with configurable link activation sets.

active during slot t, and $S_{ab}(t) = \text{OFF}$ otherwise. The topology state $S(t)$ of the network is thus the matrix $(S_{ab}(t))$ composed of individual link state processes, and the link transmission rate functions are given by $C_{ab}(I(t), S(t)) = C_{ab}(S_{ab}(t))$, where:

$$C_{ab}(S_{ab}(t)) = \begin{cases} C_{ab} & \text{if } S_{ab}(t) = \text{ON} \\ 0 & \text{if } S_{ab}(t) = \text{OFF} \end{cases} .$$

In this example, the link transmission rate function depends on a time varying topology state variable $S(t)$, but there is still no notion of resource allocation. Further note that the stochastics of the link activation processes $S_{ab}(t)$ are not specified here. A simple model might be that each process $S_{ab}(t)$ is independently inactive with some outage probability p_{ab} every slot, and active otherwise. However, in general, the $S_{ab}(t)$ processes could be correlated with each other and also correlated in time.

Example 2.3. *A static wireless network with configurable link activation sets.* Consider a wireless network with stationary nodes and time invariant channel conditions between each node pair. Suppose that due to interference and/or hardware constraints, transmission over a link can take place only if certain constraints are imposed on transmissions over the other links in the network. For example, a node may not transmit and receive at the same time over some of its attached links, or a node may not transmit when a neighboring node is receiving, etc. A given link (a, b) can support a transmission rate C_{ab}, provided that it

is scheduled for activation and no other interfering links are activated. For each link (a,b), we define a control process $I_{ab}(t)$, where $I_{ab}(t) = 1$ if link (a,b) is activated during slot t, and 0 else. The control input process $I(t)$ thus consists of the matrix $(I_{ab}(t))$, and this matrix is restricted every timeslot to the set $\mathcal{I}$ consisting of all *feasible link activation sets*. That is, the set $\mathcal{I}$ contains all sets of links that can be simultaneously activated without creating inter-link interference. The link transmission rate function is thus given by $C_{ab}(I(t), S(t)) = C_{ab}(I_{ab}(t))$, where:

$$C_{ab}(I_{ab}(t)) = \begin{cases} C_{ab} & \text{if } I_{ab}(t) = 1 \\ 0 & \text{if } I_{ab}(t) = 0 \end{cases} .$$

where the input satisfies the constraint $(I_{ab}(t)) \in \mathcal{I}$ for all t. An example network with three activated links is shown in Fig. 2.1b. While this link transmission rate function is similar in structure to that of Example 2.2, we note that the link capacities of Example 2.2 depend on random and uncontrollable channel processes, while the link capacities in this example are determined by the network control decisions made every timeslot. This is an important distinction, and the notion of link activation sets can be used to model general problems involving *network server scheduling*. Such problems are treated in [143] for multi-hop radio networks with general activation sets $\mathcal{I}$. An interesting special case is when $\mathcal{I}$ is defined as the collection of all link sets such that no node is the transmitter or receiver of more than one link in the set. Such sets of links are called *matchings*. This special case has been used extensively in the literature on crossbar constrained packet switches, where the network nodes are arranged according to a bipartite graph (see for example, [87, 103, 109, 113, 143, 150, 162]). Matchings are also used in [29, 61, 91, 150, 163] to treat scheduling in computer systems and ad-hoc networks with arbitrary graph structures. Note that there is an inherent difficulty in implementing control decisions in a distributed manner under this model. Indeed, the constraint $I(t) \in \mathcal{I}$ couples the link activation decisions at every node, and often extensive message passing is required before a matching is computed and its feasibility is verified. Generally, the complexity associated with finding a valid matching increases with the size of the network. Complexity can also be reduced by considering sub-optimal matchings, which often

yields throughput within a certain factor of optimality. This approach is considered in [29, 91, 163] (see also Section 4.7).

Example 2.4. *A time varying wireless downlink.* Consider a single wireless node that transmits to M downlink users (such as a satellite unit or a wireless base station, see Fig. 2.2a). Let $S_i(t)$ represent the condition of downlink i during slot t (for each link $i \in \{1, \ldots, M\}$). Suppose that channel conditions are grouped into four categories, so that: $S_i(t) \in \{GOOD, MEDIUM, BAD, ZERO\}$. Suppose that at most one link can be activated during any slot, and that an active link can transmit 3 packets when in the GOOD state, 2 packets in the MEDIUM state, 1 in the BAD state, and none in the ZERO state. The topology state $S(t)$ for this system is given by the vector $(S_1(t), \ldots, S_M(t))$. The control input $I(t)$ is given by the vector $(I_1(t), \ldots, I_M(t))$, where $I_i(t)$ takes the value 1 if link i is activated in slot t, and zero else. The control space $\mathcal{I}$ is the set of all vectors $(I_1, \ldots, I_M)$ with at most one entry equal to 1 and all other entries equal to zero. As there is only a single transmitting node, we can express the link transmission rate function as a vector: $\boldsymbol{C}(I(t), S(t)) = (C_1(I(t), S(t)), \ldots, C_M(I(t), S(t)))$. Each function entry has the form $C_i(I(t), S(t)) = C_i(I_i(t), S_i(t))$, where:

$$C_i(I_i(t), S_i(t)) = \begin{cases} 3I_i(t) & \text{if } S_i(t) = \text{GOOD} \\ 2I_i(t) & \text{if } S_i(t) = \text{MEDIUM} \\ 1I_i(t) & \text{if } S_i(t) = \text{BAD} \\ 0 & \text{else} \end{cases}.$$

This type of downlink model is used to treat satellite and wireless systems in [4, 95, 110, 144]. The model can be extended to include *power allocation* in cases when transmission rates depend upon a continuous power parameter [110]. For example, the transmission rate on each downlink $i \in \{1, \ldots, M\}$ might be approximated by the expression for Shannon capacity over an additive white Gaussian noise channel:

$$C_i(P_i(t), S_i(t)) = \log(1 + P_i(t)\alpha_{S_i(t)}), \tag{2.1}$$

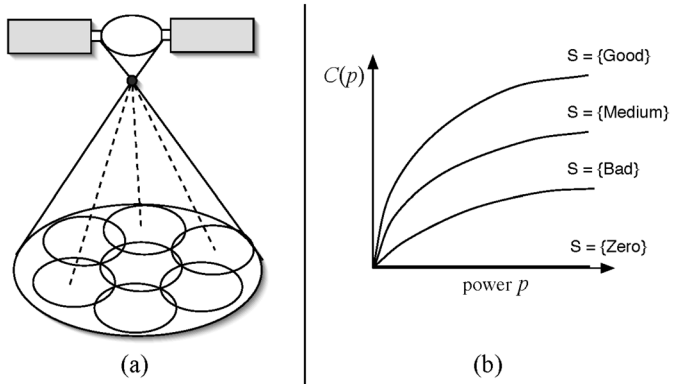

Fig. 2.2 (a) An example satellite downlink with M downlink channels ($M = 7$ in the example). (b) An example set of rate-power curves for the power allocation problem with four discrete channel states.

where $P_i(t)$ is the power allocated to channel i during timeslot t, and $\alpha_{S_i(t)}$ is the attenuation-to-noise coefficient associated with channel state $S_i(t)$ (see Fig. 2.2b). In this case, the control input $I(t)$ is given by the power vector $\boldsymbol{P}(t) = (P_1(t), \ldots, P_M(t))$, and the control space can be a continuum of feasible power vectors, such as all vectors that satisfy a peak power constraint $\sum_{i=1}^{M} P_i \leq P_{peak}$.

Example 2.5. *A time varying ad-hoc network with interference.* Consider an ad-hoc wireless network with a set of nodes $\mathcal{N}$ and set of links $\mathcal{L}$. We assume that each link $l = (a, b)$, has a transmitter located at node a and a receiver located at node b. Let $P_l(t)$ represent the power that the transmitter of link l allocates for transmission over that link, and let $\boldsymbol{P}(t) = (P_l(t))_{l \in \mathcal{L}}$ represent the power allocation vector. In this case, the control input $I(t)$ is equal to the power vector $\boldsymbol{P}(t)$, and the constraint set $\mathcal{I}$ is given by the set $\mathcal{P}$ consisting of all power vectors that satisfy peak power constraints at every node. The transmission rate function for link l is given by $C_l(I(t), S(t)) = C_l(\boldsymbol{P}(t), S(t))$. Assume that this function depends on the overall Signal to Interference plus Noise Ratio (SINR) according to a logarithmic capacity curve:

$$C_l(\boldsymbol{P}(t), S(t)) = \log\left(1 + SINR_l(\boldsymbol{P}(t), S(t))\right).$$

Here $SINR_l(\boldsymbol{P}(t), S(t))$ is given by:

$$SINR_l(\boldsymbol{P}(t), S(t)) = \frac{P_l(t)\alpha_{ll}(S(t))}{N_0 + \sum_{\substack{k \in \mathcal{L} \\ k \neq l}} P_k(t)\alpha_{kl}(S(t))},$$

where N_0 is the background noise intensity on each link and $\alpha_{kl}(S(t))$ is the attenuation factor at the receiver of link l of the signal power transmitted by the transmitter of link k when the topology state is $S(t)$. Hence, in this model the interference caused at the receiver of link l by the signals transmitted by the transmitters of the other links in modeled as additional noise. This *SINR* network model is quite common in the wireless and ad-hoc networking literature. For example, [111] considers this model for mobile ad-hoc networks, and [31, 36, 42, 66, 92, 123, 124, 127, 128, 129, 167, 171] for static ad-hoc networks and cellular systems. This model in the case of a system with antenna arrays and beamforming capabilities is considered in [28, 47, 48, 130]. It is quite challenging to implement optimal controllers for this type of link transmission rate function. Indeed, as in Example 2.3, the control input decisions are coupled at every node, because the power allocated for a particular data link can act as interference at all other links, and this interference model can change depending on the network topology state. While distributed algorithms exist for computing the rate associated with a particular power allocation, and for determining if a power allocation exists that leads to a given set of link rates [167, 171], there are no known low complexity algorithms for finding the power vectors that optimize the performance metrics required for optimal network control. However, randomized distributed approximations exist for such systems and offer provable performance guarantees [57, 111, 115]. Furthermore, important special cases of the *low $SINR$ regime* are treated in [36, 127, 129] using the approximation $\log(1 + SINR) \approx SINR$, and the *high $SINR$ regime* is treated in [31, 66] using the approximation $\log(1 + SINR) \approx \log(SINR)$.

Example 2.6. *An ad-hoc mobile network.* Consider a network with a set $\mathcal{N}$ of mobile users. The location of each user is quantized according

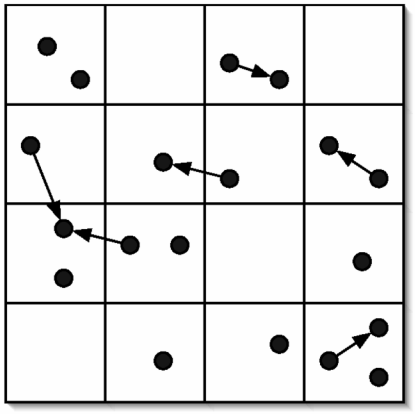

Fig. 2.3 An ad-hoc mobile network with a cell partitioned structure.

to a rectilinear cell partitioning that covers the network region of interest, as shown in Fig. 2.3b. We assume that the channel conditions (noise, attenuation factor) are time-invariant throughout the region so that link transmission capabilities are determined solely by node locations. Let $S_a(t)$ represent the cell location of node a during slot t. The topology state variable $S(t)$ consists of the vector $(S_a(t))_{a\in\mathcal{N}}$ (one component for each node $a \in \mathcal{N}$), and can change from slot to slot as nodes move from cell to cell (according to some mobility process that is potentially different for every node). In this case, the link transmission rate function can be given by the $SINR$ model of Example 2.5, where the attenuation coefficients $\alpha_{kl}(S(t))$ are determined by the current node locations. Note that the mobility model has been left unspecified. Any desired mobility model can be used, such as Markovian random walks [111], periodic walks, random waypoint mobility [25], independent cell hopping [90, 114], etc. The network model can be simplified by assuming *no inter-cell interference*. Specifically, suppose that nodes can only transmit to other nodes in the same cell or in adjacent cells, and that at most one node can transmit per cell during a single timeslot. Suppose that transmissions in adjacent cells use orthogonal frequency bands, and that interference from non-adjacent cells is negligible. In this case, transmission decisions can be distributed cell-by-cell. Let $I_{ab}(t)$ be a control process that takes the value 1 if link (a, b) is activated

during slot t, and zero else (as in Example 2.3). Let $I(t) = (I_{ab}(t))$ represent the matrix of transmission decisions, restricted to the control space $\mathcal{I}_{S(t)}$ that specifies all feasible link activations under a given topology state $S(t)$. Suppose that the transmission rate of an in-cell transmission is h packets/slot, and that of an adjacent cell transmission is l packets/slot (where $h \geq l$). The link transmission rate function is thus given by $C_{ab}(I(t), S(t)) = C_{ab}(I_{ab}(t), S_a(t), S_b(t))$ (where $C_{ab}(\cdot)$ takes units of packets/slot), and we have:

$$C_{ab}(I_{ab}(t), S_a(t), S_b(t)) = \begin{cases} h & \text{if } I_{ab}(t) = 1 \text{ and } S_a(t) = S_b(t) \\ l & \text{if } I_{ab}(t) = 1 \text{ and } S_a(t) \neq S_b(t) \\ 0 & \text{else} \end{cases} .$$

where $(I_{ab}(t)) \in \mathcal{I}_{S(t)}$. Similar *cell partitioned network models* are used in [114, 115, 116, 160]. Note that this model allows the possibility of a single node transmitting over one frequency band while simultaneously receiving over another frequency band. In systems where this is infeasible, the additional constraint that a node cannot simultaneously transmit and receive must be imposed. This couples transmission decisions over the entire network and complicates optimal distributed control. One (potentially sub-optimal) scheduling alternative is to randomly choose a set of *transmitter nodes* and a set of *receiver nodes* every timeslot (as in [57, 111]). Only nodes in the receiving set are valid options for the transmitters. Another approach is to allow nodes to send *transmission requests*, and allow an arbiter to determine which requests are granted. Several rounds of arbitration can take place to improve scheduling decisions. Simple types of one-step arbitration schemes are designed into wireless protocols such as 802.11, where *request to send* and *clear to send* messages regulate which network links are simultaneously active [125]. Multi-step arbitration schemes are frequently used in packet switches for computer systems [3, 41, 104, 139]. The control techniques that we develop in this text reveal principled strategies for making these scheduling decisions in terms of current network conditions and desired performance objectives.

These examples illustrate the wide class of data networks that fall within the scope of our model. In summary, the function $\boldsymbol{C}(I(t), S(t))$

describes the physical and multiple access layer properties of a given network.[2] Viewing the network in terms of this abstract function provides insight into the fundamental control techniques applicable to *all data networks* while enabling these techniques to take maximum advantage of the unique properties of each data link.

2.2 Routing and network layer queueing

All data that enters the network is associated with a particular *commodity*, which minimally defines the destination of the data, but might also specify other information, such as the source node of the data or its priority service class. Let $\mathcal{K}$ represent the set of commodities in the network, and let K represent the number of distinct commodities in this set. Let $A_i^{(c)}(t)$ represent the amount of new commodity c data that exogenously arrives to source node i during slot t (for all $i \in \mathcal{N}$ and all $c \in \mathcal{K}$). We assume that $A_i^{(c)}(t)$ takes units of bits, although it can take other units when appropriate (such as units of *packets*). The $A_i^{(c)}(t)$ data is generated from the user or application associated with source node i, and is not necessarily admitted directly to the network layer. Rather, we view the $A_i^{(c)}(t)$ data as arriving to the *transport layer* at node i, and for each timeslot t we define $R_i^{(c)}(t)$ as the amount of commodity c data allowed to enter the network layer from the transport layer at node i.

Each node i maintains a set of internal queues for storing network layer data according to its commodity (Fig. 2.4). Let $U_i^{(c)}(t)$ represent the current backlog, or *unfinished work*, of commodity c data stored in a network layer queue at node i. The queue backlog $U_i^{(c)}(t)$ can contain both data that arrived exogenously from the transport layer at node i as well as data that arrived endogenously through network layer transmissions from other nodes. In the special case when node i is the destination of commodity c data, we formally define $U_i^{(c)}(t)$ to be 0 for all t, so that any data that is successfully delivered to its destination is assumed to exit the network layer. We assume that all network

[2] See [18] for a definition and discussion of the various layers associated with the standard 7 layer Open Systems Interconnection (OSI) networking model, including the transport, network, and physical layers, and the multiple access sub-layer.

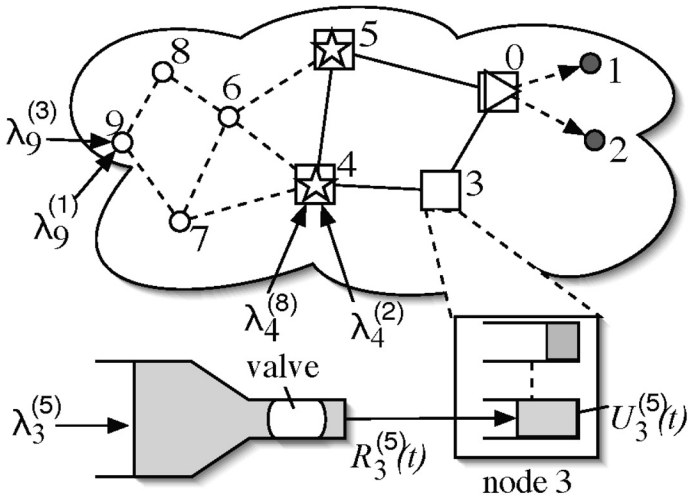

Fig. 2.4 A heterogeneous network with transport layer storage reservoirs and internal network layer queues at each node.

layer queues have *infinite buffer storage space*. Our primary goal for this layer is to ensure that all queues are *stable*, so that time average backlog is finite. This performance criterion tends to yield algorithms that also perform well when network queues have finite buffers that are sufficiently large.

A *network layer control algorithm* makes decisions about routing, scheduling, and resource allocation in reaction to current topology state and queue backlog information. The resource allocation decision $I(t) \in \mathcal{I}_{S(t)}$ determines the transmission rates $\mu_{ab}(t) = C_{ab}(I(t), S(t))$ offered over each link (a, b) on timeslot t. In general, multiple commodities might be transmitted over this link during a single timeslot.[3] Define $\mu_{ab}^{(c)}(t)$ as the rate offered to commodity c data over the (a, b) data link during slot t. These $\mu_{ab}^{(c)}(t)$ values represent *routing decision variables* chosen by the network controller. It is often convenient to impose routing restrictions for each commodity, and hence we define $\mathcal{L}_c$ as the set of all links (a, b) that commodity c data is allowed to use.

[3] We shall find that we can restrict control laws to transmitting only a single commodity per link, without loss of optimality.

Thus, the controller at each node $a \in \mathcal{N}$ chooses the routing decision variables $\mu_{ab}^{(c)}(t)$ subject to the following *routing constraints*:

$$\sum_{c \in \mathcal{K}} \mu_{ab}^{(c)}(t) \leq \mu_{ab}(t), \tag{2.2}$$

$$\mu_{ab}^{(c)}(t) = 0, \quad \text{if } (a,b) \notin \mathcal{L}_c. \tag{2.3}$$

We assume that only the data currently in node i at the beginning of slot t can be transmitted during that slot. Hence, the slot-to-slot dynamics of the queue backlog $U_i^{(c)}(t)$ satisfies the following inequality:

$$U_i^{(c)}(t+1) \leq \max\left[U_i^{(c)}(t) - \sum_b \mu_{ib}^{(c)}(t), 0\right] + R_i^{(c)}(t) + \sum_a \mu_{ai}^{(c)}(t). \tag{2.4}$$

The above expression is an inequality rather than an equality because the actual amount of commodity c data arriving to node i during slot t may be less than $\sum_a \mu_{ai}^{(c)}(t)$ if the neighboring nodes have little or no commodity c data to transmit.

2.2.1 On the link constraint sets $\mathcal{L}_c$

The routing constraint (2.3) restricts commodity c data from using links outside of the set $\mathcal{L}_c$. The constraint sets $\mathcal{L}_c$ are arbitrary, and hence the above model includes the special case of *single-hop* networks where only direct transmissions between nodes is allowed. This can be accomplished by setting $\mathcal{L}_c = \{(a,b)\}$ for each commodity c whose traffic is originated at node a and destined to node b**.** Also, the above model includes the special case of *unconstrained routing*, where each set $\mathcal{L}_c$ contains all of the links of the network. In this case, the network does not require a pre-specified route. Routing decisions can be made dynamically at each node, and packets of the same commodity can potentially traverse different paths. While unconstrained routing allows for the largest set of options, it can often be complex and may lead to large network delay in cases when some packets are transmitted in directions that take them further away from their destinations.

To ensure more predictable performance and to (potentially) reduce these delay problems, the link sets $\mathcal{L}_c$ can be designed in advance

to ensure that all transmissions move commodities closer to their destinations. Note that restricting the routing options makes the network less capable of adapting to random link failures, outages, or user mobility, whereas unconstrained routing can in principle adapt by dynamically choosing a new direction.

Both unconstrained and constrained routing allow for a multiplicity of paths. In cases when it is desirable to restrict sessions to a single path (perhaps to ensure in-order packet delivery), each set $\mathcal{L}_c$ can be specified as a *directed tree* with final node given by the destination node for commodity c. Alternatively, in cases when it is desired for different paths to cross but not merge, a different commodity c can be associated with each different source-destination pair, and the link set $\mathcal{L}_c$ is defined as the set of all links in the path for commodity c.

2.3 Flow control and the transport layer

All exogenous arrivals $A_i^{(c)}(t)$ first enter the transport layer at their corresponding source nodes, and this data is held in *storage reservoirs* to await acceptance to the network layer (Fig. 2.4). We assume there is a separate storage reservoir for each commodity at each node, and define $L_i^{(c)}(t)$ as the backlog of commodity c bits currently stored in the transport layer storage reservoir at node i. Every timeslot, each source node i makes *flow control decisions* by choosing the amount of bits $R_i^{(c)}(t)$ to deliver to the network layer at node i, subject to the constraint $R_i^{(c)}(t) \leq L_i^{(c)}(t) + A_i^{(c)}(t)$ for all (i,c) and all t, and subject to some additional constraints made precise in Section 5.

The storage reservoirs for each commodity may be infinite or finite, with size $0 \leq L_i^{max} \leq \infty$. Therefore, some data must be dropped if the new exogenous arrivals are not admitted to the network layer and do not fit into the storage reservoir. Hence, $L_i^{(c)}(t) \leq L_i^{max}$ for all t, and the dynamics of storage buffer (i,c) from one timeslot to the next satisfies the following inequality:

$$L_i^{(c)}(t+1) \leq \min\left[L_i^{(c)}(t) - R_i^{(c)}(t) + A_i^{(c)}(t),\ L_i^{max}\right].$$

The reason that the above expression is an inequality (rather than an equality), is that the amount of bits to drop is chosen arbitrarily by the

flow controller, and in particular the controller might decide to drop all bits associated with a particular packet in the case when a complete packet does not fit into the storage reservoir. The storage buffer size L_i^{max} is arbitrary, possibly zero. In the case when $L_i^{max} = 0$, all data that is not immediately admitted to the network layer is necessarily dropped. In cases when $L_i^{max} > 0$, the flow controller must make additional decisions about which data to drop whenever appropriate.

In Sections 3–4 we shall find it useful to neglect flow control decisions entirely, so that all arriving data is immediately admitted to the network layer and $R_i^{(c)}(t) = A_i^{(c)}(t)$ for all timeslots t. In this case we say the flow controllers are "turned off." This action of "turning off" the flow controllers is only used as a thought experiment to build understanding of network layer routing and stability issues. In practice, turning off the flow controllers can lead to instability problems in cases when network traffic exceeds network capabilities, and these issues are considered in detail in Sections 5–6 when flow control is again integrated into the problem formulation.

2.4 Discussion of the assumptions

In this section we discuss the assumptions stated previously about the network model and its mode of operation.

2.4.1 The time slot assumption

Timeslots are used to facilitate analysis and to cleanly represent periods corresponding to new channel conditions and control actions. However, this assumption presumes synchronous operation, where control actions throughout the network take place according to a common timeclock. Although asynchronous networking is not formally considered in this text, the timescale expansion and approximate scheduling results of [111, 115, 134] suggest that the algorithms and analysis developed here can be extended to systems with independent network components that operate on their own timescales. Asynchronous systems are further explored in [26].

The assumption that channels hold their states for the duration of a timeslot is clearly an approximation, as real physical systems do not

conform to fixed slot boundaries and may change continuously. This approximation is valid in cases where slots are short in comparison to the speed of channel variation. In a wireless system with predictable slow fading and non-predictable fast fading [23, 105], the timeslot is assumed short in comparison to the slow fading (so that a given measurement or prediction of the fade state lasts throughout the timeslot) and long in comparison to the fast fading (so that a transmission of many symbols encoded with knowledge of the slow fade state and the fast-fade statistics can be successfully decoded with sufficiently low error probability).

2.4.2 Channel measurement

We assume that network components have the ability to monitor channel quality so that intelligent control decisions can be made. This measurement can be in the form of a specific set of attenuation coefficients, or can be according to a simple channel classification such as "Good," "Medium," "Bad." Channel measurement technology is currently being implemented for cellular communication with High Data Rate (HDR) services [63], and the ability to measure and react to channel information is expected to improve significantly.[4] In systems where it is difficult to obtain timely feedback about channel quality, such as satellite systems with long round-trip times, channel measurement can be combined with channel prediction. Accurate channel prediction schemes for satellites are developed in [32, 33, 69].

2.4.3 The error-free transmission assumption

All data transmissions from one node to the next are considered to be successful with sufficiently high probability. For example, the link budget curves for wireless transmissions could be designed so that decoding errors occur with probability less than 10^{-6}. In such a system, there must be some form of error recovery protocol which allows a source to re-inject lost data back into the network [18]. If transmission errors

[4] Indeed, it is claimed in [152] that channel measurements can be obtained almost as often as the symbol rate of the link in certain local area wireless networks.

are rare, the extra arrival rate due to such errors is small and does not appreciably change network performance. Throughout this text, we neglect such errors and treat all transmissions as if they are error-free. An alternate model in which transmissions are successful with a given probability can likely be treated using similar analysis. Recent work in [74, 75] considers channel uncertainty for transmission scheduling in MIMO systems, and work in [118] considers routing in multi-hop networks with unreliable channels and multi-receiver diversity.

3

Stability and Network Capacity

Here we establish the fundamental throughput limitations of a general multi-commodity network as defined in the previous section. Specifically, we characterize the *network layer capacity region*. This region describes the set of traffic rates that the network can stably support, considering all possible strategies for choosing the control decision variables that affect routing, scheduling, and resource allocation. We begin with a precise definition of stability for single queues and for queueing networks.

3.1 Queue stability

Consider a single queue with an input process $A(t)$ and transmission rate process $\mu(t)$, where $A(t)$ represents the amount of new arrivals that enter the queue during slot t, and $\mu(t)$ represents the transmission rate of the server during slot t. We assume that the $A(t)$ arrivals occur at the end of slot t, so that they cannot be transmitted during that slot. Let $U(t)$ represent the current backlog in the queue. The $U(t)$ process evolves according to the following discrete time queueing law:

$$U(t+1) = \max[U(t) - \mu(t), 0] + A(t).$$

The queue might be located within a larger network, in which case the arrival process $A(t)$ is composed of random exogenous arrivals as well as endogenous arrivals resulting from routing and transmission decisions from other nodes of the network. Likewise, the transmission rate $\mu(t)$ can be determined by a combination of random channel state variations and controlled network resource allocations, both of which can change from slot to slot. Therefore, it is important to develop a general definition of queueing stability that handles arbitrary $A(t)$ and $\mu(t)$ processes.

Definition 3.1. A queue is called *strongly stable* if:

$$\limsup_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t-1}\mathbb{E}\{U(\tau)\} < \infty.$$

That is, a queue is strongly stable if it has a bounded time average backlog. This leads to a natural definition of network stability:

Definition 3.2. A *network is strongly stable* if all individual queues of the network are strongly stable.

A discussion of more general stability definitions can be found in [12, 43, 58, 111, 115]. Throughout this text we shall restrict attention to the strong stability definition given above, and shall often use the term "stability" to refer to strong stability. The following simple but important necessary condition holds for strongly stable queues with any arbitrary arrival and server processes (possibly without well defined time averages). Its proof can be found in [122].

Lemma 3.3. (Necessary Condition for Strong Stability) If a queue is strongly stable and either $\mathbb{E}\{A(t)\} \leq A_{\max}$ for all t, or $\mathbb{E}\{\mu(t) - A(t)\} \leq D_{\max}$ for all t, where $A_{\max}$, $D_{\max}$ are finite nonnegative constants, then:

$$\lim_{t\to\infty} \frac{\mathbb{E}\{U(t)\}}{t} = 0. \tag{3.1}$$

3.1.1 The arrival process assumptions

To analyze network capacity, we assume that all exogenous arrival processes $A_i^{(c)}(t)$ satisfy the following structural properties for *admissible inputs*.

Definition 3.4. An arrival process $A(t)$ is *admissible with rate* λ if:

- The time average expected arrival rate satisfies:
$$\lim_{t\to\infty}\frac{1}{t}\sum_{\tau=0}^{t-1}\mathbb{E}\{A(\tau)\}=\lambda.$$
- There exists a finite value A_{max} such that $\mathbb{E}\left\{A(t)^2\,|\boldsymbol{H}(t)\right\}\leq A_{max}^2$ for all timeslots t, where $\boldsymbol{H}(t)$ represents the history up to time t, i.e., all events that take place during slots $\tau\in\{0,\ldots,t-1\}$.
- For any $\delta>0$, there exists an interval size T (that may depend on δ) such that for any initial time t_0 the following condition holds:
$$\mathbb{E}\left\{\frac{1}{T}\sum_{k=0}^{T-1}A(t_0+k)\,|\boldsymbol{H}(t_0)\right\}\leq\lambda+\delta. \tag{3.2}$$

Some examples of admissible arrival processes are the following.

Example 3.1. Let $X(t)$ be an ergodic Markov chain with a finite state space $\{1,\ldots,Q\}$. When $X(t)=m$, let $A(t)$ be chosen independently and identically distributed (i.i.d.) with distribution $P_A^{(m)}(a)$. If $\pi_m,\ m\in\{1,...,Q\}$ is the steady state distribution of $X(t)$ and $\mathbb{E}\{A(t)\,|X(t)=m\}=\lambda_m$, then the process $A(t)$ is admissible with rate $\lambda=\sum_{m=1}^{Q}\lambda_m\pi_m$. An important special case is when there is only one state, so that $A(t)$ is i.i.d. every slot with $\mathbb{E}\{A(t)\}=\lambda$ for all t.

Example 3.2. Let $A(t)$ satisfy the following burstiness constraints

$$\lambda(t_2-t_1)+\sigma_1\geq\sum_{t=t_1}^{t_2-1}A(t)\geq\lambda(t_2-t_1)-\sigma_2,\ \text{for all } t_2>t_1\geq 0,$$

where σ_1 and σ_2 are nonnegative numbers. Then $A(t)$ is admissible with rate λ. Burstiness constrained models have been used extensively in wired networks [27, 35, 82, 148] and in a wireless context in [157].

Below we define the concept of an *admissible service process* $\mu(t)$:

Definition 3.5. A server process $\mu(t)$ is *admissible with time average service rate* $\overline{\mu}$ if:

- The time average expected service rate satisfies:
$$\lim_{t\to\infty}\frac{1}{t}\sum_{\tau=0}^{t-1}\mathbb{E}\left\{\mu(\tau)\right\} = \overline{\mu}.$$
- There exists an upper bound μ_{max} such that $\mu(t) \leq \mu_{max}$ for all t.
- For any $\delta > 0$, there exists an interval size T (that may depend on δ), such that for any initial time t_0 the following condition holds:
$$\mathbb{E}\left\{\frac{1}{T}\sum_{k=0}^{T-1}\mu(t_0+k)\,|\boldsymbol{H}(t_0)\right\} \geq \overline{\mu} - \delta. \tag{3.3}$$

Lemma 3.6. (Stability Conditions under Admissibility) Consider a queue with an admissible input process $A(t)$ with arrival rate λ, and an admissible server process with time average rate $\overline{\mu}$. Then: (a) $\lambda \leq \overline{\mu}$ is a necessary condition for strong stability. (b) $\lambda < \overline{\mu}$ is a sufficient condition for strong stability.

The necessary condition is quite intuitive. Indeed, if $\lambda > \overline{\mu}$, then expected queue backlog necessarily grows to infinity, leading to instability. The sufficient condition is also intuitive, but its proof requires the structure of admissible arrival and service processes as defined above (see [115] for a proof). We note that strong stability also holds in cases when the infinite horizon time average conditions for $A(t)$ and $\mu(t)$

do not necessarily hold, but these processes satisfy all other inequality conditions of the admissibility definitions (for some values λ and $\overline{\mu}$ such that $\lambda < \overline{\mu}$). We say that such an arrival process is *admissible with arrival rate less than or equal to* λ, and such a service process is *admissible with average service rate greater than or equal to* $\overline{\mu}$.

3.2 The network layer capacity region

Consider a network with a general link transmission rate matrix $\boldsymbol{C}(I(t), S(t)) = (C_{ab}(I(t), S(t)))$. Recall that $I(t) \in \mathcal{I}_{S(t)}$ and $S(t) \in \mathcal{S}$, where $\mathcal{I}_{S(t)}$ is the control space for a given topology state $S(t)$, and $\mathcal{S}$ is the finite set of all possible topology states for the network. The function $\boldsymbol{C}(\cdot, \cdot)$ is arbitrary (possibly discontinuous) and is only assumed to be *bounded*, so that for all links (a, b) and all $I \in \mathcal{I}_s$ and $s \in \mathcal{S}$ we have:

$$0 \leq C_{ab}(I, s) \leq \mu_{max}. \tag{3.4}$$

for some maximum transmission rate μ_{max}. The topology state $S(t)$ is assumed to evolve according to a finite state, irreducible Markov chain (possibly periodic). Such chains have well defined time averages π_s, representing the time average fraction of time that $S(t) = s$. Specifically, with probability 1 we have:[1]

$$\lim_{t\to\infty} \frac{1}{t} \sum_{\tau=0}^{t-1} 1_{[S(t)=s]} = \pi_s \ , \quad \text{for all } s \in \mathcal{S}. \tag{3.5}$$

where $1_{[S(t)=s]}$ is an indicator function that takes the value 1 whenever $S(t) = s$, and takes the value zero otherwise.

Let $\mathcal{N}$ and $\mathcal{K}$ represent the set of nodes and commodities, with sizes N and K respectively. Let $\mathcal{L}_c$ be the set of links defining the routing constraints for each commodity $c \in \mathcal{K}$. Define $U_i^{(c)}(t)$ as the internal queue backlog of commodity c data at node i. Due to the routing constraints, some commodities might never be able to visit certain nodes. Further, some nodes might only be associated with destinations, and hence these nodes do not keep any internal queues. Hence, we define K_i as the number of internal queues kept by node i, where

[1] The Markov structure for $S(t)$ is used only to facilitate presentation. Our results hold more generally for any $S(t)$ that satisfies the *channel convergent* property defined in [115].

$K_i \in \{0, 1, \ldots, K\}$. Define $\mathcal{D}$ as the set of all node-commodity pairs (i, c) associated with internal queues in the network, and let D represent the number of such queues:

$$D \triangleq \sum_{i=1}^{N} K_i.$$

The integer D defines the *relative dimension* of the network.

Let $A_i^{(c)}(t)$ represent the process of commodity c bits arriving exogenously to node i. Assume that these arrival processes are admissible with time average rates $\lambda_i^{(c)}$ (in units of bits/slot), and let $\boldsymbol{\lambda} = (\lambda_i^{(c)})$ represent the arrival rate matrix. We assume throughout that $\lambda_i^{(c)} = 0$ whenever $(i, c) \notin \mathcal{D}$, so that the number of non-zero rates is less than or equal to D. Assume that the network flow controllers are turned off, so that all incoming traffic arrives directly to the network layer.

Definition 3.7. The *network layer capacity region* Λ is the closure of the set of all arrival rate matrices $\left(\lambda_i^{(c)}\right)$ that can be stably supported by the network, considering all possible strategies for choosing the control variables to affect routing, scheduling, and resource allocation (including strategies that have perfect knowledge of future events).

Note that this is a network layer notion of capacity that considers all choices of the decision variables $\mu_{ab}^{(c)}(t)$ and $I(t)$ for a network that operates according to a given $\boldsymbol{C}(I(t), S(t))$ function; it was introduced in [143, 147] and generalized further in [111, 115, 149]. This is distinct from the *information theoretic* notion of network capacity, which includes optimization over all possible modulation and coding strategies and involves many of the unsolved problems of network information theory [34]. The network layer and information theoretic capacity regions are called "stability" and "capacity" regions respectively in [98], where a third notion of "throughput region" referring to the case when all nodes have infinite backlogs, is also examined. In this work, we shall use the term "capacity region" to refer to the network layer capacity region as described above.

The issue of capacity scaling was raised in [59] where it was found that the capacity vanishes asymptotically as the number of nodes

increases, with a rate that is inversely proportional to a fractional power of the number of nodes. The type of capacity considered in [59] is similar to the one we consider here. More specifically a certain model is considered for the local interaction of the radio transmissions, that imply interference restrictions on simultaneous transmissions of nodes with geographical proximity. Assuming a uniform end-to-end traffic load matrix then, the capacity can be specified by a scalar, i.e. the maximum traffic intensity that is sustainable under any transmission control and traffic forwarding policy. In [59] a bound to that capacity is obtained that vanishes inversely proportional to the square root of the number of nodes. That important result indicates that large scale ad-hoc wireless networks with flat architecture may only have limited usability as a general purpose communication infrastructure. It motivated a lot of follow up work generalizing the result in various ways, some indicative examples are [57, 60, 78, 90, 94, 112, 114, 151, 166].

3.2.1 Constructing the capacity region

To build intuition about the set Λ, we first consider the capacity region of a traditional wireline network with no time variation, such as the static network of Example 2.1 in Section 2. Such a network is described by a constant matrix (G_{ab}), where G_{ab} is the fixed rate at which data can be transferred over link (a,b), and $G_{ab} = 0$ if there is no physical link from node a to node b. The network capacity region in this case is described implicitly as the set of all arrival rate matrices $\left(\lambda_i^{(c)}\right)$ for which there exist *multi-commodity flow variables* $f_{ab}^{(c)}$ (for $a, b \in \mathcal{N}$ and $c \in \mathcal{K}$) that satisfy a set of flow conservation equations, and that additionally satisfy the link constraint $\sum_c f_{ab}^{(c)} \leq G_{ab}$ for all links (a,b). This constraint ensures that the total flow over any link does not exceed the link transmission rate. Intuitively, this coincides with the necessary and sufficient conditions for queue stability described in Lemma 3.6. Indeed, stability requires that the data arrival rate to any link is no more than the transmission rate of the link, regardless of whether data flows as a continuous fluid or as packetized units.

The capacity region of a general network differs from that of a static wireline network only in the description of the link constraint.

Indeed, first note that the time varying network topology requires link transmission capabilities to be defined in a time average sense, where the resulting transmission rate over a given link (a,b) is averaged over all possible topology states. Second, the resulting time average link rates are not fixed, but depend on the resource allocation policy for choosing $I(t) \in \mathcal{I}_{S(t)}$. Thus, instead of describing the network as a single graph with a single transmission rate matrix (G_{ab}), the network is described by a *collection* of graphs defined by a *graph family* Γ. The graph family Γ can be viewed as the set of all long-term transmission rate matrices (G_{ab}) that the network can be configured to support on the single-hop links connecting node pairs (a,b), and is defined as follows:

$$\Gamma \triangleq \sum_{s\in\mathcal{S}} \pi_s Conv\{\boldsymbol{C}(I,s)\,|I \in \mathcal{I}_s\}, \tag{3.6}$$

where addition and scalar multiplication of sets is used[2], and where $Conv\{\mathcal{A}\}$ represents the convex hull of the set $\mathcal{A}$. Specifically, $Conv\{\mathcal{A}\}$ is defined as the set of all finite weighted combinations $p_1a_1 + p_2a_2 + \ldots + p_ma_m$ of elements $a_i \in \mathcal{A}$ (where $\{p_i\}$ are nonnegative numbers summing to 1). Such weighted combinations are called *convex combinations*. To intuitively understand why the graph family Γ has the form given in (3.6), we note the following basic result from convex set theory:

Fact 1: If $\boldsymbol{\mu}$ is any random vector that takes values within some general set $\mathcal{A}$, then $\mathbb{E}\{\boldsymbol{\mu}\} \in Conv\{\mathcal{A}\}$ (assuming the expectation is well defined). □

Consider now the set $\mathcal{A}_s$ defined as the set of all transmission rate matrices possible under channel state s:

$$\mathcal{A}_s \triangleq \{\boldsymbol{C}(I,s)\,|I \in \mathcal{I}_s\}.$$

Two example sets $\mathcal{A}_s$, corresponding to two different topology states, are shown in Fig. 3.1. Suppose we have a resource allocation algorithm that every time slot independently chooses a random control

[2] For vector sets $\mathcal{A},\mathcal{B}$ and scalars α, β, the set $\alpha\mathcal{A} + \beta\mathcal{B}$ is defined as $\{\gamma\,|\,\gamma = \alpha a + \beta b:$ for some $a \in \mathcal{A}, b \in \mathcal{B}\}$.

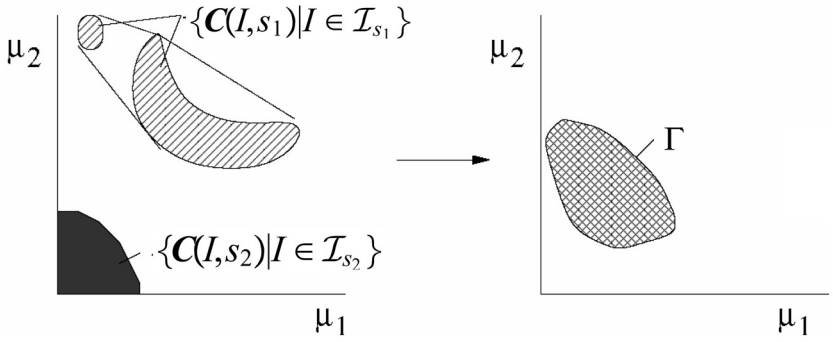

Fig. 3.1 A construction of the set Γ for the case of 2 dimensions, illustrating the set of all achievable long term link rates (μ_1, μ_2). In this example, we consider only two channel states s_1 and s_2, each equally probable. Note that for the first channel state, the set $\{\boldsymbol{C}(I,s)\,|I \in \mathcal{I}_s\}$ is disconnected and non-convex. Its convex hull is shown in the first plot. The secont plot illustrates the weighted sum of the convex hull of the regions associated with each of the two channel states. This is the Γ region, and is necessarily convex.

input $I(t) \in \mathcal{I}_s$ according to some probability law whenever $S(t) = s$, yielding a random rate matrix $\boldsymbol{\mu}(t) = \boldsymbol{C}(I(t), s)$. By definition, this random rate matrix satisfies $\boldsymbol{\mu}(t) \in \mathcal{A}_s$, and hence by Fact 1 it follows that the *expected* rate matrix satisfies: $\mathbb{E}\{\boldsymbol{\mu}(t)\,|S(t) = s\} \in Conv\{\mathcal{A}_s\}$. Thus, randomizing the control decisions allows the expected rate matrix to expand beyond the set $\mathcal{A}_s$ to reach points within the larger set $Conv\{\mathcal{A}_s\}$ (see Fig. 3.1). Further, by appropriately choosing the randomized probabilities, any point within the set Γ can be reached in this way. This is summarized in the following fact.

Fact 2: A matrix $\boldsymbol{G} = (G_{ab})$ is in the graph family Γ if and only if there exists a randomized policy that bases control decisions on the current channel state, such that:

$$\boldsymbol{G} = \sum_{s \in \mathcal{S}} \pi_s \mathbb{E}\{\boldsymbol{\mu}(t)\,|S(t) = s\}, \tag{3.7}$$

where $\mathbb{E}\{\boldsymbol{\mu}(t)\,|S(t) = s\}$ is the expected rate matrix offered by the randomized policy when $S(t) = s$. $\square$

By ergodicity of the topology state process $S(t)$ together with the law of large numbers, it is easy to see that the right hand side of (3.7) is an expression for the time average transmission rate $\lim_{t\to\infty} \frac{1}{t} \sum_{\tau=0}^{t-1} \boldsymbol{\mu}(\tau)$. Thus, the network can be configured to achieve

long term link transmission rates for any rate matrix $\boldsymbol{G}$ within the graph family Γ. Let $Cl\{\Gamma\}$ represent the *closure* of the set Γ. The following theorem from [111, 115] characterizes the network layer capacity region. Recall that every timeslot, a network controller must choose $I(t) \in \mathcal{I}_{S(t)}$, and must choose routing variables $\mu_{ab}^{(c)}(t)$ that satisfy: $\sum_c \mu_{ab}^{(c)}(t) \leq C_{ab}(I(t), S(t))$, $\mu_{ab}^{(c)}(t) = 0$ if $(a,b) \notin \mathcal{L}_c$.

Theorem 3.8. (Capacity Region for a Network) The capacity region of the network is given by the set Λ consisting of all input rate matrices $\left(\lambda_i^{(c)}\right)$ such that $\lambda_i^{(c)} = 0$ whenever $(i,c) \notin \mathcal{D}$, and such that there exists a rate matrix $(G_{ab}) \in Cl\{\Gamma\}$ together with multi-commodity flow variables $\left\{f_{ab}^{(c)}\right\}$ satisfying:

(Flow Efficiency Constraints)

$$f_{ab}^{(c)} \geq 0, \quad f_{aa}^{(c)} = f_{dest(c),b}^{(c)} = 0, \quad \text{for all } a,b \in \mathcal{N}, c \in \mathcal{K}, \tag{3.8}$$

(Flow Conservation Constraints)

$$\lambda_i^{(c)} = \sum_b f_{ib}^{(c)} - \sum_a f_{ai}^{(c)}, \quad \text{for all } (i,c) \in \mathcal{D} \text{ with } i \neq dest(c), \tag{3.9}$$

(Routing Constraints)

$$f_{ab}^{(c)} = 0, \quad \text{for all } a,b \in \mathcal{N},\ c \in \mathcal{K} \text{ with } (a,b) \notin \mathcal{L}_c, \tag{3.10}$$

(Link Constraints)

$$\sum_c f_{ab}^{(c)} \leq G_{ab}, \quad \text{for all } a,b \in \mathcal{N}, \tag{3.11}$$

where $dest(c)$ represents the destination node for commodity c data.

Thus, a rate matrix $\left(\lambda_i^{(c)}\right)$ is in the capacity region Λ if there exists a matrix $(G_{ab}) \in Cl\{\Gamma\}$ that defines link capacities in a traditional graph network, such that there exist multi-commodity flow variables $\left\{f_{ab}^{(c)}\right\}$ which support the $\left(\lambda_i^{(c)}\right)$ rates with respect to this graph. Note that inequalities (3.8) constrain flow variables to be non-negative and to be "efficient," in that no node transmits data to itself and no node

re-injects delivered data back into the network. Inequality (3.9) is a conservation constraint that ensures the total flow of commodity c data into a given node i is less than or equal to the total flow out of that node, provided that node i is not the destination. We note that (3.9) is expressed as an equality constraint only to facilitate understanding. The same theorem holds if (3.9) is replaced by the following inequality constraint:

$$\lambda_i^{(c)} \leq \sum_b f_{ib}^{(c)} - \sum_a f_{ai}^{(c)} \quad \text{for all } (i,c) \in \mathcal{D} \text{ with } i \neq dest(c).$$

The above constraint is more useful because it leads to a simpler proof of the theorem (see [111, 115]), and also simplifies construction of dual algorithms for finding the multi-commodity flows $\left\{f_{ab}^{(c)}\right\}$ in the case when the set $Cl\{\Gamma\}$ is known in advance and the problem is treated as a convex program (see Section 4.10).

The following useful corollary establishes an important property of the capacity region:

Corollary 3.9. If $\Gamma = Cl(\Gamma)$ and if the topology state $S(t)$ is i.i.d. from slot to slot, then a rate matrix $\left(\lambda_i^{(c)}\right)$ is within the capacity region Λ if and only if there exists a stationary randomized control algorithm that makes valid $\mu_{ab}^{(c)}(t)$ decisions based only on the current topology state $S(t)$, and that yields for all (i,c) and all time t:

$$\mathbb{E}\left\{\sum_b \mu_{ib}^{(c)}(t) - \sum_a \mu_{ai}^{(c)}(t)\right\} = \lambda_i^{(c)}, \text{ for all } (i,c) \in \mathcal{D} \text{ with } i \neq dest\,(c)\,,$$

where the expectation is taken with respect to the random topology state $S(t)$ and the (potentially) random control action based on this state.

The constraints of Theorem 3.8 lead to the following structural properties for Λ:

- The set Λ is convex, closed, and bounded [17, 115].
- The set Λ contains the all-zero matrix (so that $\mathbf{0} \in \Lambda$).

- If $\boldsymbol{\lambda} \in \Lambda$, then $\tilde{\boldsymbol{\lambda}} \in \Lambda$, where $\tilde{\boldsymbol{\lambda}}$ is any rate matrix that is entrywise less than or equal to $\boldsymbol{\lambda}$.

Let $\widetilde{\mathcal{D}}$ represent the subset of $\mathcal{D}$ consisting of all node-commodity pairs (i,c) for which it is possible to stably support a non-zero input rate $\lambda_i^{(c)}$ (assuming that all other input rates are zero). The *relative interior* of the set Λ is given by the set of all rate matrices $\boldsymbol{\lambda}$ for which there exists an $\epsilon > 0$ such that $\boldsymbol{\lambda} + \boldsymbol{\epsilon} \in \Lambda$, where $\boldsymbol{\epsilon}$ is a matrix with entries $\epsilon_i^{(c)} = \epsilon$ for all $(i,c) \in \widetilde{\mathcal{D}}$, and all other entries equal to zero. The proof of Theorem 3.8 involves showing that $\left(\lambda_i^{(c)}\right) \in \Lambda$ is necessary for stability, and that $\left(\lambda_i^{(c)}\right)$ within the relative interior of Λ is sufficient. Note that although the exogenous arrival processes are assumed to be admissible, the capacity region must capture *all possible routing, scheduling, and resource allocation strategies*, including strategies that result in non-admissible arrival or service processes at the individual queues of the network. In the next section, we prove the sufficient conditions of Theorem 3.8 for the special case of a one-hop network. For the complete proof for the general multi-hop case, the reader is referred to [111, 115].

3.3 The capacity of one hop networks

Consider the special case of a one-hop network with L different exogenous input processes. For simplicity, assume that data from each input process is intended for transmission over a unique link. Let $\boldsymbol{\mu}(t) = (\mu_1(t), \ldots, \mu_L(t))$ represent the vector of link transmission rates during slot t, where $\mu_l(t) = C_l(I(t), S(t))$ denotes the rate over link l under control input $I(t)$ and topology state $S(t)$. The corresponding graph family Γ defined in (3.6) is thus a set of rate *vectors* rather than rate *matrices*. Let $\boldsymbol{A}(t) = (A_1(t), \ldots, A_L(t))$ represent the vector of exogenous arrivals, where $A_l(t)$ is the number of bits that arrive to link l during slot t (for $l \in \{1, \ldots, L\}$). Assume these processes are admissible with rate vector $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_L)$. We have the following corollary to Theorem 3.8.

Corollary 3.10. (Single Hop Capacity Region) The *single hop capacity region* Λ consists of all rate vectors $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_L)$ such that there exists a vector $(G_1, \ldots, G_L) \in Cl\{\Gamma\}$ such that $\lambda_l \leq G_l$ for all network links $l \in \{1, \ldots, L\}$.

We skip the proof of necessity of this corollary and concentrate on the proof of sufficiency since this is important for the subsequent development and demonstrates the design issues involved.

Proof. (Sufficiency) Suppose for simplicity that Λ has full dimension. Suppose each link $l \in \{1, \ldots, L\}$ receives an admissible input process of rate λ_l. Let $\boldsymbol{\lambda}$ represent the input rate vector. Assume that $\boldsymbol{\lambda}$ is in the interior of the set Λ defined in Corollary 3.10, so that there exists an $\epsilon > 0$ such that $(\lambda_1 + \epsilon, \ldots, \lambda_L + \epsilon) \in \Lambda$. Thus, there exists a vector $\overline{\boldsymbol{G}} = (\overline{G}_1, \ldots, \overline{G}_L)$ such that $\overline{\boldsymbol{G}} \in Cl\{\Gamma\}$ and $\lambda_l + \epsilon \leq \overline{G}_l$ for all links $l \in \{1, \ldots, L\}$. Then there must exist a matrix $\boldsymbol{G} = (G_1, \ldots, G_L)$ such that $\boldsymbol{G} \in \Gamma$ and $\lambda_l + \epsilon/2 \leq G_l$ for all links l. Because $\boldsymbol{G} \in \Gamma$, it can be written as:

$$\boldsymbol{G} = \sum_{s \in \mathcal{S}} \pi_s \boldsymbol{G}_s, \tag{3.12}$$

where $\boldsymbol{G}_s \in Conv\{\boldsymbol{C}(I, s) \,|\, I \in \mathcal{I}_s\}$ for each channel state $s \in \mathcal{S}$. Furthermore, by Caratheodory's Theorem [17], each matrix $\boldsymbol{G}_s$ can be decomposed into a convex combination of at most $L + 1$ elements of $\{\boldsymbol{C}(I, s) \,|\, I \in \mathcal{I}_s\}$:

$$\boldsymbol{G}_s = p_s^1 \boldsymbol{C}(I_s^1, s) + p_s^2 \boldsymbol{C}(I_s^2, s) + \ldots + p_s^{L+1} \boldsymbol{C}(I_s^{L+1}, s), \tag{3.13}$$

where $I_s^i \in \mathcal{I}_s$ for all $s \in \mathcal{S}$ and $i \in \{1, \ldots, L + 1\}$. Given such a decomposition for each channel state $s \in \mathcal{S}$, the following control algorithm can be constructed: On each timeslot, observe the current network topology state $S(t)$. Given that $S(t) = s$, randomly choose one of the $L + 1$ control options I_s^i with probability p_s^i (for $i \in \{1, \ldots, L + 1\}$). It follows that:

$$\mathbb{E}\{\boldsymbol{\mu}(t) \,|\, S(t) = s\} = \mathbb{E}\{\boldsymbol{C}(I(t), S(t)) \,|\, S(t) = s\} = \boldsymbol{G}_s. \tag{3.14}$$

This strategy results in a matrix $\boldsymbol{\mu}(t)$ consisting of individual service rate processes $\mu_l(t)$ for each link l. *Claim:* If $S(t)$ evolves according to a finite state irreducible Markov chain, then each service process $\mu_l(t)$ is admissible with time average rate $\overline{\mu}_l = G_l$. A formal proof of the claim is given in [115]. Intuitively, the claim holds because the time average of the $\boldsymbol{\mu}(t)$ process can be expressed as a sum over the steady state topology state probabilities:

$$\lim_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t-1} \boldsymbol{\mu}(\tau) = \sum_{s\in\mathcal{S}} \pi_s \mathbb{E}\{\boldsymbol{\mu}(t)\,|S(t)=s\} \quad \text{with probability 1,}$$

and hence $\overline{\boldsymbol{\mu}} = \boldsymbol{G}$ by (3.12) and (3.14). Thus, the queue for each link l has an admissible input with rate λ_l and an admissible service process with time average rate $\overline{\mu}_l = G_l$, where $\overline{\mu}_l$ is strictly larger than λ_l. From the sufficient condition of Lemma 3.6 in Section 3.1.1, it follows that each queue of the network is stable, proving the result. □

The above stabilizing algorithm is not intended as a practical control strategy, as it cannot be implemented without extensive offline preparation. Indeed, the input rate matrix and the network capacity region would need to be known in advance, which requires a-priori knowledge of the topology state probabilities π_s for each state $s \in \mathcal{S}$. Further, assuming all of the probabilities could be accurately estimated, the network controller would still need to pre-compute the decomposition of $\boldsymbol{G}_s$ in (3.13) for each possible topology state. As the number of states can grow geometrically in the number of network links, a direct attempt to implement the above policy would be very difficult even for a relatively small network. However, the fact that the above policy *exists* plays a crucial role in the analysis of a more practical stabilizing strategy presented in Section 4.

Example 3.3. *The capacity of ON-OFF downlink.* Consider a simple example of a two-queue wireless downlink that transmits data to two downlink users 1 and 2 over two different channels. Assume that the arrival processes are independent *Bernoulli processes* with rates λ_1 and λ_2, so that every timeslot a single packet independently arrives to queue i with probability λ_i, and no packet arrives to queue i otherwise

(for $i \in \{1,2\}$). All packets are assumed to have fixed lengths, so that the queue backlog is measured in units of packets. Let $U_1(t)$ and $U_2(t)$ respectively represent the current backlog of packets waiting for transmission to user 1 and user 2, respectively. Channels independently vary between ON and OFF states every slot according to independent Bernoulli processes with ON probabilities p_1 and p_2. Every timeslot, a controller observes the channel states and chooses to transmit over either channel 1 or channel 2. We assume that a single packet can be transmitted if a channel is ON, and no packet can be transmitted when a channel is OFF, so that the only decision is which channel to serve when both channels are ON. In this case, there are only four possible topology states $S(t)$, and the graph family Γ is given as the following two dimensional set of rate pairs (g_1, g_2), expressed as a sum of sets as in (3.6):

$$\begin{aligned}\Gamma = (1-p_1)(1-p_2)\{(0,0)\} + p_1(1-p_2)\{(1,0)\} \\ + p_2(1-p_1)\{(0,1)\} + p_1p_2 Conv\{(1,0),(0,1)\}.\end{aligned} \tag{3.15}$$

It can be verified that the resulting capacity region of this system, characterized by Corollary 3.10, is given by the set of all non-negative rate vectors (λ_1, λ_2) satisfying the following three inequalities (in addition to $\lambda_1 \geq 0,\ \lambda_2 \geq 0$):

$$\lambda_1 \leq p_1, \quad \lambda_2 \leq p_2, \quad \lambda_1 + \lambda_2 \leq p_1 + p_2(1-p_1). \tag{3.16}$$

This is a polyhedral region, where the set Γ forms the dominant face of the polyhedron (See Figure 4.1 in the next section). That these three inequalities are *necessary* for stability is quite intuitive: The first two inequalities bound the individual input rates λ_i in terms of the maximum possible average transmission rates of their respective queues, and the last inequality bounds the sum input rate in terms of the average sum transmission rate. This simple capacity expression arises from the special ON/OFF structure of the system. In [144], it is shown that any downlink with L independent ON/OFF channels has a capacity region that is given by a set of $2^L - 1$ inequalities: Each inequality corresponds

to a subset of channels and indicates that the sum input rate into this subset is less than or equal to the probability that at least one channel within the subset is ON. A similar structure for the capacity region holds when there are burstiness constraints on the channel state processes instead of the Bernoulli process assumption made above [157].

4

Dynamic Control for Network Stability

In this section we develop a general algorithm for stabilizing networks without requiring knowledge of the arrival rates or topology state probabilities. Unlike the offline algorithm considered in the previous section, which makes randomized resource allocation decisions based only on the observed topology state, the algorithm in this section is an *online dynamic algorithm* that bases decisions both on the observed topology state and on the current queue backlogs. We begin with a motivating example that illustrates the design challenges.

4.1 Scheduling in an ON/OFF downlink

Consider the ON/OFF downlink of Example 3.3, with arrival rates λ_1, λ_2 and independent Bernoulli channels with ON probabilities p_1 and p_2, and assume that $p_1 < p_2$. Recall that the capacity region of the system is given by the three inequality constraints in (3.16). While the controller is constrained to serving only a single queue in any given timeslot, the fact that there are two independent channels creates a *multi-user diversity gain*, creating a larger probability that at least one of the channels is ON during any particular timeslot. This gain

is evident in the inequalities (3.16) that describe the capacity region, where it is clear that the sum output rate of the system can be larger than the output rate of any single queue alone. However, this diversity gain is mitigated on timeslots in which one of the queues is empty. Therefore, even for this simple system, scheduling must be done carefully in order to ensure stability.

For example, assuming the arrival rates are interior to the capacity region, one might suspect that stability can be achieved simply by serving any non-empty ON queue. However, in the case when the choice is to serve an ON queue with two packets versus an ON queue with twenty packets, serving the shorter queue can potentially create a higher probability that this queue is empty in the near future, leading to a loss of multi-user diversity and creating potential instability. This phenomenon holds also for multi-rate systems: Choosing the non-empty queue with the largest offered transmission rate can lead to instability and sub-optimal throughput, even though this policy would maximize throughput in the special case when all queues are "infinitely backlogged" and always have packets to send. To illustrate this point, we compare three well known scheduling algorithms applied to the special case of the two queue downlink with ON/OFF channels: The Borst algorithm [24], the 'proportionally fair' Max μ_i/r_i algorithm [153], and the Max Weight Match (MWM) policy [144].[1]

The Borst Algorithm: The Borst algorithm chooses to serve the non-empty channel i with the largest $\tilde{\mu}_i(t)/\mathbb{E}\{\tilde{\mu}_i\}$ index, where $\tilde{\mu}_i(t)$ is the current rate offered by link i if this link is chosen for transmission, and $\mathbb{E}\{\tilde{\mu}_i\}$ is the expected value of this rate taken over its steady state distribution (which is assumed to be known a-priori). This algorithm is shown in [24] to have desirable fairness properties for wireless networks with an "infinite" number of channels, where each incoming packet is destined for a unique user with its own channel. Although the algorithm was not designed for a two-queue downlink, it is closely related to the Max μ_i/r_i policy (to be described below), and it is illuminating to evaluate its performance in this context. Applied to the two-queue

[1] The MWM policy is also called the Longest Connected Queue (LCQ) policy for this special case of ON/OFF channels.

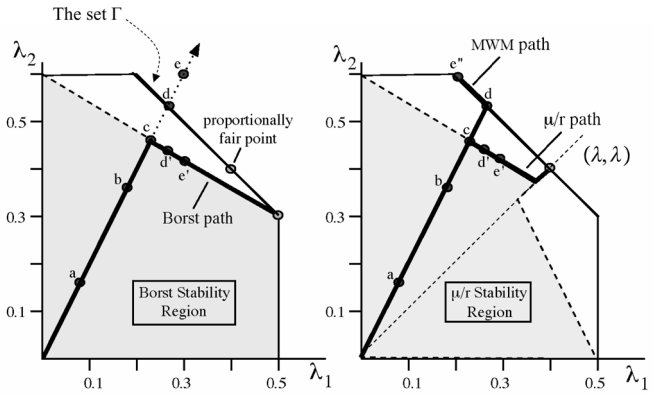

Fig. 4.1 The downlink capacity region Λ and the stability regions of the Borst policy and the Max μ_i/r_i policy. Input rates (λ_1, λ_2) are pushed toward point $(0.5, 1.0)$, and the simulated throughputs under the Borst, Max μ_i/r_i, and MWM policies are illustrated.

downlink, the Borst algorithm reduces to serving the non-empty ON queue with the largest value of $1/p_i$. Because $p_1 < p_2$, this algorithm effectively gives packets destined for channel 1 strict priority over channel 2 packets. It is not difficult to show that, for this simple system, the stability region of the Borst policy is given by the set of all rate pairs (λ_1, λ_2) such that $\lambda_1 \leq p_1$, $\lambda_2 \leq p_2 - \lambda_1 p_2$ [108]. This is a strict subset of the capacity region (see Fig. 4.1).

The Max μ_i/r_i Algorithm: Consider now the related policy of serving the non-empty queue with the largest value of $\tilde{\mu}_i(t)/r_i(t)$, where $r_i(t)$ is the empirical throughput achieved over channel i. This differs from the Borst algorithm in that transmission rates are weighted by the throughput actually delivered rather than the average transmission rate that is offered. This Max μ_i/r_i policy is proposed in [153] and shown to have desirable proportional fairness properties when all queues of the downlink are infinitely backlogged [81, 159]. To evaluate its performance for arbitrary traffic rates (λ_1, λ_2), suppose the running averages $r_1(t)$ and $r_2(t)$ are accumulated over the entire timeline, and suppose the system is stable so that $r_1(t)$ and $r_2(t)$ converge to λ_1 and λ_2. It follows that the algorithm eventually reduces to giving channel 1 packets strict priority if $\lambda_1 < \lambda_2$, and giving channel 2 packets strict priority if $\lambda_2 < \lambda_1$. Thus, if $\lambda_1 < \lambda_2$ then these rates must also

satisfy the inequalities $\lambda_1 \leq p_1$, $\lambda_2 \leq p_2 - \lambda_1 p_2$ of the Borst algorithm. However, if $\lambda_2 < \lambda_1$ then the rates must satisfy the inverted inequalities $\lambda_2 \leq p_2$ and $\lambda_1 \leq p_1 - \lambda_2 p_1$. Thus, at first glance it seems that the stability region of this policy is a subset of the stability region of the Borst algorithm. However, its stability region has the peculiar property of including all feasible rate pairs (λ, λ) (see Fig. 4.1).

The MWM Algorithm: The MWM algorithm serves the queue with the largest $U_i(t)\tilde{\mu}_i(t)$ index, where $U_i(t)$ is the current backlog in queue i. For this special case of an ON/OFF downlink, this policy reduces to serving the longest queue with an ON channel. The policy is shown in [144] to stabilize the system whenever the arrival rates are interior to the capacity region.

In Fig. 4.1 we consider the case when $p_1 = 0.5$, $p_2 = 0.6$, and plot the achieved throughput of the Borst, Max μ_i/r_i, and MWM policies when the rate vector (λ_1, λ_2) is scaled linearly towards the vector (0.5, 1.0), illustrated by the ray in Fig. 4.1(a). One hundred different rate points on this ray were considered (including example points a - e), and simulations were performed for each point over a period of 5 million timeslots. Fig 4.1(a) illustrates the resulting throughput of the Borst algorithm, where we have included example points d' and e' corresponding to input rate points d and e. Note that the Borst algorithm always results in throughput that is strictly interior to the capacity region, even when input rates are outside of capacity. Fig. 4.1(b) illustrates performance of the Max μ_i/r_i and MWM policies. Note that the MWM policy supports all (λ_1, λ_2) traffic when this rate vector is within the capacity region. However, when traffic is outside of the capacity region the achieved throughput moves along the boundary in the wrong direction, yielding throughputs that are increasingly "unfair" because it favors service of the higher traffic rate stream. Like the Borst policy, the Max μ_i/r_i policy leads to instability for all (stabilizable) input rates on the ray segment c-d, and yields throughput that is strictly interior to the capacity region even when inputs exceed system capacity (compare points e and e'). However, the throughput eventually touches the capacity region boundary at the "proportionally fair" point (0.4, 0.4) when input rates are sufficiently far outside of the capacity region.

This example illustrates two important points. First, the MWM algorithm provides stability whenever possible, while other reasonable algorithms may not. This first point is a special case of a general stability result for networks with arbitrary $\boldsymbol{C}(I(t), S(t))$ functions, developed in the Section 4.4 via a theory of *Lyapunov stability*. Second, the MWM policy does not necessarily offer *fairness* in cases when input rates exceed the capacity region. This issue is considered in Section 5, where our stabilizing algorithms are complemented with an optimal flow control technique via a theory of *Lyapunov optimization*.

4.2 Network model

We consider the general network model of Section 2, where there are N nodes and K commodities (with node and commodity sets $\mathcal{N}$ and $\mathcal{K}$, respectively). The network is characterized by:

- A topology state process $S(t)$ that evolves according to an irreducible Markov chain with a finite state space $\mathcal{S}$ and time average probabilities π_s for $s \in \mathcal{S}$.
- A control decision variable $I(t)$ (representing resource allocation) with a potentially topology state-dependent control space $\mathcal{I}_{S(t)}$.
- A matrix valued transmission rate function $\boldsymbol{C}(I(t), S(t)) = (C_{ab}(I(t), S(t)))$, where $C_{ab}(I(t), S(t))$ is the transmission rate over link (a, b) under the control action $I(t)$ and the topology state $S(t)$ (for $a, b \in \{1, \ldots, N\}$).

Recall that the $C_{ab}(I(t), S(t))$ functions are arbitrary, and are only assumed to be bounded. Define $\mu_{ab}^{(c)}(t)$ as the *routing control variables*, representing the amount of commodity c data delivered over link (a, b) during slot t. These routing variables are constrained as follows:

$$\sum_{c=1}^{K} \mu_{ab}^{(c)}(t) \leq C_{ab}(I(t), S(t)) \text{ for all } (a, b) \text{ and all } t, \tag{4.1}$$

$$\mu_{ab}^{(c)}(t) = 0 \text{ if } (a, b) \notin \mathcal{L}_c, \tag{4.2}$$

where $\mathcal{L}_c$ is the set of all links that are allowed to transmit commodity c data. Let $A_i^{(c)}(t)$ represent the process of exogenous commodity c data arriving to source node i (for $i \in \mathcal{N}$ and $c \in \mathcal{K}$). We assume that the flow controllers are turned off, so that all exogenous arrivals directly enter the network layer at their source nodes. Let $U_i^{(c)}(t)$ represent the backlog of commodity c data currently stored in node i. The queueing dynamics thus satisfy:

$$U_i^{(c)}(t+1) \leq \max[U_i^{(c)}(t) - \sum_b \mu_{ib}^{(c)}(t), 0] + A_i^{(c)}(t) + \sum_a \mu_{ai}^{(c)}(t). \tag{4.3}$$

Each process $A_i^{(c)}(t)$ is assumed to be admissible with rate less than or equal to $\lambda_i^{(c)}$. Define $\boldsymbol{\lambda} = (\lambda_i^{(c)})$ as the matrix of arrival rates. We assume that the input rate matrix is *stabilizable*, and in particular that is within the relative interior of the capacity region Λ. Recall from Section 3.2 that this means there exists a value $\epsilon > 0$ such that $\boldsymbol{\lambda} + \boldsymbol{\epsilon} \in \Lambda$, where:

- The set $\mathcal{D}$ contains all node-commodity pairs (i, c) for which there exist network queues $U_i^{(c)}(t)$.
- The set $\widetilde{\mathcal{D}}$ is the subset of $\mathcal{D}$ consisting of all node-commodity pairs (i, c) for which it is possible to support a non-zero rate $\lambda_i^{(c)}$ (assuming there is no other traffic).
- The matrix $\boldsymbol{\epsilon}$ has entries $\epsilon_i^{(c)} = \epsilon$ for all $(i, c) \in \widetilde{\mathcal{D}}$.

Thus, $\boldsymbol{\lambda} + \boldsymbol{\epsilon} \in \Lambda$ implies that it is possible to find multi-commodity flows to support the augmented traffic rate matrix associated with adding a new stream of rate ϵ to each of the source queues $U_i^{(c)}(t)$ (for $(i, c) \in \widetilde{\mathcal{D}}$). Data from any new stream of rate ϵ is simply treated as if it has the same commodity as the source queue it enters. To simplify analysis, we assume that $\mathcal{D} = \widetilde{\mathcal{D}}$. This is equivalent to the following assumption.

"No Trapping Nodes" Assumption: If it is possible to send commodity c data to a particular node i (so that a commodity c queue exists for that node), then it is possible to support a non-zero communication rate from node i to the destination of commodity c (possibly by using multi-hop paths).

For example, a "trapping node" might be a node with no outgoing links, or a node that is part of a group of nodes with outgoing links that only connect to other nodes of the group. Note that the "no trapping node" assumption holds whenever it is possible to deliver data back to the same node it came from. While this assumption is not required for the capacity theorem (Theorem 3.8), it shall be useful in analyzing the performance of the dynamic control policy introduced in the next subsection. Indeed, without this assumption, a general dynamic policy with no routing constraints might inadvertently send data to a trapping node that prevents this data from ever reaching its destination.

The dynamic policy introduced in the next section is most easily analyzed under the no trapping node assumption, although it can also be applied to achieve maximum throughput in networks without this assumption. In practice, if a particular node dies, or enough of its outgoing links die, then the node can become a trapping node. In this case, although all of the data contained in this node will be lost, the rate of adding more data to this node will approach zero, and so the dynamic routing policy simply finds alternate routes for all future data.

4.2.1 Input and output rate bounds

To analyze network performance, it is useful to define the maximum transmission rates out of and into a given node $i \in \mathcal{N}$ as follows:

$$\mu_{max,i}^{out} \triangleq \sup_{[s\in\mathcal{S}, I\in\mathcal{I}_s]} \sum_{b=1}^{N} C_{ib}(I,s), \quad \mu_{max,i}^{in} \triangleq \sup_{[s\in\mathcal{S}, I\in\mathcal{I}_s]} \sum_{a=1}^{N} C_{ai}(I,s).$$

Finite values of the above constants exist because the $\boldsymbol{C}(I(t), S(t))$ function is bounded. To simplify network analysis, it is also useful to assume that the total exogenous arrivals to any node $i \in \mathcal{N}$ are deterministically bounded by constants A_i^{max}, so that for all t we have:[2]

$$\sum_{c\in\mathcal{K}} A_i^{(c)}(t) \leq A_i^{max}$$

[2] The deterministic arrival bound is not necessary, and in [111, 115] the network is analyzed under the general definition of admissible inputs, which assumes only a bound on the second moment of exogenous arrivals.

4.3 The stabilizing dynamic backpressure algorithm

We describe below an algorithm for resource allocation and routing which, as will be shown, stabilizes the network whenever the vector of arrival rates lies within the capacity region of the network. The notion of controlling the system to maximize its stability region and the following algorithm that achieves it was introduced in [143, 147] and generalized further in [111, 115, 149].

The Dynamic Backpressure and Resource Allocation Algorithm: Every timeslot t, the network controller observes the queue backlog matrix $\boldsymbol{U}(t) = \left(U_i^{(c)}(t)\right)$ and the topology state variable $S(t)$ and performs the following actions for routing and resource allocation.

Resource Allocation: For each link $(a,\ b)$, define the *optimal commodity* $c_{ab}^*(t)$ as the commodity that maximizes the differential backlog (ties broken arbitrarily):

$$c_{ab}^*(t) \triangleq \arg \max_{\{c|(a,b)\in\mathcal{L}_c\}} \left[U_a^{(c)}(t) - U_b^{(c)}(t)\right],$$

and define $W_{ab}^*(t)$ as the corresponding optimal weight:

$$W_{ab}^*(t) \triangleq \max[U_a^{(c_{ab}^*(t))}(t) - U_b^{(c_{ab}^*(t))}(t), 0]. \tag{4.4}$$

Choose the control action $I(t)$ that solves the following optimization:

$$\begin{aligned} &\text{Maximize: } \sum_{ab} W_{ab}^*(t) C_{ab}(I(t), S(t)), \\ &\text{Subject to: } I(t) \in \mathcal{I}_{S(t)}. \end{aligned} \tag{4.5}$$

Routing: For each link (a,b) such that $W_{ab}^*(t) > 0$, offer a transmission rate of $\mu_{ab}(t) = C_{ab}(I(t), S(t))$ to data of commodity $c_{ab}^*(t)$. Recall that, by definition, $c_{ab}^*(t)$ is a valid commodity to send over link (a,b) (that is, $(a,b) \in \mathcal{L}_{c_{ab}^*(t)}$). If there is not enough data of commodity $c_{ab}^*(t)$ in node a to transmit over all outgoing links requesting this commodity, idle fill bits are transmitted, with an arbitrary allocation of actual data and idle fill data over the corresponding outgoing links.

The weights $W^*_{ab}(t)$ can be determined at each node provided that nodes are aware of the backlog sizes of their neighbors. However, the optimization problem (4.5) that must be solved at the beginning of each time slot requires in general knowledge of the whole network state. There are important special cases where this optimization problem can be solved in a distributed fashion with each node requiring knowledge only of the local state information on each of its outgoing channels. This issue will be described in Section 4.8. The resource allocation problem can also be approximated or optimized to within a constant factor using the schemes that will be described in Section 4.7.

The optimal routing commodities $c^*_{ab}(t)$ can be determined provided that nodes are aware of the backlog levels of their neighbors. Note that this routing strategy does not require paths to be specified in advance: Paths are chosen *dynamically* at each timestep according to the backpressure between neighboring nodes. The resulting algorithm assigns larger transmission rates to links with larger differential backlog, and zero transmission rates to links with negative differential backlog.

In the special case when the routing constraint sets $\mathcal{L}_c$ consist of all data links, then the above policy is equivalent to the Dynamic Routing and Power Control (DRPC) policy of [111] (where the $I(t)$ control variable used here plays the role of the $\boldsymbol{P}(t)$ power matrix from [111]). The DRPC policy itself is a generalization of the original backpressure algorithm developed for multi-hop packet radio networks in [143], where stable scheduling algorithms were developed using maximum weight activation sets with link weights equal to differential backlog.

Consider now the case of a single-hop network where a commodity c is associated with each link (a, b) traffic. As mentioned in Section 2.2.1 this case can be treated by simply setting $\mathcal{L}_c = \{(a, b)\}$. Taking into account that the traffic backlog at the destination node is considered zero, we have the following simplified algorithm in this case.

The Single-hop Dynamic Backpressure and Resource Allocation Algorithm: Every timeslot t, the network controller observes the queue backlog matrix $\boldsymbol{U}(t) = \left(U^b_a(t)\right)$ ($U^b_a(t)$ is the backlog at node a of traffic destined to node b) and the topology state variable $S(t)$ and performs the following actions for routing and resource allocation.

Resource Allocation: Choose the control action $I(t)$ that solves the following optimization:

$$\begin{aligned} \text{Maximize: } & \sum_{ab} U_a^b(t) C_{ab}(I(t), S(t)), \\ \text{Subject to: } & I(t) \in \mathcal{I}_{S(t)}. \end{aligned}$$

Each link then transmits with rate $\mu_{ab}(t) = C_{ab}(I(t), S(t))$.

We close this section by describing an important property of the proposed algorithm; in fact the algorithm is designed so that this property is satisfied. Consider any routing control variables $\tilde{\mu}_{ab}^{(c)}(t)$ that are admissible (i.e., satisfy (4.1) and (4.2)). Notice that by the definition of the Resource Allocation and Routing policy,

$$\begin{aligned} \sum_{ab} \sum_{c} \tilde{\mu}_{ab}^{(c)}(t) \left[U_a^{(c)}(t) - U_b^{(c)}(t) \right] &\leq \sum_{ab} \sum_{c} \tilde{\mu}_{ab}^{(c)}(t) W_{ab}^*(t) \\ &\leq \sum_{ab} W_{ab}^*(t) C_{ab}(I(t), S(t)), \end{aligned}$$

where the final inequality follows from the routing constraints (4.1) (4.2). Moreover, the upper bound above is *achievable* by the control policy that allocates resources to maximize the weighted sum of transmission rates $\sum_{ab} W_{ab}^*(t) C_{ab}(I(t), S(t))$ subject to $I(t) \in \mathcal{I}_{S(t)}$, and then chooses control variables:

$$\mu_{ab}^{(c)}(t) = \begin{cases} C_{ab}(I(t), S(t)) & \text{if } c = c_{ab}^*(t) \\ 0 & \text{otherwise} \end{cases} . \tag{4.6}$$

These are exactly the routing control variables defined by the Dynamic Backpressure Algorithm. Taking into account the following simple but important identity

$$\sum_{ic} U_i^{(c)}(t) \left[\sum_b \mu_{ib}^{(c)}(t) - \sum_a \mu_{ai}^{(c)}(t) \right] = \sum_{ab} \sum_{c} \mu_{ab}^{(c)}(t) \left[U_a^{(c)}(t) - U_b^{(c)}(t) \right],$$

we conclude from the above the following basic property:

Basic Property: If $\mu_{ab}^{(c)}(t)$ are the routing control variables defined by the Dynamic Backpressure and Resource Allocation Algorithm and

$\tilde{\mu}_{ab}^{(c)}(t)$ those of any other feasible algorithm, then for any time $t \geq 0$,

$$\sum_{ic} U_i^{(c)}(t) \left[\sum_b \tilde{\mu}_{ab}^{(c)}(t) - \sum_a \tilde{\mu}_{ab}^{(c)}(t) \right]$$

$$\leq \sum_{ic} U_i^{(c)}(t) \left[\sum_b \mu_{ib}^{(c)}(t) - \sum_a \mu_{ai}^{(c)}(t) \right]. \qquad (4.7)$$

In the next section we provide the basic tools for proving stability via Lyapunov function techniques, which are then used in Section 4.5 for proving the stability of the Dynamic Backpressure Algorithm.

4.4 Lyapunov stability

One of the most important mathematical tools for proving stability of queueing networks and for developing stabilizing control algorithms is the technique of *Lyapunov drift*. The idea is to define a non-negative function, called a *Lyapunov function*, as a scalar measure of the aggregate congestion of all queues in the network. Network control decisions are then evaluated in terms of how they affect the change in the Lyapunov function from one slot to the next.

Specifically, consider a network with L queues, and let $\boldsymbol{U}(t) = (U_1(t), \ldots, U_L(t))$ represent the vector process of backlog in each queue as a function of time. We define the following *quadratic Lyapunov function* $L(\boldsymbol{U})$:

$$L(\boldsymbol{U}) = \sum_{i=1}^{L} U_i^2.$$

Note that $L(\boldsymbol{U}(t)) = 0$ if and only if all network queues are empty at time t, and that $L(\boldsymbol{U}(t))$ is large whenever one or more components of $\boldsymbol{U}(t)$ is large.

Assume that $\boldsymbol{U}(t)$ evolves according to some probabilistic law, and that the initial conditions are such that $\mathbb{E}\{U_i(0)\} < \infty$ for all queues $i \in \{1, \ldots, L\}$.

Lemma 4.1. (Lyapunov Stability) If there exist constants $B > 0$, $\epsilon > 0$, such that for all timeslots t we have:

$$\mathbb{E}\{L(\boldsymbol{U}(t+1)) - L(\boldsymbol{U}(t))|\boldsymbol{U}(t)\} \leq B - \epsilon \sum_{i=1}^{L} U_i(t), \qquad (4.8)$$

then the network is strongly stable, and furthermore:

$$\limsup_{t\to\infty} \frac{1}{t} \sum_{\tau=0}^{t-1} \sum_{i=1}^{L} \mathbb{E}\{U_i(\tau)\} \leq \frac{B}{\epsilon}.$$

The expression on the left hand side of (4.8) is the *Lyapunov drift*, representing the expected change in the Lyapunov function from one slot to the next. With slight abuse of notation,[3] we shall often use $\Delta(\boldsymbol{U}(t))$ as a formal representation of this Lyapunov drift:

$$\Delta(\boldsymbol{U}(t)) \triangleq \mathbb{E}\{L(\boldsymbol{U}(t+1)) - L(\boldsymbol{U}(t))|\boldsymbol{U}(t)\}.$$

Note that if the condition (4.8) holds, then for any $\delta > 0$, the Lyapunov drift satisfies $\Delta(\boldsymbol{U}(t)) \leq -\delta$ whenever $\sum_{i=1}^{L} U_i(t) \geq (B+\delta)/\epsilon$. That is, the condition of the Lemma ensures that the Lyapunov drift is negative whenever the sum of queue backlogs is sufficiently large. Intuitively, this property ensures network stability because whenever the queue backlog vector leaves the bounded region

$$\left\{ \boldsymbol{U} \geq \boldsymbol{0} \middle| \sum_{i=1}^{L} U_i \leq (B+\delta)/\epsilon \right\},$$

the negative drift eventually drives it back to this region.

Proof. (Lemma 4.1) Assume the condition (4.8) holds for all timeslots t. Taking expectations of (4.8) (with respect to the distribution for the random queue backlog $\boldsymbol{U}(t)$ at time t) we have by the law of iterated expectations:

$$\mathbb{E}\{L(\boldsymbol{U}(t+1))\} - \mathbb{E}\{L(\boldsymbol{U}(t))\} \leq B - \epsilon \sum_{i=1}^{L} \mathbb{E}\{U_i(t)\}.$$

[3] Strictly speaking, the Lyapunov drift should be expressed as $\Delta(\boldsymbol{U}(t), t)$, as it could potentially depend on the timeslot t as well as the queue backlog values $\boldsymbol{U}(t)$. However, to simplify notation, we use $\Delta(\boldsymbol{U}(t))$ as a formal symbolic representation of the same thing.

The above inequality holds for all timeslots $t \in \{0, 1, 2, \ldots\}$. Summing the inequality over timeslots $\tau \in \{0, \ldots, M-1\}$ yields a *telescoping series* on the left hand side, resulting in:

$$\mathbb{E}\{L(\boldsymbol{U}(M))\} - \mathbb{E}\{L(\boldsymbol{U}(0))\} \leq BM - \epsilon \sum_{\tau=0}^{M-1} \sum_{i=1}^{L} \mathbb{E}\{U_i(\tau)\}.$$

Dividing the above inequality by M, shifting terms, and using the fact that $L(\boldsymbol{U}(M)) \geq 0$, we have:

$$\frac{1}{M} \sum_{\tau=0}^{M-1} \sum_{i=1}^{L} \mathbb{E}\{U_i(\tau)\} \leq \frac{B}{\epsilon} + \frac{\mathbb{E}\{L(\boldsymbol{U}(0))\}}{M\epsilon}.$$

The above inequality holds for all positive integers M. Taking a limsup as $M \to \infty$ yields the result. □

The theory of Lyapunov drift has a long history in the field of discrete stochastic processes and Markov chains (see, for example, [10, 106]). The first application of the theory to the design of dynamic control algorithms for radio networks appeared in [143], where a general algorithm was developed to stabilize a multi-hop packet radio network with configurable link activation sets. The concepts of maximum weight matching and differential backlog scheduling, developed in [143], play important roles in the dynamic control strategies we present in later sections. The Lyapunov drift approach has been successfully used to optimize allocation of computer resources [21, 22], stabilize packet switch systems [72, 80, 87, 103, 148, 150] satellite and wireless systems [4, 67, 110, 144, 145, 149, 158], and ad-hoc mobile networks [111]. Recently, a simple extension of Lyapunov drift theory is developed in [108, 115, 116] to provide both stability and performance optimization (addressed in more detail in Sections 5 and 6).

Lyapunov drift theory for queueing networks is traditionally presented in terms of *Foster's criterion for stability* (see, for example, [10]). Roughly, Foster's criterion applies to queueing processes that evolve according to ergodic Markov chains with countably infinite state spaces, and ensures that the Markov chain has a well defined steady state provided that some mild assumptions hold, and provided that the

Lyapunov drift is negative whenever the queue state is outside of a bounded region of the state space. The form of the Lyapunov drift Lemma (Lemma 4.1) does not require Foster's criterion, and is adapted from similar statements in [67, 87, 110]. This approach to Lyapunov stability yields a simpler stability proof as well as an upper bound on average queue occupancy. If the conditions of Lemma 4.1 are supplemented with the additional Markovian assumptions of Foster's criterion, then the resulting limsup of the backlog bound in Lemma 4.1 can be replaced with a regular limit, as Foster's criterion guarantees the limit exists.

A statement similar to Lemma 4.1 can also be made concerning *T-slot Lyapunov drift*, which is useful in cases when network stochastics require more than one timeslot to ensure a negative drift:

Lemma 4.2. (T-slot Lyapunov drift) If there is a positive integer T such that $\mathbb{E}\{\boldsymbol{U}(\tau)\} < \infty$ for $\tau \in \{0, \ldots, T-1\}$, and if there are positive values B, ϵ such that for all timeslots t_0 we have:

$$\mathbb{E}\{L(\boldsymbol{U}(t_0+T)) - L(\boldsymbol{U}(t_0))|\boldsymbol{U}(t_0)\} \leq B - \epsilon \sum_{i=1}^{L} U_i(t_0),$$

then the network is strongly stable, and the average congestion satisfies:

$$\limsup_{t\to\infty} \frac{1}{t} \sum_{\tau=0}^{t-1} \sum_{i=1}^{L} \mathbb{E}\{U_i(\tau)\} \leq \frac{B}{\epsilon}.$$

The proof is similar to the proof of Lemma 4.1, and is omitted for brevity (see [111, 115] for details).

As a preliminary demonstration of the power of Lyapunov drift theory, we use the T-slot theorem to prove the sufficient condition for queue stability given in Lemma 3.6 of Section 3.1.1. The next simple lemma will be useful.

Lemma 4.3. If V, U, μ, A are nonnegative real numbers and

$$V \leq \max[U - \mu, 0] + A,$$

then

$$V^2 \leq U^2 + \mu^2 + A^2 - 2U(\mu - A).$$

Consider a single queue with backlog $U(t)$ and with arrival and server processes $A(t)$ and $\mu(t)$.

Lemma 4.4. If $A(t)$ is admissible with rate less than or equal to λ, and if $\mu(t)$ is admissible with rate greater than or equal to $\overline{\mu}$, and if $\lambda < \overline{\mu}$, then the queue is strongly stable, and queue backlog satisfies:

$$\limsup_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t-1} \mathbb{E}\{U(\tau)\} \leq \frac{T(\mu_{max}^2 + A_{max}^2)}{\overline{\mu} - \lambda}, \tag{4.9}$$

where μ_{max} is a bound such that $\mu(t) \leq \mu_{max}$ for all t, and where T is the smallest integer such that at every timeslot t and (regardless of past history of the system) the following condition holds:

$$\mathbb{E}\left\{\frac{1}{T}\sum_{\tau=t}^{t+T-1} \mu(\tau) - \frac{1}{T}\sum_{\tau=t}^{t+T-1} A(\tau)\,|\boldsymbol{H}(t)\right\} \geq (\overline{\mu} - \lambda)/2. \tag{4.10}$$

Further, if the arrivals $A(t)$ are i.i.d. every slot with mean $\mathbb{E}\{A(t)\} = \lambda$, and if $\mu(t)$ service rates are i.i.d. every slot with mean $\mathbb{E}\{\mu(t)\} = \overline{\mu}$, then:

$$\limsup_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t-1} \mathbb{E}\{U(\tau)\} \leq \frac{(\mu_{max}^2 + A_{max}^2)}{2(\overline{\mu} - \lambda)}. \tag{4.11}$$

The parameter T used in the above theorem can be viewed as the time required for the system to reach "near steady state." Note that the general bound (4.9) differs by a factor of $2T$ from the bound (4.11) for the i.i.d. case. This is due to the fact that non-i.i.d. systems may have system states that yield low transmission rates or large arrival bursts for many timeslots in a row. While the i.i.d. case can be viewed as a special case when $T = 1$, the extra factor of 2 arises because i.i.d. systems effectively "reach steady state" on each and every timeslot, so that the left hand side of (4.10) is exactly equal to $(\overline{\mu} - \lambda)$ for all t. Below we present a proof of Lemma 4.4 for the non-i.i.d. case. The argument

below is the only T-slot analysis that we present in this text. In later sections we restrict proofs to cases that involve simpler i.i.d. assumptions, with the understanding that these proofs can be modified using T-slot analysis to yield similar results for non-i.i.d. systems.

Proof. (Lemma 4.4) The unfinished work in the queue T slots into the future can be bounded in terms of the current unfinished work as follows:

$$U(t+T) \leq \max\left[U(t) - \sum_{\tau=t}^{t+T-1} \mu(\tau), 0\right] + \sum_{\tau=t}^{t+T-1} A(\tau).$$

The above expression is an inequality instead of an equality because new arrivals may depart before the T slot interval is finished. From Lemma 4.3 we have:

$$U^2(t+T) \leq U^2(t) + T^2\mu_{max}^2 + \left(\sum_{\tau=t}^{t+T-1} A(\tau)\right)^2 - 2TU(t)\left[\frac{1}{T}\sum_{\tau=t}^{t+T-1}\mu(\tau) - \frac{1}{T}\sum_{\tau=t}^{t+T-1} A(\tau)\right].$$

Taking conditional expectations with respect to $U(t)$, noting that

$$\mathbb{E}\{A(\tau_1)A(\tau_2)\,|U(t)\} \leq \sqrt{\mathbb{E}\{A(\tau_1)^2\,|U(t)\}\mathbb{E}\{A(\tau_2)^2\,|U(t)\}} \leq A_{max}^2,$$

and using the definition of T yields:

$$\mathbb{E}\left\{U^2(t+T) - U^2(t)\,|U(t)\right\} \leq T^2\mu_{max}^2 + T^2A_{max}^2 - 2TU(t)(\overline{\mu} - \lambda)/2.$$

Applying Lemma 4.2 to the above inequality (using $L(U) = U^2$) yields (4.9), proving the result. □

4.5 Lyapunov drift for networks

In this section we show that the Dynamic Backpressure Algorithm stabilizes the network, using Lyapunov drift techniques described in Section 4.4. Let $\boldsymbol{U}(t)$ represent the matrix of queue backlogs, and define the following Lyapunov function:

$$L(\boldsymbol{U}) = \sum_{ic}\left(U_i^{(c)}\right)^2.$$

The above sum is taken over all (i,c) entries, where we formally define $U_i^{(c)}(t) = 0$ for all t whenever $(i,c) \notin \mathcal{D}$. To calculate the drift $\Delta(\boldsymbol{U}(t))$ we apply Lemma 4.3 to the queueing equation (4.3) and obtain,

$$\left(U_i^{(c)}(t+1)\right)^2 \leq \left(U_i^{(c)}(t)\right)^2 + \left(\sum_b \mu_{ib}^{(c)}(t)\right)^2 + \left(A_i^{(c)}(t) + \sum_a \mu_{ai}^{(c)}(t)\right)^2 - 2U_i^{(c)}(t)\left[\sum_b \mu_{ib}^{(c)}(t) - A_i^{(c)}(t) - \sum_a \mu_{ai}^{(c)}(t)\right].$$

Summing over all valid entries (i,c) and using the fact that the sum of squares of non-negative variables is less than or equal to the square of the sum, it is not difficult to show that the above inequality implies:

$$L(\boldsymbol{U}(t+1)) - L(\boldsymbol{U}(t)) \leq 2BN - 2\sum_{ic} U_i^{(c)}(t)\left[\sum_b \mu_{ib}^{(c)}(t) - A_i^{(c)}(t) - \sum_a \mu_{ai}^{(c)}(t)\right].$$

where

$$B \triangleq \frac{1}{2N}\sum_{i\in\mathcal{N}}\left[(\mu_{max,i}^{out})^2 + (A_i^{max} + \mu_{max,i}^{in})^2\right]. \tag{4.12}$$

Taking conditional expectations yields the following bound for Lyapunov drift:

$$\Delta(\boldsymbol{U}(t)) \leq 2BN + 2\sum_{ic} U_i^{(c)}(t)\mathbb{E}\left\{A_i^{(c)}(t)\middle|\boldsymbol{U}(t)\right\} - 2\mathbb{E}\left\{\sum_{ic} U_i^{(c)}(t)\left[\sum_b \mu_{ib}^{(c)}(t) - \sum_a \mu_{ai}^{(c)}(t)\right]\middle|\boldsymbol{U}(t)\right\}. \tag{4.13}$$

Using the Basic Property discussed in at the end of Section 4.3 we conclude that the Backpressure Algorithm is designed to minimize at each time slot, over all admissible policies, the bound in the right hand side of (4.13). This is the key property on which the proof of stability of the algorithm, as described in the next theorem, is based.

Theorem 4.5. (Backpressure Algorithm Performance) If there exists a value $\epsilon > 0$ such that $\boldsymbol{\lambda} + \boldsymbol{\epsilon} \in \Lambda$ (where $\boldsymbol{\epsilon}$ is a matrix with all entries

$(i,c) \in \mathcal{D}$ equal to ϵ, and all other entries equal to zero), then the above dynamic backpressure algorithm stabilizes the network. Furthermore, in the special case when the arrival processes $\boldsymbol{A}(t)$ are i.i.d. over timeslots and the topology state process is i.i.d. over timeslots (so that $\mathbb{E}\{\boldsymbol{A}(t)\} = \boldsymbol{\lambda}$ and $Pr[S(t) = s] = \pi_s$ for all timeslots t), the average congestion satisfies:

$$\limsup_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t-1}\sum_{ic}\mathbb{E}\left\{U_i^{(c)}(\tau)\right\} \leq \frac{NB}{\epsilon_{max}},$$

where ϵ_{max} is the largest value of ϵ such that $\boldsymbol{\lambda} + \boldsymbol{\epsilon} \in \Lambda$, and B is defined in (4.12).

The theorem is proven in [111, 115] for the case of general admissible inputs and Markov modulated topology state processes, where a congestion bound is also derived. As in the case of single hop networks, the congestion bound for the non-i.i.d. case is roughly a factor of T larger than the i.i.d. congestion bound given above, where T is the duration required for the system to reach "near steady state." Below we prove the theorem for the i.i.d. case.

Proof. (Theorem 4.5) For simplicity, we assume that $Cl(\Gamma) = \Gamma$. Since arrivals $\boldsymbol{A}(t)$ are i.i.d. over timeslots, we have $\mathbb{E}\left\{A_i^{(c)}(t)\,|\boldsymbol{U}(t)\right\} = \lambda_i^{(c)}$ for all (i,c). Hence we can rewrite (4.13) as:

$$\Delta(\boldsymbol{U}(t)) \leq 2BN + 2\sum_{ic} U_i^{(c)}(t)\lambda_i^{(c)} - 2\sum_{ic} U_i^{(c)}(t)\left[\mathbb{E}\left\{\sum_b \mu_{ib}^{(c)}(t) - \sum_a \mu_{ai}^{(c)}(t)\middle|\boldsymbol{U}(t)\right\}\right]. \quad (4.14)$$

However, recall the basic inequality (4.7), which states that the Dynamic Backpressure policy minimizes the final term on the right hand side of the above inequality over all possible alternative policies $\tilde{\mu}_{ab}^{(c)}(t)$. However, because $\boldsymbol{\lambda} + \boldsymbol{\epsilon} \in \Lambda$, we know from Corollary 3.9 of Section 3 that there exists a stationary randomized algorithm that makes decisions based only on the current topology state $S(t)$ (and

hence independent of the current queue backlog) so that for all $(i,c) \in \mathcal{D}$ we have:

$$\mathbb{E}\left\{ \sum_b \tilde{\mu}_{ib}^{(c)}(t) - \sum_a \tilde{\mu}_{ai}^{(c)}(t) \middle| \boldsymbol{U}(t) \right\} = \epsilon + \lambda_i^{(c)}.$$

Using the above in (4.14) and considering (4.7), we conclude:

$$\Delta(\boldsymbol{U}(t)) \leq 2BN - 2\epsilon \sum_{ic} U_i^{(c)}(t).$$

This drift inequality is in the exact form for application of the Lyapunov drift lemma (Lemma 4.1 of Section 4.4), proving the result. □

4.6 Time varying arrival rates

The Dynamic Backpressure Algorithm does not require knowledge of input rates or topology state probabilities, and hence it easily adapts to *time varying system statistics* [109, 115]. For example, suppose the arrivals $\boldsymbol{A}(t)$ are i.i.d. with expected rate vector $\mathbb{E}\{\boldsymbol{A}(t)\} = \boldsymbol{\lambda}^{(1)}$ for some duration of time $t_1 \leq t \leq t_2$, but that user demands change after time t_2, so that $\mathbb{E}\{\boldsymbol{A}(t)\} = \boldsymbol{\lambda}^{(2)}$ for $t_2 < t \leq t_3$. After time t_3, the rates might change again, and so on, so that rate changes occur at arbitrary times. It can be shown that the system is strongly stable and has average congestion bounded by a uniform constant, provided that there is a positive value ϵ such that all rate matrices are within a given distance ϵ of the capacity region boundary [109, 115].

4.7 Imperfect scheduling

It is not difficult to show that if a "sub-optimal" control decision $I(t)$ is chosen that satisfies:

$$\sum_{ab} W_{ab}^*(t) C_{ab}(I(t), S(t)) \geq \gamma \max_{I \in \mathcal{I}_{S(t)}} \sum_{ab} W_{ab}^*(t) C_{ab}(I(t), S(t)) - D,$$

for some constants γ, D such that $0 \leq \gamma \leq 1$ and $0 \leq D < \infty$, then the network is also stable, provided that the arrival rates are interior to $\gamma\Lambda$, which is a γ *scaled version of the capacity region.* Thus, if the controller is off from the optimum by no more than an additive constant D

(i.e., $\gamma = 1$), then full stability is still possible (although the resulting congestion increases by an additive constant proportional to D). However, if the controller deviates from optimality by a multiplicative constant, the achievable throughput region may be a subset of the capacity region. This result is presented in [115], and similar results are presented for low complexity switch scheduling in [109, 134]. Related "imperfect scheduling" statements are developed for utility optimization in [91] using a convex optimization theory framework. The result can be shown by a simple modification of the proof of Theorem 4.5. Specifically, the result follows by replacing (4.7) with the following inequality:

$$\gamma \sum_{ic} U_i^{(c)}(t) \left[\sum_b \tilde{\mu}_{ab}^{(c)}(t) - \sum_a \tilde{\mu}_{ab}^{(c)}(t) \right] - D \leq \sum_{ic} U_i^{(c)}(t) \left[\sum_b \mu_{ib}^{(c)}(t) - \sum_a \mu_{ai}^{(c)}(t) \right],$$

and by assuming that $(\boldsymbol{\lambda} + \boldsymbol{\epsilon})/\gamma \in \Lambda$.

This simple result leads to two significant conclusions: First, any effort to allocate resources to increase the value of $\sum_{ab} W_{ab}^*(t) C_{ab}(I(t), S(t))$ will lead to improved network throughput, even if the maximum is not attained. Second, full network capacity can be achieved by using *queue backlog estimates*, provided that the difference between the estimate and the exact value is bounded by a constant [115]. Therefore, assuming that the maximum per-timeslot backlog change in any queue is bounded, full network stability can be achieved by using out of date queue backlog information. Queue updates can be arbitrarily infrequent without affecting stability, although the average congestion may increase in proportion to the duration between updates. Related work in the area of imperfect scheduling for wireless networks is developed in [29, 91, 163].

4.8 Distributed implementation

As mentioned in Section 4.3, the weights $W_{ab}^*(t)$ of the optimization problem (4.5) can be determined by node a provided that this node is aware of the backlog sizes of its neighbors. However, the optimization problem itself is not always easily amenable to a distributed solution,

as it could require full knowledge of the topology state $S(t)$ and full coordination of all network nodes. This is the case in the server allocation problem of Example 2.3, where the constraint $I(t) \in \mathcal{I}$ can only be met by a coordinated decision, and in the power allocation problem of Example 2.5 that requires knowledge of the full power matrix.

These problems can be avoided if the network is designed so that the link rate functions $C_{ab}(I(t), S(t))$ depend only on network conditions and control input decisions *that are local to link* (a, b). In particular, suppose that the topology state $S(t)$, the link control input $I(t)$, and the control space $\mathcal{I}_{S(t)}$ can be decomposed into terms associated with G independent groups of nodes:

$$\begin{aligned} S(t) &= \left(S^1(t),\ S^2(t),\ldots,\ S^G(t)\right), \\ I(t) &= \left(I^1(t),\ I^2(t),\ldots,\ I^G(t)\right), \\ \mathcal{I}_{S(t)} &= \mathcal{I}^1_{S^1(t)} \times \mathcal{I}^2_{S^2(t)} \times \ldots \times \mathcal{I}^G_{S^G(t)}, \end{aligned}$$

where $S^g(t)$ represents the local components of the topology state $S(t)$ measured at nodes within group g (for $g \in \{1, \ldots, G\}$). Likewise, $I^g(t)$ represents the transmission control input decisions corresponding to nodes within group g, and satisfies the constraint $I^g(t) \in \mathcal{I}^g_{S^g(t)}$. Further suppose that the link transmission rate functions for each outgoing link (a, b) of every node a associated with a particular group g_a can be written as pure functions of $S^{g_a}(t)$ and $I^{g_a}(t)$:

$$C_{ab}(I(t), S(t)) = C_{ab}\left(I^{g_a}(t), S^{g_a}(t)\right).$$

In this case, resource allocation decisions associated with a given group g can be made independently of allocation decisions for other groups. An example network where such a decomposition is possible is the ad-hoc mobile network of Example 2.6, where groups are defined according to cell partitions. Similarly, such a decomposition is possible when communication takes place over wireline data links that do not influence other channels, or when all wireless network nodes transmit over orthogonal frequency bands so that there is no inter-channel interference.

However, networks with general interference properties cannot be decomposed in this way. One approach is to make *random* control decisions $I(t)$, and then to have the flow control and routing layers simply

react to the resulting transmission rates $\mu_{ab}(t) = C_{ab}(I(t), S(t))$. This approach is considered in [57, 111, 115] for distributed control of wireless ad-hoc networks. Another approach is to use simplified sub-optimal scheduling to achieve results within a constant factor of optimality. For example, in a network where link activation sets conform to matching constraints, it is well known that a *greedy* contention based scheduling algorithm achieves within a factor of 2 of optimality. Specifically, each node greedily requests to transmit over its maximum weight outgoing link, and conflicting requests are resolved by granting the largest weight contender (breaking ties arbitrarily). The contention scheme must pass through several iterations before reaching a point when no new links can be matched (where each iteration includes at least one new link). Related "factor of k" results apply for greedy scheduling in systems where each link has at most k other interfering links. Greedy scheduling strategies of this type are considered for somewhat different control algorithms in [29, 91, 163, 164]. We will see in Section 5 that the same distributed implementation issues arise when one considers resource optimization problems and fairness issues in addition to stability.

4.9 Algorithm enhancements and shortest path service

Note that the routing constraint sets $\mathcal{L}_c$ can be designed to ensure that data is routed over links that make progress toward the appropriate destination. However, these routing restrictions potentially reduce the network capacity region, and can limit adaptivity when link or node failures necessitate re-routing. These issues can be avoided if all sets $\mathcal{L}_c$ are equal to the set of all network links. While this leads to the largest capacity region Λ, it can also lead to large end-to-end network delay. For example, if a single packet is injected into an empty network, there is no backpressure to suggest an appropriate path. Hence, the packet might take a random walk through the network, or might take a periodic walk that never leads to the destination. In this case, although the network congestion is quite low (only one packet), network delay can be infinite. Similarly, in cases when the network is lightly loaded, the end-to-end delay can be large even though the congestion bound of Theorem 4.5 is satisfied.

To achieve low delay while still avoiding the routing restrictions associated with the sets $\mathcal{L}_c$, we can program a *shortest path bias* into the weights of the dynamic backpressure algorithm. This leads to the following "enhanced" version of the dynamic backpressure algorithm, defined in terms of constants $\theta_i^{(c)} > 0$ and $Q_i^{(c)} \geq 0$.

Enhanced Dynamic Backpressure Routing Algorithm (EDR): For all links (a, b), find the commodity $c_{ab}^*(t)$ such that:

$$c_{ab}^*(t) \triangleq \arg \max_{c \in \{1,\ldots,K\}} \left\{ \theta_a^{(c)}(U_a^{(c)}(t) + Q_a^{(c)}) - \theta_b^{(c)}(U_b^{(c)}(t) + Q_b^{(c)}) \right\},$$

and define:

$$W_{ab}^*(t) \triangleq \max \left[\theta_a^{(c_{ab}^*)} \left(U_a^{(c_{ab}^*)}(t) + Q_a^{(c_{ab}^*)} \right) - \theta_b^{(c_{ab}^*)} \left(U_b^{(c_{ab}^*)}(t) + Q_b^{(c_{ab}^*)} \right), 0 \right].$$

Control decisions are then made as before, using these new weights and commodities $W_{ab}^*(t)$ and $c_{ab}^*(t)$ as a replacement for the originals.

This enhanced strategy is developed in [111, 115] and called the "Enhanced Dynamic Routing and Power Control (EDRPC)" strategy. Using the Lyapunov function $L(\boldsymbol{U}) = \sum_{ic} \theta_i^{(c)} \left(U_i^{(c)} \right)^2$, it is not difficult to show that the enhanced algorithm stabilizes the network whenever the original algorithm does.

In particular, a shortest path bias can be programmed into the algorithm by setting all $\theta_i^{(c)}$ weights to 1, but choosing the weights $Q_i^{(c)}$ to be proportional to the distance (or number of hops) between node i and the destination of commodity c along the shortest path through the network (where $Q_i^{(c)} = 0$ if node i is the destination of commoditiy c). These distances can either be estimated or computed by running a shortest path algorithm. With these bias values, packets are inclined to move in the direction of their shortest paths, providing low delay in lightly loaded conditions while still ensuring stability throughout the entire capacity region.

We note that the combined weight $U_i^{(c)}(t) + Q_i^{(c)}$ associated with commodity c data in node i can be used in the same manner as a routing table, and in situations where the network nodes do not change their

relative locations, the unfinished work quantities can be updated each timeslot by having neighboring nodes transmit their backlog changes over a low bandwidth control channel. As each link transmits only a single commodity every timeslot, the number of such backlog increments required to be transmitted over the control channel by any user is on the order of neighboring nodes.

Below, we consider the special case of a simple "multiple source single sink" static sensor network consisting of 100 sensor nodes randomly placed in a 10×10 square. We use a cell-partitioned model (similar to the one considered in [116]) with 100 equal-sized cells. Each node can communicate with neighboring nodes in the same or adjacent cells. Time is slotted and each node can transmit to at most one of its neighbors in a time slot, though a node can receive from multiple nodes. Each sensor node has independent data of input rate λ to deliver to a centrally located sink node, so that there is only one commodity in the network, and only one destination. Nodes are assumed to have a fixed transmission power P_{MAX}. The data rate achievable over a link is then taken as $BWlog(1 + \alpha P_{MAX})$ where α is the attenuation over that link, being a function of the distance between the nodes and BW is the bandwidth of the link.

We consider two interference scenarios, one in which a node's transmission doesn't cause interference at other nodes and the other where a node's transmission causes interference at nodes in the same or adjacent cells. The performance of a distributed implementation of DRPC and EDRPC is compared with a pure shortest path based routing scheme in this setup. The left plot in Fig. 4.2 shows the performance of these schemes under the "no interference" model with asymmetric bottleneck link capacities. The right plot shows the performance with interference and symmetric link capacities. It can be seen that in both cases, DRPC/EDRPC significantly outperform the shortest path scheme as input rate λ is pushed up. At low data rates, backlog based decisions are likely to lead to false turns, which degrades the performance of DRPC. By incorporating path lengths into the backpressure calculation, EDRPC improves upon the performance of DRPC at low data rates while maintaining the advantages of backlog aware schemes at high data rates.

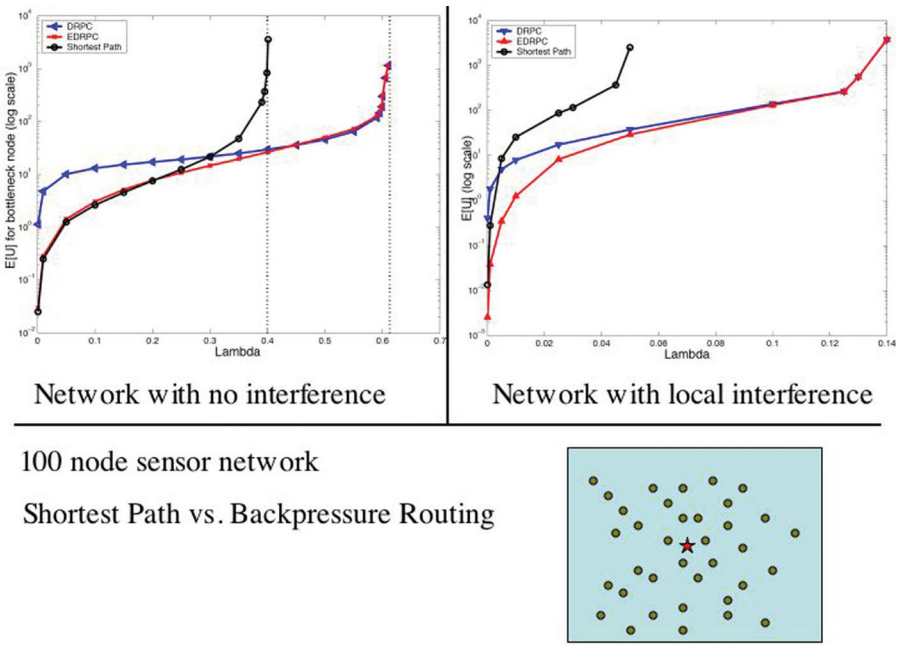

Fig. 4.2 Simulation results comparing backpressure routing, enhanced backpressure routing, and pure shortest path routing, for a 100 node wireless sensor network with a single centrally located destination. We would like to thank Rahul Urgaonkar (University of Southern California) for developing these simulation experiments.

4.10 Multi-commodity flows and convex duality

The Dynamic Backpressure Algorithm stabilizes the network and offers average delay guarantees whenever the input rate matrix is inside the capacity region of the wireless network. Here we consider a related problem of computing an offline multi-commodity flow given a known rate matrix $(\lambda_i^{(c)})$. Classical multi-commodity flow problems for wired networks can be reduced to linear programs, and fast approximation algorithms are developed in [85]. A distributed algorithm was first given in [51], and pricing and game theory approaches are developed in [70, 71].

Here we consider the special case of a network with a time invariant topology state, and formally pose our network stability problem as a static multi-commodity flow problem (following the development given in [115, 111]). We show that a classical subgradient

search method for solving the problem via convex duality theory corresponds almost exactly to a deterministic network simulation of the Dynamic Backpressure Algorithm. Notions of duality are also used in [31, 66, 70, 71, 79, 83, 100, 101, 165] to consider static network optimization, where in many cases the dual variables play the role of prices charged by the network to multiple users competing for shared network resources in order to maximize their own utility. Applications of static duality theory to the area of internet congestion control are developed in [97].

In our context, the dual variables correspond to *queue backlogs*, rather than network prices. This illustrates a relationship between static optimization and the Lyapunov stability theory, suggesting that static algorithms can be modified and applied in dynamic settings while preserving analytical optimality. This observation also motivates our study of *stochastic network optimization* in the next section.

Consider a time invariant network with transmission rate function $\boldsymbol{C}(I(t))$ for $I(t) \in \mathcal{I}$ (note that there is no notion of a time varying topology state here). Suppose that a commodity corresponds to a destination, so that all commodity c data is destined for node c. Given a particular rate matrix $(\lambda_i^{(c)})$, the problem of finding a multi-commodity flow corresponds to the following convex optimization problem.

$$\begin{aligned} &\textit{Maximize}: 1 \\ &\textit{Subject to}: \lambda_i^{(c)} + \sum_a f_{ai}^{(c)} \leq \sum_b f_{ib}^{(c)} \ \forall i, c \text{ with } i \neq c, \\ &\qquad \left(\left\{f_{ab}^{(c)}\right\}, \{\mu_{ab}\}\right) \in \Omega, \end{aligned} \tag{4.15}$$

where Ω is the set of all variables $\left(\left\{f_{ab}^{(c)}\right\}, \{\mu_{ab}\}\right)$ such that:

$$\begin{aligned} &f_{ab}^{(c)} \geq 0 \text{ for all } a, b, c \in \{1, \ldots, N\}, \\ &f_{aa}^{(c)} = f_{ab}^{(a)} = 0 \text{ for all } a, b, c \in \{1, \ldots, N\}, \\ &\left(\sum_c f_{ab}^{(c)}\right) \leq (\mu_{ab}) \text{ for some } (\mu_{ab}) \in Cl\{\Gamma\}. \end{aligned} \tag{4.16}$$

The maximization function "1" is used as an artifice to pose this multi-commodity flow problem in the framework of an optimization problem.

Note that the set Ω is convex and compact (it inherits convexity and compactness from the set $Cl(\Gamma)$ consisting of all link transmission rate matrices (G_{ab}) entrywise less than or equal to some element of Γ, see [115]). Moreover, the objective function "1" and all inequality constraints are linear. The optimization problem is therefore convex [17], and has a dual formulation, where the optimal solution of the dual problem exactly corresponds to an optimal solution of the original 'primal' problem (4.15). To form the dual problem, we introduce non-negative Lagrange multipliers $\{U_i^{(c)}\}$ for each of the inequality constraints in (4.15), and define the dual function:

$$L\left(\{U_i^{(c)}\}\right) \triangleq \max_{\left(\left\{f_{ab}^{(c)}\right\},\{\mu_{ab}\}\right)\in\Omega} \left[1 + \sum_{i\neq c} U_i^{(c)} \left(\sum_b f_{ib}^{(c)} - \sum_a f_{ai}^{(c)} - \lambda_i^{(c)}\right)\right]. \tag{4.17}$$

The dual problem to (4.15) is:

$$\begin{aligned} &\textit{Minimize: } L\left(\left\{U_i^{(c)}\right\}\right), \\ &\textit{Subject to: } U_i^{(c)} \geq 0 \ \text{ for all } i,c \in \{1,\ldots,N\}. \end{aligned}$$

The dual problem is always convex, and the minimizing solution can be obtained using classical subgradient search methods (where the function $-L\left(\left\{U_i^{(c)}\right\}\right)$ is maximized). Consider a fixed stepsize method with stepsize $T = 1$. The basic subgradient search routine starts with an initial set of values $U_i^{(c)}(0)$ for the Lagrange multipliers, and upon each iteration $t = \{1,\ 2,\ldots\}$ these values are updated by computing a subgradient $\underline{\eta}$ for one time unit, and, if necessary, projecting the result back onto the set of non-negative values [17]:

$$U_i^{(c)}(t+1) = \max\left[U_i^{(c)}(t) + \eta_i^{(c)}, 0\right]. \tag{4.18}$$

However, it is shown in [17] that a particular subgradient of $-L\left(\left\{U_i^{(c)}\right\}\right)$ is:

$$\underline{\eta} = \left(-\sum_b f_{ib}^{*(c)} + \sum_a f_{ai}^{*(c)} + \lambda_i^{(c)}\right)\Bigg|_{(i,c)\in\{1,\ldots,N\}^2}, \tag{4.19}$$

where the $\left\{f_{ab}^{*(c)}\right\}$ variables are solutions to the maximization in (4.17) with $U_i^{(c)} = U_i^{(c)}(t)$. Using (4.19) in (4.18) for all $i \neq c$, we find:

$$U_i^{(c)}(t+1) = \max\left[U_i^{(c)}(t) - \sum_b f_{ib}^{*(c)} + \sum_a f_{ai}^{*(c)} + \lambda_i^{(c)}, 0\right]. \quad (4.20)$$

From the above equation, it is apparent that the Lagrange multipliers $\left\{U_i^{(c)}(t)\right\}$ play the role of unfinished work in a multi-node queueing system with input rates $\lambda_i^{(c)}$, where $U_i^{(c)}(t)$ represents the amount of commodity c bits in node i. In this way, the $f_{ab}^{*(c)}$ values can be viewed as the transmission rates allocated to commodity c traffic on link (a,b). Equation (4.20) thus states that the unfinished work at time $t+1$ is equal to the unfinished work at time t plus the net influx of bits into node i. Thus, the operation of projecting the Lagrangian variables onto the positive orthant acts exactly as an implementation of the standard queueing equation.

It is illuminating to calculate the optimal $f_{ab}^{*(c)}$ values by performing the maximization in (4.17). To this end, we need to maximize $\sum_{i\neq c} U_i^{(c)}(t)\left(\sum_b f_{ib}^{(c)} - \sum_a f_{ai}^{(c)}\right)$ subject to the constraints of (4.16). However, as in Section 4.5, we can switch the sum to find:

$$\sum_{i\neq c} U_i^{(c)}\left(\sum_b f_{ib}^{(c)} - \sum_a f_{ai}^{(c)}\right) = \sum_{abc} f_{ab}^{(c)}\left[U_a^{(c)} - U_b^{(c)}\right].$$

Remarkably, from the right hand side above, it is apparent that the optimal values $f_{ab}^{*(c)}$ are identical to the resulting transmission rates $\mu_{ab}^{(c)}(t)$ that would be computed if the Dynamic Backpressure and Resource Allocation Algorithm of Section 4.3 were used to calculate routing and resource allocation decisions in a network problem with unfinished work levels $U_i^{(c)}(t)$. It follows that this backpressure algorithm can be viewed as a dynamic implementation of a subgradient search method for computing the solution to an optimization problem using convex duality. This suggests a deeper relationship between stochastic network control algorithms and subgradient search methods. Further, it suggests an approach to stochastic network optimization. Indeed, note that the optimization problem (4.15), which maximizes

the function "1," can be adjusted to maximize some other performance criteria, which may be of interest in the corresponding dynamic network control problem. One approach is to use the theory of *stochastic approximation* and *stochastic subgradients* (see for example [44, 68]), which has recently been applied in [84] to a downlink scheduling problem with infinitely backlogged sources. In the next section, we develop a method that uses a novel extension of Lyapunov drift theory to allow stability and performance optimization simultaneously [108, 115, 116], yielding explicit tradeoffs in utility optimization and average delay.

5

Networking Outside of the Capacity Region: Utility Optimization and Fairness

Up to this point we have focused attention only on the problem of controlling a network to achieve stability. In this section we begin our treatment of *stochastic network optimization*, where the goal is to stabilize the network while additionally optimizing some performance metric and/or satisfying some additional constraints. Specifically, this section investigates the situation when the exogenous arrival rates are outside of the network capacity region Λ. In this case, the network cannot be stabilized without a transport layer flow control mechanism to limit the amount of data that is admitted. The goal is to design a cross-layer strategy for flow control, routing, and resource allocation that provides stability while achieving *optimal network fairness*. Here, we measure fairness in terms of a general utility function of the long term admission rates of each session.

The solution to this flow control problem involves three new concepts. The first is a simple extension of Lyapunov drift theory that enables stability and performance optimization to be treated simultaneously [108, 115, 116]. The second is the introduction of *auxiliary variables* that are used to hold additional state information useful for optimizing functions of time averages [108, 121, 120]. The third

is the technique of using *virtual cost queues* to transform performance constraints into queueing stability problems [108, 116, 136]. These techniques are instrumental in the design of utility-optimal flow controllers, and shall also be used in Section 6 to address more general problems of network optimization, including minimum energy routing and cost constrained scheduling.

5.1 The flow control model and fairness objective

Consider the general multi-hop network model introduced in Section 2, where the network has N nodes, K commodities, a topology state process $S(t)$, and a link transmission rate function $\boldsymbol{C}(I(t), S(t))$ (where $I(t) \in \mathcal{I}_{S(t)}$). Recall that a general network control algorithm must choose the resource allocation and routing decision variables as follows:

- *Resource (Rate) Allocation:* Observe the current topology state $S(t)$ and choose a transmission control $I(t) \in \mathcal{I}_{S(t)}$ to yield link transmission rates $\boldsymbol{\mu}(t) = \boldsymbol{C}(I(t), S(t))$.
- *Routing/Scheduling:* For each link (a, b) and each commodity c, choose $\mu_{ab}^{(c)}(t)$ to satisfy the following constraints:

$$\sum_c \mu_{ab}^{(c)}(t) \leq \mu_{ab}(t), \tag{5.1}$$

$$\mu_{ab}^{(c)}(t) = 0 \quad \text{if } (a,b) \notin \mathcal{L}_c, \tag{5.2}$$

(where $\mathcal{L}_c$ is the set of all network links that are acceptable for commodity c data to traverse).

Newly arriving data does not immediately enter the network layer. Rather, it first enters a *transport layer storage reservoir*. Specifically, new data of commodity c that arrives to source node n is first placed in a transport layer reservoir (n, c). A control valve determines the amount of data $R_n^{(c)}(t)$ released from this reservoir on each timeslot. This $R_n^{(c)}(t)$ process acts as the exogenous arrival process to the network layer queue $U_n^{(c)}(t)$. As discussed in Section 2.2, the following inequality holds for

the network queues.

$$U_n^{(c)}(t+1) \leq \max\left[U_n^{(c)}(t) - \sum_b \mu_{nb}^{(c)}(t), 0\right] + \sum_a \mu_{an}^{(c)}(t) + R_n^{(c)}(t). \tag{5.3}$$

Let $L_n^{(c)}(t)$ represent the current backlog in the transport layer reservoir (n,c) at time t. The flow control decision variables $R_n^{(c)}(t)$ are chosen every timeslot according to the following restriction:

- *Flow Control (Type 1)*: Choose $R_n^{(c)}(t)$ such that:

$$R_n^{(c)}(t) \leq L_n^{(c)}(t) + A_n^{(c)}(t) \quad \text{for all } (n,c) \text{ and all } t,$$
$$\sum_c R_n^{(c)}(t) \leq R_n^{max} \quad \text{for all } n \text{ and all } t,$$

  where the constants R_n^{max} are chosen to be positive and suitably large, to be made precise in the development of our CLC1 flow control algorithm (introduced in the next section). The first constraint above ensures that admitted data is less than or equal to the actual data available, and the second is important for limiting the burstiness of the admitted arrivals. We label the above flow control constraints as "Type 1" to distinguish them from the following alternative constraint specifications:

- *Flow Control (Type 2)*: Choose $R_n^{(c)}(t)$ such that:

$$R_n^{(c)}(t) \leq L_n^{(c)}(t) + A_n^{(c)}(t) \quad \text{for all } (n,c) \text{ and all } t,$$
$$R_n^{(c)}(t) \leq \hat{R}_n^{(c)} \quad \text{for all } n \text{ and all } t,$$

  where $\hat{R}_n^{(c)}$ are suitably large positive constants, to be made precise in our CLC2 algorithm in Section 5.4.2. The Type 2 constraints are simpler but are also less restrictive and generally lead to a larger bound on average network congestion.

5.1.1 The fairness objective

Let $\boldsymbol{\lambda} = \left(\lambda_n^{(c)}\right)$ denote the arrival rate matrix of the exogenous arrival streams $(A_n^{(c)}(t))$. This rate matrix is arbitrary, and in particular the

rates can either be *inside or outside of the capacity region* Λ. Flow control decisions about what data to admit are crucial in the case when the input rates exceed network capacity, and it is important to establish a quantitative measure of *network fairness*. To this end, we define a set of *utility functions* $g_n^{(c)}(r)$, representing the "satisfaction" received by sending commodity c data from node n to the destination node of this commodity at a time average rate of r bits/slot. The utility functions are assumed to be non-decreasing and concave. Such utility functions are a conventional means of measuring network fairness. For example, concave utilities are used to evaluate fairness for wireline networks in [70, 71, 97, 107], for static wireless networks in [16, 31, 66, 83, 91], and for stochastic wireless networks in [46, 84, 108, 115, 136]. Furthermore, different choices of the $g_n^{(c)}(r)$ functions lead to different fairness properties [71, 140].

For our problem, the goal is to support a fraction of the traffic demand matrix $\boldsymbol{\lambda}$ to achieve a long term throughput matrix $\boldsymbol{r} = \left(r_n^{(c)}\right)$ that maximizes the sum of user utilities. The *optimal sum utility* is thus defined by the following optimization problem:

$$\text{Maximize: } \sum_{n,c} g_n^{(c)}\left(r_n^{(c)}\right) \tag{5.4}$$

$$\text{Subject to: } \boldsymbol{r} \in \Lambda, \tag{5.5}$$

$$0 \leq r_n^{(c)} \leq \lambda_n^{(c)} \quad \text{for all } (n,c). \tag{5.6}$$

Inequality (5.5) is the *stability constraint* and ensures that the long term admitted rates are stabilizable by the network. Inequality (5.6) is the *demand constraint* that ensures the admission rate of session (n,c) is no more than the incoming traffic rate of this session.

Because the functions $g_n^{(c)}(r)$ are non-decreasing, it is clear that if $\boldsymbol{\lambda} \in \Lambda$, then the above optimization is solved by the matrix $\boldsymbol{r}^* = \boldsymbol{\lambda}$ (so that $r_n^{*(c)} = \lambda_n^{(c)}$ for all (n,c)). If $\boldsymbol{\lambda} \notin \Lambda$, then the solution $\boldsymbol{r}^*$ will lie somewhere on the capacity region boundary. The above optimization could in principle be solved if the arrival rates $(\lambda_n^{(c)})$ and the capacity region Λ were known in advance, and all users could coordinate by sending data according to the optimal solution. However, the capacity region depends on the topology state probabilities, which are unknown

to the network controllers and to the individual users. Furthermore, the individual users do not know the data rates or utility functions of the other users. In this section, we develop a practical dynamic control strategy that yields a resulting matrix of throughputs $\overline{\boldsymbol{r}}$ that is arbitrarily close to the optimal solution of (5.4)–(5.6). The distance to the optimal solution is shown to decrease like $1/V$, where V is a control parameter affecting a tradeoff in average delay for data that is served by the network.

5.1.2 Capacity region geometry

Recall that $\mathcal{D}$ is the set of all (n, c) pairs that represent valid network layer queues $U_n^{(c)}(t)$ (so that it is possible for commodity c data to be present at node n). The integer D represents the total number of these queues, and defines the *effective dimension* of the network. We assume throughout that $U_n^{(c)}(t) \triangleq 0$, $R_n^{(c)}(t) \triangleq 0$ for all t whenever $(n, c) \notin \mathcal{D}$, and that $g_n^{(c)}(r) \triangleq 0$, $\lambda_n^{(c)} \triangleq 0$ whenever $(n, c) \notin \mathcal{D}$.

The capacity region Λ can be shown to be compact and convex with D effective dimensions [115]. It shall be useful to define the parameter μ_{sym} to be the largest time average admission rate that is simultaneously supportable by all sessions $(n, c) \in \mathcal{D}$, so that $(\mu_{sym} 1_n^{(c)}) \in \Lambda$ (where $1_n^{(c)}$ is an indicator function that is equal to 1 whenever $(n, c) \in \mathcal{D}$, and zero else). Geometrically, the value μ_{sym} represents the edge size of the largest D-dimensional hypercube that can be fit into the capacity region Λ, and is a value that unexpectedly arises in our analysis. We assume throughout that $\mu_{sym} > 0$.

In the next section, we present a solution to the fairness problem (5.4)–(5.5) in the special case when all active sessions are *infinitely backlogged* (so that the demand constraint (5.6) is removed). A modified algorithm that uses *auxiliary variables* and *flow state queues* is then presented in Section 5.4 to solve the general problem (5.4)–(5.6).

5.2 Dynamic control for infinite demand

Here we develop a practical control algorithm that stabilizes the network and ensures that utility is arbitrarily close to optimal, with a

corresponding tradeoff in network delay. We define an *active session* (n,c) to be a source-commodity pair such that $(n,c) \in \mathcal{D}$ and $g_n^{(c)}(r)$ is not identically zero. To highlight the fundamental issues of routing, resource allocation, and flow control, in this section we assume that all active sessions (n,c) have *infinite backlog* in their corresponding reservoirs, so that flow variables $R_n^{(c)}(t)$ can be chosen without first establishing that this much data is available for admission. Flow control is imperative in this infinite backlog scenario, and the resulting problem is simpler as it does not involve the demand constraint (5.6).

The following control strategy, developed in [108, 115], is decoupled into separate algorithms for resource allocation, routing, and flow control.

Cross-Layer Control Algorithm 1 (CLC1) [108, 115]:

- *Flow Control* — (algorithm FLOW1) Every timeslot, the flow controller at each node n observes the current level of queue backlogs $U_n^{(c)}(t)$ for each commodity $c \in \{1,\ldots,K\}$. It then sets $R_n^{(c)}(t) = r_n^{(c)}$, where the $r_n^{(c)}$ values are solutions to the following optimization:

$$\text{Maximize : } \sum_{c=1}^{K}\left[V g_n^{(c)}\left(r_n^{(c)}\right) - r_n^{(c)}U_n^{(c)}(t)\right], \tag{5.7}$$

$$\text{Subject to: } \left(r_n^{(c)}\right) \geq 0, \ \sum_{c=1}^{K} r_n^{(c)} \leq R_n^{max}, \tag{5.8}$$

  where $V > 0$ is a chosen constant that effects the performance of the algorithm.
- *Routing and Scheduling* — Each node n observes the backlog in all neighboring nodes j to which it is connected by a valid outgoing link (n,j). Let $W_{nj}^{(c)}(t) = U_n^{(c)}(t) - U_j^{(c)}(t)$ represent the *differential backlog* of commodity c data, and define $W_{nj}^*(t) \triangleq \max_{[c|l\in\mathcal{L}_c]}\left\{W_{nj}^{(c)}(t), 0\right\}$. Let $c_{nj}^*(t)$ represent the maximizing commodity. Data of commodity $c_{nj}^*(t)$ is selected for (potential) routing over link (n,j) whenever $W_{nj}^*(t) > 0$.

- *Resource Allocation* — The current topology state $S(t)$ is observed, and a transmission decision $I(t) \in \mathcal{I}_{S(t)}$ is selected by maximizing $\sum_{n,j} W^*_{nj}(t)\mu_{nj}(t)$, where $\mu_{nj}(t) = C_{nj}(I(t), S(t))$. The resulting transmission rate of $\mu_{nj}(t)$ is offered to commodity $c^*_{nj}(t)$ data on link (n, j). If any node does not have enough bits of a particular commodity to send over all outgoing links requesting that commodity, *null bits* are delivered.

The flow control policy (5.7)–(5.8) uses a parameter V that determines the extent to which utility optimization is emphasized. Indeed, if V is large relative to the current backlog in the source queues, then the admitted rates $R^{(c)}_n(t)$ will be large, increasing the time average utility while consequently increasing congestion. This effect is mitigated as backlog grows at the source queues and flow control decisions become more conservative. Note that the routing and resource allocation strategies of CLC1 are identical to the Dynamic Backpressure strategy developed for network stability in Section 4.3. Issues of distributed implementation can be dealt with using the methods of Section 4.8.

The flow control algorithm of CLC1 is decentralized, where the control valves for each node n require knowledge only of the queue backlogs in node n. Note that CLC1 uses "Type 1" flow control constraints. The resulting problem (5.7)–(5.8) involves maximizing a sum of concave functions subject to a simplex constraint, and can easily be solved using standard derivative matching techniques [19]. It is useful to note that this flow control strategy can be replaced by a strategy that uses "Type 2" flow control constraints, and thus maximizes (5.7) over the less restrictive constraint set $0 \leq r^{(c)}_n \leq R^{max}_n$. This constraint set is simpler because it allows each flow control decision variable $R^{(c)}_n(t)$ to be determined by finding the maximum of a concave function of one variable over a given interval, a problem in which closed form solutions are often readily available. However, this simplicity comes at the cost of a potential increase in average network congestion and delay, due to the fact that the maximum admission burst into node n during a single slot would be KR^{max}_n, rather than R^{max}_n (see delay analysis in

Section 5.3). The simplified constraint set is identical to (5.8) in the special case when there is only a single active session (n, c) at any given node n, considered in examples in Sections 5.2.2–5.2.4.

We also note that the nature of the CLC1 flow control algorithm assumes that data can be admitted as "fractional packets." This is because the resulting $R_n^{(c)}(t)$ values might not be integers or integer multiples of a given packet length. This problem arises whenever the utility functions $g_n^{(c)}(r)$ are non-linear. The problem can be mitigated when R_n^{max} is large in comparison to the packet granularity, or can be overcome by appending an additional stage to the flow control reservoir that only sends actual packets into the network when the accumulated "admitted but undelivered" data exceeds the packet length. The problem is avoided entirely in the modified algorithm CLC2 that uses *auxiliary variables* to handle non-linear effects (Section 5.4.2).

5.2.1 Algorithm performance

To analyze the performance of the above CLC1 algorithm, we define the maximum transmission rates out of and into a given node n as follows:

$$\mu_{max,n}^{out} \triangleq \sup_{[S,I\in\mathcal{I}_S]} \sum_{b=1}^{N} C_{nb}(I,S), \quad \mu_{max,n}^{in} \triangleq \sup_{[S,I\in\mathcal{I}_S]} \sum_{a=1}^{N} C_{an}(I,S).$$

Assume that the flow control constants R_n^{max} are positive and satisfy $R_n^{max} \geq \mu_{max,n}^{out}$ for all n. Assume utilities $g_n^{(c)}(r)$ are non-negative, non-decreasing, continuous, and concave, and define $G_{max} \triangleq \max_{\left[\left(r_n^{(c)}\right)\middle|\sum_c r_n^{(c)} \leq R_n^{max}:\forall n\right]} \sum_{n,c} g_n^{(c)}\left(r_n^{(c)}\right)$. Define the constant B as follows:

$$B \triangleq \frac{1}{2N}\sum_{n=1}^{N}\left[(R_n^{max}+\mu_{max,n}^{in})^2 + (\mu_{max,n}^{out})^2\right]. \tag{5.9}$$

Theorem 5.1. If channel states are i.i.d. over timeslots and all active reservoirs have infinite backlog, then for any flow parameter $V > 0$ the CLC1 algorithm stabilizes the network and yields the following time

average congestion and utility bounds:[1]

$$\overline{\sum_{n,c} U_n^{(c)}} \leq \frac{NB + VG_{max}}{\mu_{sym}}, \tag{5.10}$$

$$\liminf_{t\to\infty} \sum_{n,c} g_n^{(c)}(\overline{r}_{nc}(t)) \geq \sum_{n,c} g_n^{(c)}\left(r_n^{*(c)}\right) - \frac{BN}{V}, \tag{5.11}$$

where $\boldsymbol{r}^* = (r_n^{*(c)})$ is the optimal solution of (5.4) subject to constraint (5.5), and where:

$$\overline{\sum_{n,c} U_n^{(c)}} \triangleq \limsup_{t\to\infty} \frac{1}{t} \sum_{\tau=0}^{t-1} \left[\sum_{n,c} \mathbb{E}\left\{ U_n^{(c)}(\tau) \right\} \right],$$

$$\overline{r}_{nc}(t) \triangleq \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}\left\{ r_n^{(c)}(\tau) \right\}. \tag{5.12}$$

The above result holds for all $V > 0$. Thus, the value of V can be chosen so that BN/V is arbitrarily small, resulting in achieved utility that is arbitrarily close to optimal. This performance comes at the cost of a (potential) linear increase in network congestion with the parameter V. By Little's theorem, average queue backlog is proportional to average bit delay, and hence performance can be pushed towards optimality with a corresponding tradeoff in end-to-end network delay.

A result similar to the above theorem holds if the "Type 1" flow control constraint $\sum_c R_n^{(c)}(t) \leq R_n^{max}$ in (5.8) is replaced with the simpler and less restrictive "Type 2" constraint $R_n^{(c)}(t) \leq R_n^{max}$ (for all (n,c)). However, the constant B in the performance bounds (5.10) (5.11) would then need to be replaced with a constant $\tilde{B}$ that is larger than B roughly by a factor equal to the largest number of distinct commodities that are sourced at any single node.

The proof of Theorem 5.1 uses a Lyapunov technique that allows stability and performance optimization to be treated simultaneously,

[1] Using T-slot Lyapunov analysis, the CLC1 algorithm can be shown to yield similar performance for non-i.i.d. systems (with an increased constant B). The same is true for the other performance optimal algorithms presented in this work.

and is provided in Section 5.3. Below we present a simple corollary that is useful in characterizing the performance of CLC1 under *suboptimal* resource allocation strategies:

Corollary 5.2. If the resource allocation policy of CLC1 is replaced with any (potentially randomized) policy that yields a transmission rate function $\boldsymbol{\mu}(t) = \boldsymbol{C}(I(t), S(t))$ that satisfies the following for all slots t:

$$\sum_{ab} W^*_{ab}(t)\mathbb{E}\left\{\mu_{ab}(t)\,|\boldsymbol{U}(t)\right\} \geq \gamma\left(\max_{I\in\mathcal{I}_{S(t)}} \sum_{ab} W^*_{ab}(t)C_{ab}(I, S(t))\right) - C,$$

for some fixed constants γ and C such that $0 < \gamma \leq 1$ and $C \geq 0$, then

$$\overline{\sum_{n,c} U_n^{(c)}} \leq \frac{C + NB + VG_{max}}{\mu_{sym}\gamma}, \tag{5.13}$$

$$\liminf_{t\to\infty} \sum_{n,c} g_n^{(c)}(\overline{r}_{nc}(t)) \geq \sum_{n,c} g_n^{(c)}(\tilde{r}^*_{nc}) - \frac{C + NB}{V}, \tag{5.14}$$

where $(\tilde{r}^*_{nc})$ is the optimal solution to the following optimization:

$$\begin{aligned} \text{Maximize: } & \sum_{n,c} g_n^{(c)}(r_n^{(c)}) \\ \text{Subject to: } & (r_n^{(c)}) \in \gamma\Lambda, \\ & 0 \leq r_n^{(c)} \leq \lambda_{nc}. \end{aligned} \tag{5.15}$$

That is, allocating resources to come within a factor γ of the optimal solution of the CLC1 resource allocation yields a utility that is close to the optimal utility with respect to a γ scaled version of the capacity region. The above corollary is related to the approximate Lyapunov scheduling results presented in Section 4.7. A closely related "imperfect scheduling" result is developed from a convex programming perspective in [91]. Below we present the implications of these results.

5.2.2 Maximum throughput and the threshold rule

Suppose utilities are linear, so that $g_n^{(c)}(r) = \alpha_{nc} r$ for some non-negative weights α_{nc}. The resulting objective is to maximize the weighted sum of throughput, and the resulting FLOW1 algorithm of CLC1 has a simple threshold form, where some commodities receive as much of the R_n^{max} delivery rate as possible, while others receive none. In the special case where the user at node n desires communication with a single destination node c_n (so that $g_n^{(c)}(r) = 0$ for all $c \neq c_n$), the flow control algorithm (5.7) reduces to maximizing $V\alpha_{nc_n} r - U_n^{(c_n)}(t) r$ subject to $0 \leq r \leq R_n^{max}$, and the solution is the following threshold rule:

$$R_{nc_n}(t) = \begin{cases} R_n^{max} & \text{if } U_n^{(c_n)}(t) \leq V\alpha_{nc_n} \\ 0 & \text{otherwise} \end{cases} .$$

The qualitative structure of this flow control rule is intuitive: When backlog in the source queue is large, we should refrain from sending new data. The simple threshold form is similar to the threshold scheduling rule developed in [157] for server scheduling in a downlink with N ON/OFF channels and burstiness constraints on the channel states and packet arrivals. Specifically, assuming $\alpha_n \leq \alpha_{n+1}$ (for $1 \leq n \leq N-1$), the policy developed in [157] assigns at time t to channel queue n an index

$$I_n\left(U\left(t\right)\right) = \min\left\{U\left(t\right),\ nT\right\},$$

where T is a given threshold parameter, and transmits a packet from the ON queue with the highest index. It is shown that for T large enough this policy maximizes the weighted sum of channel throughputs. In the special case of this downlink scenario (a commodity is identified with a channel), the resource allocation layer of the CLC1 policy would always serve the longest ON queue, and the flow control layer would only allow new packets of type i into queue i if the current backlog in this queue is below V (and so backlog in queue i would never grow beyond $V + R_i^{max}$). In comparison, we see that the V and T parameters play similar roles.

5.2.3 Proportional fairness and the $1/U$ rule

Consider now utility functions of the form $g_n^{(c)}(r) = \log\left(1 + \beta r_n^{(c)}\right)$ (for some constant $\beta > 0$). It is shown in [71] that maximizing a sum of such utilities over any convex set Λ leads to *proportional fairness*.[2] In the special case when there is only one destination c_n for each user n, the flow control algorithm reduces to maximizing $V\log(1+\beta r) - U_n^{(c_n)} r$ subject to $0 \leq r \leq R_n^{max}$, which leads to the following '$1/U$' flow control function:

$$R_{nc_n}(t) = \min\left[\max\left[\frac{V}{U_n^{(c_n)}(t)} - \frac{1}{\beta}, 0\right], R_n^{max}\right].$$

Here we see that the flow control valve restricts flow according to a continuous function of the backlog level at the source queue, being less conservative in its admission decisions when backlog is low and more conservative when backlog is high.

5.2.4 Mechanism design and network pricing

The flow control policy (5.7) has a simple interpretation in terms of network pricing. Specifically, consider a scenario where the $g_n^{(c)}(r)$ functions are measured in units of dollars, representing the amount the user at source node n is willing to pay for rate r service to destination c. The *social optimum operating point* $\left(r_n^{*(c)}\right)$ is defined as the point that maximizes the sum of utilities $\sum_{n,c} g_n^{(c)}\left(r_n^{(c)}\right)$ subject to $\left(r_n^{(c)}\right) \in \Lambda$. For simplicity, we again assume that there is a single user at node n that desires to send a single commodity c_n. Every timeslot, each user n determines the amount of data $r_n^{(c_n)}(t)$ it desires to send based on a per-unit

[2] Strictly speaking, the proportionally fair allocation seeks to maximize $\sum_{n,c}\log(r_n^{(c)})$, leading to $\sum_{n,c}\frac{\overline{r}_{nc}^{opt} - r_n^{(c)}}{\overline{r}_{nc}^{opt}} \geq 0$ for any other operating point $(r_n^{(c)}) \in \Lambda$. We use non-negative utilities $\log(1+\beta r)$ and thereby obtain a proportionally fair allocation with respect to the quantity $\overline{r}_{nc}^{opt} + 1/\beta$, leading to $\sum_{n,c}\frac{\overline{r}_{nc}^{opt} - r_n^{(c)}}{\overline{r}_{nc}^{opt} + 1/\beta} \geq 0$. This can be used to approximate proportionally fair scheduling for large β. Alternatively, it can be used with $\beta = 1$, yielding a utility function $\log(1+r)$ which is different from proportionally fair utility but still has many desirable fairness properties.

price $PRICE_{nc_n}(t)$ charged by the network. The transaction between user and network takes place in a distributed fashion at each node n. We assume all users are 'greedy' and send data every timeslot by maximizing total utility minus total cost, subject to an R_n^{max} constraint imposed by the network. That is, each user n selects $R_n^{(c_n)}(t) = r_n^{(c_n)}$, where the $r_n^{(c_n)}$ values solve:

$$\begin{aligned} &\text{Maximize : } g_n^{(c_n)}\left(r_n^{(c_n)}\right) - PRICE_{nc_n}(t) r_n^{(c_n)}, \\ &\text{Subject to: } 0 \leq r_n^{(c_n)} \leq R_n^{max}. \end{aligned} \tag{5.16}$$

Consider now the following dynamic pricing strategy used at each network node n:

$$PRICE_{nc}(t) = \frac{U_n^{(c)}(t)}{V} \text{ dollars/bit.}$$

We note that this pricing strategy is independent of the particular $g_n^{(c)}(r)$ functions, and so the network does not require knowledge of the user utilities. Using this pricing strategy in (5.16), it follows that users naturally send according to processes $R_n^{(c)}(t)$ that exactly correspond to the FLOW1 algorithm (5.7), and hence the performance bounds (5.10) and (5.11) are satisfied.

5.3 Performance analysis

To prove Theorem 5.1, we first introduce a simple result from [108, 115, 116] that extends the Lyapunov stability results presented in Lemma 4.1 of Section 4.4 to include performance optimization. In particular, the technique allows queueing stability and performance optimization to be treated using a single drift analysis.

5.3.1 Lyapunov optimization

To begin, consider any discrete time queueing system with vector backlog process $\boldsymbol{U}(t) = (U_1(t), \ldots, U_N(t))$ that evolves according to some probability law, and let $\boldsymbol{R}(t) = (R_1(t), \ldots, R_K(t))$ represent any associated vector control process that influences system dynamics (for some

integers N and K). Let $L(\boldsymbol{U})$ be any non-negative function of $\boldsymbol{U}$. Recall that the one-step Lyapunov drift $\Delta(\boldsymbol{U}(t))$ is defined as follows:

$$\Delta(\boldsymbol{U}(t)) \triangleq \mathbb{E}\{L(\boldsymbol{U}(t+1)) - L(\boldsymbol{U}(t)) | \boldsymbol{U}(t)\},$$

where the conditional expectation is taken with respect to the random one-step queueing dynamics given the current backlog $\boldsymbol{U}(t)$.

Lemma 5.3. (Lyapunov Drift) Let $\mathbb{E}\{L(\boldsymbol{U}(0))\} < \infty$. If there exist scalar random processes $x(t)$ and $y(t)$ such that for every timeslot t and for all possible values of $\boldsymbol{U}(t)$, the Lyapunov drift satisfies:

$$\Delta(\boldsymbol{U}(t)) \leq \mathbb{E}\{y(t) | \boldsymbol{U}(t)\} - \mathbb{E}\{x(t) | \boldsymbol{U}(t)\}, \tag{5.17}$$

then:

$$\limsup_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t-1} \mathbb{E}\{x(\tau)\} \leq \limsup_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t-1} \mathbb{E}\{y(\tau)\},$$
$$\liminf_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t-1} \mathbb{E}\{x(\tau)\} \leq \liminf_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t-1} \mathbb{E}\{y(\tau)\}.$$

Proof. Taking expectations of (5.17) with respect to the distribution of $\boldsymbol{U}(t)$ and using the law of iterated expectations yields:

$$\mathbb{E}\{L(\boldsymbol{U}(t+1))\} - \mathbb{E}\{L(\boldsymbol{U}(t))\} \leq \mathbb{E}\{y(t)\} - \mathbb{E}\{x(t)\}.$$

The above inequality holds for all t. Summing over $t \in \{0, \ldots, M-1\}$ yields:

$$\mathbb{E}\{L(\boldsymbol{U}(M))\} - \mathbb{E}\{L(\boldsymbol{U}(0))\} \leq \sum_{\tau=0}^{M-1} \mathbb{E}\{y(\tau)\} - \sum_{\tau=0}^{M-1} \mathbb{E}\{x(\tau)\}.$$

Shifting terms, dividing by M, and using non-negativity of $L(\boldsymbol{U})$ yields:

$$\frac{1}{M}\sum_{\tau=0}^{M-1} \mathbb{E}\{x(\tau)\} \leq \frac{1}{M}\sum_{\tau=0}^{M-1} \mathbb{E}\{y(\tau)\} + \frac{\mathbb{E}\{L(\boldsymbol{U}(0))\}}{M}.$$

The result follows by taking limits as $M \to \infty$. □

Suppose now that the goal is to stabilize the $\boldsymbol{U}(t)$ process while minimizing a concave function $g(\cdot)$ of the time average of the $\boldsymbol{R}(t)$ process. Specifically, we define the following vector of time average expectations over t slots:

$$\overline{\boldsymbol{r}}(t) \triangleq \frac{1}{t}\sum_{\tau=0}^{t-1}\mathbb{E}\left\{\boldsymbol{R}(t)\right\}. \tag{5.18}$$

Let $g(\boldsymbol{r})$ be any scalar valued, concave utility function of a K dimensional variable $\boldsymbol{r}$, and let g^* represent a desired "target" utility value.

Theorem 5.4. (Lyapunov Optimization) If there are positive constants V, ϵ, B such that for all timeslots t and all unfinished work matrices $\boldsymbol{U}(t)$, the Lyapunov drift satisfies:

$$\Delta(\boldsymbol{U}(t)) - V\mathbb{E}\left\{g(\boldsymbol{R}(t))|\boldsymbol{U}(t)\right\} \leq B - \epsilon\sum_{i=1}^{N}U_i(t) - Vg^*, \tag{5.19}$$

then time average utility and congestion satisfies:

$$\limsup_{t\to\infty}\frac{1}{t}\sum_{\tau=0}^{t-1}\sum_{i=1}^{N}\mathbb{E}\left\{U_i(\tau)\right\} \leq \frac{B + V(\overline{g} - g^*)}{\epsilon}, \tag{5.20}$$

$$\liminf_{t\to\infty} g(\overline{\boldsymbol{r}}(t)) \geq g^* - \frac{B}{V}, \tag{5.21}$$

where $\overline{\boldsymbol{r}}(t)$ is defined in (5.18), and $\overline{g}$ is defined:

$$\overline{g} \triangleq \limsup_{t\to\infty}\frac{1}{t}\sum_{\tau=0}^{t-1}\mathbb{E}\left\{g(\boldsymbol{R}(\tau))\right\}.$$

This theorem is most useful when the quantity $(\overline{g} - g^*)$ can be bounded by a constant. For example, if $0 \leq g(\boldsymbol{R}(t)) \leq G_{max}$ for all t (for some constant G_{max}), then $(\overline{g} - g^*) \leq G_{max}$. It follows that if the parameter V can be chosen as desired, then the lower bound on achieved utility can be pushed arbitrarily close to the target utility g^*, with a corresponding increase in queue congestion that is linear in V.

Proof. (Theorem 5.4) Define:

$$x(t) \triangleq \epsilon \sum_i U_i(t) + Vg^*,$$
$$y(t) \triangleq B + Vg(\boldsymbol{R}(t)).$$

The drift condition (5.19) thus implies:

$$\Delta(\boldsymbol{U}(t)) \leq \mathbb{E}\{y(t)|\boldsymbol{U}(t)\} - \mathbb{E}\{x(t)|\boldsymbol{U}(t)\}. \tag{5.22}$$

Using the lim sup result of Lemma 5.3 yields:

$$\limsup_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t-1} \mathbb{E}\left\{\epsilon \sum_i U_i(\tau) + Vg^*\right\}$$
$$\leq \limsup_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t-1} \mathbb{E}\{B + Vg(\boldsymbol{R}(\tau))\}.$$

The right hand side of the above inequality is equal to $B + V\overline{g}$. Rearranging terms yields (5.20). Likewise, using (5.22) with the lim inf result of Lemma 5.3 yields:

$$\liminf_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t-1} \mathbb{E}\{x(t)\} \leq \liminf_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t-1} \mathbb{E}\{B + Vg(\boldsymbol{R}(t))\}$$
$$= B + V\liminf_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t-1} \mathbb{E}\{g(\boldsymbol{R}(t))\}.$$

Noting that $x(t) \geq Vg^*$ yields:

$$Vg^* \leq B + V\liminf_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t-1} \mathbb{E}\{g(\boldsymbol{R}(t))\}.$$

Dividing by V and rearranging terms yields:

$$g^* - B/V \leq \liminf_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t-1} \mathbb{E}\{g(\boldsymbol{R}(t))\} \tag{5.23}$$
$$\leq \liminf_{t\to\infty} g\left(\frac{1}{t}\sum_{\tau=0}^{t-1} \mathbb{E}\{\boldsymbol{R}(\tau)\}\right) \tag{5.24}$$
$$= \liminf_{t\to\infty} g(\overline{\boldsymbol{r}}(t)),$$

where (5.24) follows from Jensen's inequality together with concavity of $g(\cdot)$. It follows that $\liminf_{t\to\infty} g(\overline{\boldsymbol{r}}(t)) \geq g^* - B/V$, proving the theorem. □

It is useful to note that a similar result can be shown for *minimizing* a convex cost function $h(\boldsymbol{r})$ by defining $g(\boldsymbol{r}) = -h(\boldsymbol{r})$ and reversing inequalities where appropriate. Further, the proof uses concavity of $g(\boldsymbol{r})$ only in the inequality (5.24). Thus, without the concavity assumption, the congestion bound (5.20) still holds, while the utility bound (5.24) is replaced by (5.23).

5.3.2 Computing the drift

To apply the Lyapunov Optimization Theorem (Theorem 5.4) to the design and analysis of our CLC1 control policy, we define the utility function $g(\boldsymbol{r}) \triangleq \sum_{n,c} g_n^{(c)}\left(r_n^{(c)}\right)$. This utility function is used to evaluate the utility associated with the flow control decision variables $\boldsymbol{R}(t) = \left(R_n^{(c)}(t)\right)$. Further, we define the Lyapunov function $L(\boldsymbol{U}) \triangleq \frac{1}{2}\sum_{n,c}\left(U_n^{(c)}\right)^2$ (the factor $1/2$ is used for notational convenience later). The Lyapunov Optimization Theorem suggests that a good control strategy is to greedily minimize the following drift metric every timeslot:

$$\Delta(\boldsymbol{U}(t)) - V\sum_{n,c}\mathbb{E}\left\{g_n^{(c)}\left(R_n^{(c)}(t)\right)|\boldsymbol{U}(t)\right\}. \tag{5.25}$$

This is indeed the principle behind the CLC1 control algorithm.

To begin, recall that the Lyapunov drift $\Delta(\boldsymbol{U}(t))$ for any control policy can be computed using methods in Section 4.5, and we have:

$$\Delta(\boldsymbol{U}(t)) \leq NB - \sum_{n,c} U_n^{(c)}(t)\mathbb{E}\left\{\sum_b \mu_{nb}^{(c)}(t) - \sum_a \mu_{an}^{(c)}(t) - R_n^{(c)}(t)\,|\boldsymbol{U}(t)\right\}. \tag{5.26}$$

where B is defined in (5.9). The expectations above are taken with respect to the distribution of the random topology state $S(t)$ at time t,

and with respect to the (potentially randomized) choice of control decision variables.

Now define the *flow function* $\Psi(\boldsymbol{U}(t))$ and the *network function* $\Phi(\boldsymbol{U}(t))$ as follows:

$$\Psi(\boldsymbol{U}(t)) \triangleq \sum_{n,c} \mathbb{E}\left\{ V g_n^{(c)}(R_n^{(c)}(t)) - U_n^{(c)}(t) R_n^{(c)}(t) \,|\boldsymbol{U}(t) \right\}, \tag{5.27}$$

$$\Phi(\boldsymbol{U}(t)) \triangleq \sum_{n,c} U_n^{(c)}(t) \mathbb{E}\left\{ \sum_b \mu_{nb}^{(c)}(t) - \sum_a \mu_{an}^{(c)}(t) \,|\boldsymbol{U}(t) \right\}. \tag{5.28}$$

Subtracting the utility component $V \sum_{n,c} \mathbb{E}\left\{ g_n^{(c)}\left(R_n^{(c)}(t)\right) |\boldsymbol{U}(t) \right\}$ from both sides of (5.26) yields:

$$\begin{aligned} &\Delta(\boldsymbol{U}(t)) - V \sum_{n,c} \mathbb{E}\left\{ g_n^{(c)}\left(R_n^{(c)}(t)\right) |\boldsymbol{U}\,(t) \right\} \\ &\qquad \leq NB - \Phi(\boldsymbol{U}(t)) - \Psi(\boldsymbol{U}(t)). \end{aligned} \tag{5.29}$$

The design strategy for CLC1 is now apparent: Given a particular $\boldsymbol{U}(t)$ matrix at time t, *the CLC1 policy is designed to greedily minimize the right hand side of (5.29) over all possible routing, resource allocation, and flow control options.* Indeed, it is clear that the flow control strategy (5.7) maximizes $\Psi(\boldsymbol{U}(t))$ over all feasible choices of the $R_n^{(c)}(t)$ values (compare (5.7) and (5.27)). That the routing and resource allocation policy of CLC1 maximizes $\Phi(\boldsymbol{U}(t))$ has already been seen in Section 4.3, where the differential backlog policy was presented. Our analysis proceeds by finding a *stationary* control policy for choosing the decision variables that does not depend on the queue backlog, and plugging the resulting $\Phi(\boldsymbol{U}(t))$ and $\Psi(\boldsymbol{U}(t))$ functions associated with such a policy into the right hand side of (5.29).

5.3.3 A near-optimal operating point

To design a suitable stationary control policy, it is important to first consider a *near-optimal* solution to the optimization problem (5.4)–(5.6). Specifically, for any $\epsilon > 0$, we define the set Λ_ϵ as follows:

$$\Lambda_\epsilon \triangleq \left\{ \left(r_n^{(c)}\right) \middle| (r_n^{(c)} + \epsilon 1_n^{(c)}) \in \Lambda, r_n^{(c)} \geq 0 \text{ for all } (n,c) \in \mathcal{D} \right\}, \tag{5.30}$$

where $1_n^{(c)}$ is equal to 1 whenever $(n,c) \in \mathcal{D}$, and zero else. Thus, the set Λ_ϵ can be viewed as the resulting set of rate matrices within the network capacity region when an "ϵ-layer" of the boundary is stripped away from the D effective dimensions. Note that this set is compact and non-empty whenever $\epsilon \leq \mu_{sym}$ (where μ_{sym} is defined in Section 5.1.2). The *near-optimal operating point* $\left(r_n^{*(c)}(\epsilon)\right)$ is defined as a solution to the following optimization problem:[3]

$$\begin{aligned} \text{Maximize : } & \sum_{n,c} g_n^{(c)}\left(r_n^{(c)}\right) \\ \text{Subject to: } & \left(r_n^{(c)}\right) \in \Lambda_\epsilon \\ & 0 \leq r_n^{(c)} \leq \lambda_{nc} \quad \text{for all } (n,c) \end{aligned} \tag{5.31}$$

This optimization differs from the optimization in (5.4)–(5.6) in that the set Λ is replaced by the set Λ_ϵ.

Lemma 5.5. (Continuity of Near-Optimal Solutions) If utility functions $g_n^{(c)}(r)$ are non-negative and concave, and if there is a positive scalar μ_{sym} such that $(\mu_{sym} 1_n^{(c)}) \in \Lambda$, then:

$$\sum_{n,c} g_n^{(c)}\left(r_n^{*(c)}(\epsilon)\right) \to \sum_{n,c} g_n^{(c)}\left(r_n^{*(c)}\right) \quad \text{as } \epsilon \to 0, \tag{5.32}$$

where $\left(r_n^{*(c)}\right)$ is the optimal solution of (5.4)–(5.6).

Proof. The proof uses convexity of the capacity region Λ, and is given in Section 5.5.2 of [115]. □

5.3.4 Derivation of Theorem 5.1

The proof of Theorem 5.1 relies on the following two lemmas.

[3] Note that the final constraint $\left(r_n^{(c)}\right) \leq \left(\lambda_n^{(c)}\right)$ is satisfied automatically in the case of infinite traffic demand. We include the constraint here as this optimization is also important in the treatment of general traffic matrices $\left(\lambda_n^{(c)}\right)$ in Section 5.4.2.

Lemma 5.6. If the storage reservoirs for all active input streams are infinitely backlogged, then for any ϵ such that $0 < \epsilon \leq \mu_{sym}$, the flow control algorithm of CLC1 yields:

$$\Psi^{CLC1}(\boldsymbol{U}(t)) \geq V \sum_{n,c} g_n^{(c)} \left(r_n^{*(c)}(\epsilon) \right) - \sum_{n,c} U_n^{(c)}(t) r_n^{*(c)}(\epsilon).$$

where $\left(r_n^{*(c)}(\epsilon) \right)$ is the optimal solution of problem (5.31).

Lemma 5.7. If the topology state $\boldsymbol{S}(t)$ is i.i.d. over timeslots, then for any ϵ such that $0 < \epsilon \leq \mu_{sym}$, allocating resources and routing according to CLC1 yields:

$$\Phi^{CLC1}(\boldsymbol{U}(t)) \geq \sum_{n,c} U_n^{(c)}(t) \left(r_n^{*(c)}(\epsilon) + \epsilon \right), \tag{5.33}$$

where $\left(r_n^{*(c)}(\epsilon) \right)$ is the optimal solution of problem (5.31).

Lemma 5.6 follows because the flow control algorithm of CLC1 maximizes $\Psi(\boldsymbol{U}(t))$, defined in (5.27), over all valid resource allocation options, including the particular choice $R_n^{(c)}(t) = r_n^{*(c)}(\epsilon)$ for all (n, c). This is a *valid choice* because: 1) all reservoirs are assumed to be infinitely backlogged, so there are always $r_n^{*(c)}(\epsilon)$ units of data available, and 2) $\sum_c r_n^{*(c)}(\epsilon) \leq R_n^{max}$ (because $(r_n^{*(c)}(\epsilon)) \in \Lambda$ and hence $\sum_c r_n^{*(c)}(\epsilon) \leq \mu_{max,n}^{out}$ is required). Lemma 5.7 follows because the resource allocation and routing algorithm of CLC1 maximizes $\Phi(\boldsymbol{U}(t))$, defined in (5.28), over all other options, including the stationary randomized strategy of Corollary 3.9 that would yield for all $(n, c) \in \mathcal{D}$:

$$\mathbb{E}\left\{ \sum_b \mu_{nb}^{*(c)} - \sum_a \mu_{an}^{*(c)} | \boldsymbol{U}(t) \right\} = r_n^{*(c)}(\epsilon) + \epsilon 1_n^{(c)}. \tag{5.34}$$

Plugging the bounds of Lemmas 5.6 and 5.7 directly into the drift expression (5.29) yields the following for algorithm CLC1:

$$\Delta(\boldsymbol{U}(t)) - V\sum_{n,c}\mathbb{E}\left\{g_n^{(c)}\left(R_n^{(c)}(t)\right)|\boldsymbol{U}(t)\right\} \leq NB$$
$$- \sum_{n,c} U_n^{(c)}(t)\left(r_n^{*(c)}(\epsilon)+\epsilon\right)$$
$$- V\sum_{n,c} g_n^{(c)}\left(r_n^{*(c)}(\epsilon)\right) + \sum_{n,c} U_n^{(c)}(t) r_n^{*(c)}(\epsilon).$$

Canceling common terms yields:

$$\Delta(\boldsymbol{U}(t)) - V\sum_{n,c}\mathbb{E}\left\{g_n^{(c)}\left(R_n^{(c)}(t)\right)|\boldsymbol{U}(t)\right\} \leq NB$$
$$- \epsilon\sum_{n,c} U_n^{(c)}(t) - V\sum_{n,c} g_n^{(c)}\left(r_n^{*(c)}(\epsilon)\right).$$

The above drift expression is in the exact form specified by Theorem 5.4. Thus, network congestion satisfies:

$$\overline{\sum_{n,c} U_n^{(c)}} \leq (NB + VG_{max})/\epsilon, \tag{5.35}$$

and time average performance satisfies:

$$\sum_{n,c} g_n^{(c)}(\overline{r}_{nc}) \geq \sum_{n,c} g_n^{(c)}\left(r_n^{*(c)}(\epsilon)\right) - NB/V. \tag{5.36}$$

The performance bounds in (5.35) and (5.36) hold for any value ϵ such that $0 < \epsilon \leq \mu_{sym}$. However, the particular choice of ϵ only affects the bound calculation and does not affect the CLC1 control policy or change any sample path of system dynamics. We can thus optimize the bounds separately over all possible ϵ values. The bound in (5.35) is clearly minimized as $\epsilon \to \mu_{sym}$, yielding: $\overline{\sum_{n,c} U_n^{(c)}} \leq (NB + VG_{max})/\mu_{sym}$. Conversely, the bound in (5.36) is maximized by taking a limit as $\epsilon \to 0$, yielding by (5.32): $\sum_{n,c} g_n^{(c)}(\overline{r}_{nc}) \geq \sum_{n,c} g_n^{(c)}\left(r_n^{*(c)}\right) - NB/V$. This proves Theorem 5.1. □

5.3.5 Notes

- The utility optimization problem for network flow control was first formalized by Kelly et. al. in [70, 71] for wireline

networks with fluid-like dynamics. There, network flows are described according to primal and dual convex programs, and Lagrange multipliers are interpreted as *shadow prices* that facilitate distributed control mechanisms. The *proportionally fair* objective is considered in [70, 71], and related work in [101, 107] treats different systems and objectives. Game theory aspects of network fairness are considered in [65, 101], auction algorithms are considered in [138], and adversarial queueing theory approaches are considered in [5]. The relationship between utility optimization, convex duality theory, and classical TCP protocols for wireline networks is explored in [97].

- Convex programming approaches to static wireless networks are considered in [16, 31, 36, 66, 79, 83, 91, 100, 165]. The work in [31] investigates systems with transmission rates that depend logarithmically on $SINR$. Under the assumption that $\log(1 + SINR) \approx \log(SINR)$, a set of decoupled algorithms for flow control, routing, and resource allocation are constructed and shown to drive resources to a fixed optimal operating point. The work in [36] considers the case $\log(1 + SINR) \approx SINR$. In this case, it is shown that fixed operating points are sub-optimal, and that optimal strategies involve a time-varying link transmission schedule. A similar problem is investigated in [76], where NP-Completeness results are developed for transmission scheduling.
- Utility optimization and proportional fairness are also treated in [46, 81, 153, 159] for stochastic wireless downlinks with infinite backlog. Fairness for systems with different downlink channels for each arriving packet is treated in [24].
- Fair allocation according to the maxmin rule has been considered in [133, 146] where optimal scheduling policies were proposed for single hop and multihop traffic respectively.
- The CLC1 policy was developed for general stochastic networks in [108, 115]. We note that alternative approaches to stochastic network optimization have recently been considered in [46, 136] using fluid limit models, and in [84]

using stochastic gradient theory (see, for example, [68]). The Lyapunov optimization technique of this section is related to the theory of static and stochastic gradients, as the drift metric (5.25) is analogous to an iterative gradient projection for a static convex program (see Sections 4.7–5.7 of [115]).

5.4 Flow control for arbitrary input rates

Here we consider the general flow control problem (5.4)–(5.6), without the infinite backlog assumption. The transport layer storage buffers are assumed to have either infinite or finite capacity (possibly zero). In the special case of a size-zero storage reservoir, all data that is not immediately admitted to the network layer is necessarily dropped. Let $L_n^{(c)}(t)$ represent the current backlog of commodity c data in the transport layer storage reservoir at node n (where $L_n^{(c)}(t) = 0$ for all t in the case of a size-zero storage reservoir). Flow control decisions are now subject to the additional scheduling constraint $R_n^{(c)}(t) \leq L_n^{(c)}(t) + A_n^{(c)}(t)$. This constraint is particularly challenging, as it varies with time and so the stationary algorithm $R_n^{(c)}(t) = r_n^{*(c)}(\epsilon)$ cannot be used as a valid comparison every slot.

Assume that the $A_n^{(c)}(t)$ arrivals are i.i.d. over timeslots with arrival rates $\lambda_n^{(c)} = \mathbb{E}\left\{A_n^{(c)}(t)\right\}$. It can be shown that for any matrix $(\lambda_n^{(c)})$ (either inside or outside the capacity region), modifying the CLC1 algorithm to maximize (5.7) subject to the additional reservoir backlog constraint yields the same performance guarantees (5.10) and (5.11) *when utility functions are linear* [115]. For nonlinear utilities, such a strategy can be shown to maximize $\sum_{n,c}\mathbb{E}\left\{g_n^{(c)}\left(R_n^{(c)}(t)\right)\right\}$ over all strategies that make immediate admission/rejection decisions upon arrival, but may not necessarily maximize $\sum_{n,c} g_n^{(c)}\left(\mathbb{E}\left\{R_n^{(c)}(t)\right\}\right)$, which is the utility metric of interest.

In this section, we solve the problem by introducing two new techniques. The first is the use of *auxiliary variables* that hold additional network flow state information helpful for solving nonlinear problems [108]. The second is the use of *virtual cost queues* that transform

stochastic constraints involving time averages into simple queueing stability problems [108, 116, 121, 136] .

5.4.1 Problem transformation via auxiliary variables

We begin with the following transformation of the problem (5.4)–(5.6), which introduces new variables $\gamma_n^{(c)}$ for each input stream $R_n^{(c)}(t)$:

$$\text{Maximize:} \quad \sum_{n,c} g_n^{(c)}\left(\gamma_n^{(c)}\right) \tag{5.37}$$

$$\text{Subject to:} \quad r_n^{(c)} \geq \gamma_n^{(c)}, \tag{5.38}$$

$$\left(r_n^{(c)}\right) \in \Lambda, \tag{5.39}$$

$$0 \leq r_n^{(c)} \leq \lambda_n^{(c)}. \tag{5.40}$$

It is not difficult to show that the optimal solution of the above problem is exactly the same as the optimal solution of the original problem (5.4)–(5.6).[4] Note that if the $r_n^{(c)}$ variables are associated with time averages of the flow control inputs $R_n^{(c)}(t)$, then any control policy that stabilizes the system will naturally lead to time averages that satisfy (5.39) and (5.40). Furthermore, the utility optimization is now expressed entirely in terms of the new variables $\gamma_n^{(c)}$, while the additional inequality (5.38) expresses a *linear constraint* between the time average flow control decisions and the $\gamma_n^{(c)}$ variables.

To solve the above problem, for each active input stream $A_n^{(c)}(t)$ we define additional *flow state queues* $Y_n^{(c)}(t)$ that ensure the additional constraint (5.38) is satisfied. Specifically, we define $Y_n^{(c)}(0) = 0$ for all (n, c), and update the flow state queues every timeslot as follows:

$$Y_n^{(c)}(t+1) = \max\left[Y_n^{(c)}(t) - R_n^{(c)}(t), 0\right] + \gamma_n^{(c)}(t), \tag{5.41}$$

where $\gamma_n^{(c)}(t)$ represents a process of non-negative auxiliary variables that the flow controller computes on every timeslot. The $Y_n^{(c)}(t)$ process can also be viewed as a *virtual queue* with "arrivals" $\gamma_n^{(c)}(t)$ and "server rate" $R_n^{(c)}(t)$ (see Fig. 5.1). Let $\boldsymbol{Y}(t) = (Y_n^{(c)}(t))$ represent the matrix

[4] Recall that the $g_n^{(c)}(r)$ utility functions are non-decreasing.

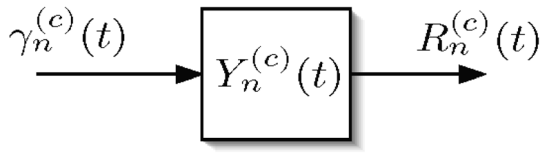

Fig. 5.1 A queueing illustration of the dynamic equation (5.41) for the $Y_n^{(c)}(t)$ queues.

of these flow state queues, and let $\boldsymbol{\Theta}(t) = [\boldsymbol{Y}(t); \boldsymbol{U}(t)]$ represent the combined matrix of flow state queues and actual queues.

To motivate the control algorithm, suppose that the $R_n^{(c)}(t)$ and $\gamma_n^{(c)}(t)$ processes have well defined time averages $\overline{r}_n^{(c)}$ and $\overline{\gamma}_n^{(c)}$, respectively.

Observation: If a control algorithm stabilizes all actual queues $\boldsymbol{U}(t)$ and flow state queues $\boldsymbol{Y}(t)$ of the system, then the resulting time averages $\overline{r}_i^{(c)}$ and $\overline{\gamma}_i^{(c)}$ must satisfy all inequality conditions (5.38)–(5.40).

That (5.39) and (5.40) hold if the $\boldsymbol{U}(t)$ queues are stable is clear because the admitted rates must always be less than or equal to the actual arrival rates, and because the admitted rate matrix must be within the network capacity region Λ for stability of the network queues. The key component of the above observation is that the constraints (5.38) hold if all virtual queues $\boldsymbol{Y}(t)$ are stable. This follows from Lemma 3.3 and the basic queueing inequality,

$$\sum_{\tau=0}^{t-1} R_n^{(c)}(t) + Y_n^{(c)}(t) \geq \sum_{\tau=0}^{t-1} \gamma_n^{(c)}(t).$$

The above observation was introduced in [116] for the purpose of designing optimal scheduling algorithms for wireless networks with average power constraints, where *virtual power queues* were used to transform stochastic inequality constraints into queueing stability problems. Similar techniques have recently been used to treat other stochastic cost constraints in [119, 120, 136].

The above observation suggests the approach of designing a network control algorithm to stabilize all queues $\boldsymbol{U}(t)$ and $\boldsymbol{Y}(t)$ while maximizing the utility function $g(\cdot)$. To this end, define the Lyapunov function

$L(\mathbf{\Theta})$ as follows:

$$L(\mathbf{\Theta}) = \frac{1}{2}\sum_{n,c}\left(U_n^{(c)}\right)^2 + \frac{\eta}{2}\sum_{n,c}\left(Y_n^{(c)}\right)^2,$$

where η is a parameter that satisfies $0 < \eta \leq 1$ and determines the relative weight of virtual queues in the control problem. It turns out that choosing η small decreases the time average backlog in the actual queues while increasing the time average backlog in the virtual flow state queues. The actual queues determine the actual congestion and delay in the network, while the virtual queues play a role in determining the transient time or "learning time" required for the system to approach optimal performance.

The conditional Lyapunov drift is given by:

$$\Delta(\mathbf{\Theta}(t)) \triangleq \mathbb{E}\left\{L(\mathbf{\Theta}(t+1)) - L(\mathbf{\Theta}(t))\,|\mathbf{\Theta}(t)\right\}.$$

Motivated by the drift condition of the Lyapunov Optimization Theorem (Theorem 5.4), we design a control policy to minimize the following metric:

$$\text{Minimize: } \Delta(\mathbf{\Theta}(t)) - V\mathbb{E}\left\{\sum_{n,c} g_n^{(c)}\left(\gamma_n^{(c)}(t)\right)|\mathbf{\Theta}(t)\right\}.$$

5.4.2 The cross-layer control algorithm

To simplify exposition, suppose that the exogenous arrivals to a given node n are deterministically bounded by a constant R_n^{max} every timeslot, so that:

$$\sum_c A_n^{(c)}(t) \leq R_n^{max} \quad \text{for all } t.$$

Further, we shall use the "Type 2" flow control constraints, so that:

$$R_n^{(c)}(t) \leq \min\left[L_n^{(c)}(t) + A_n^{(c)}(t), \hat{R}_n^{(c)}\right],$$

where $\hat{R}_n^{(c)}$ are suitably large constants that satisfy:

$$A_n^{(c)}(t) \leq \hat{R}_n^{(c)} \quad \text{for all } t$$

(note that choosing $\hat{R}_n^{(c)} = R_n^{max}$ for all n ensures the above inequality is satisfied). This "Type 2" flow control constraint simplifies the algorithm at the expense of increasing the average congestion bound.

Using the queueing equations (5.41) and (5.3), the following drift bound can be computed:

$$\begin{aligned}\Delta(\mathbf{\Theta}(t)) - V\mathbb{E}\left\{\sum_{n,c} g_n^{(c)}\left(\gamma_n^{(c)}(t)\right)|\mathbf{\Theta}(t)\right\} \leq C_2 \\ -\sum_{n,c} U_n^{(c)}(t)\mathbb{E}\left\{\left[\sum_b \mu_{nb}^{(c)}(t) - \sum_a \mu_{an}^{(c)}(t)\right]|\mathbf{\Theta}(t)\right\} \\ +\sum_{n,c} U_n^{(c)}(t)\mathbb{E}\left\{R_n^{(c)}(t)\,|\mathbf{\Theta}(t)\right\} \\ -\eta\sum_{n,c} Y_n^{(c)}(t)\mathbb{E}\left\{\left[R_n^{(c)}(t) - \gamma_n^{(c)}(t)\right]|\mathbf{\Theta}(t)\right\} \\ -V\mathbb{E}\left\{\sum_{n,c} g_n^{(c)}\left(\gamma_n^{(c)}(t)\right)|\mathbf{\Theta}(t)\right\}, \qquad (5.42)\end{aligned}$$

where C_2 is a constant that depends on $\mu_{max,n}^{in}, \mu_{max,n}^{out}$, and $\hat{R}_n^{(c)}$. Choosing the control decision variables to minimize the right hand side of the above inequality leads to the following Cross Layer Control algorithm:[5]

Cross Layer Control Algorithm (CLC2b): Every timeslot, the topology state $S(t)$ and the queue values $\boldsymbol{U}(t), \boldsymbol{Y}(t)$ are observed, and controllers perform the following actions:

(1) *Flow Control:* Every timeslot and for each active input stream (n,c), observe $U_n^{(c)}(t)$ and $Y_n^{(c)}(t)$ and choose:

$$R_n^{(c)}(t) = \begin{cases} \min\left[L_n^{(c)}(t) + A_n^{(c)}(t), \hat{R}_n^{(c)}\right] & \text{if } \eta Y_n^{(c)}(t) > U_n^{(c)}(t) \\ 0 & \text{otherwise} \end{cases}.$$

[5] This control algorithm is a modified version of the original CLC2 algorithm from [108], and hence we label it "CLC2b."

Furthermore, for each (n,c), choose $\gamma_n^{(c)}(t) = \gamma_n^{(c)}$, where $\gamma_n^{(c)}$ solves:

$$\begin{aligned} \text{Maximize:} \quad & Vg_n^{(c)}(\gamma) - \eta Y_n^{(c)}(t)\gamma, \\ \text{Subject to:} \quad & 0 \leq \gamma \leq \hat{R}_n^{(c)}. \end{aligned}$$

The flow state queues $Y_n^{(c)}(t)$ are then updated according to (5.41).

(2) *Routing and Resource Allocation:* Same as the dynamic backpressure algorithm of CLC1.

The following theorem assumes arrival matrices $\boldsymbol{A}(t)$ and topology states $S(t)$ are i.i.d. over timeslots, and assumes that G_{max} is a parameter such that $\sum_{n,c} g_n^{(c)}\left(R_n^{(c)}(t)\right) \leq G_{max}$ for all t.

Theorem 5.8. For arbitrary rate matrices $\left(\lambda_n^{(c)}\right)$ (possibly outside of the capacity region), for any $V > 0$, and for any reservoir buffer size (possibly zero), the CLC2b algorithm stabilizes the network and yields the following congestion and utility bounds:

$$\limsup_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t-1}\sum_{n,c} U_n^{(c)}(\tau) \leq \frac{C_2 + VG_{max}}{\mu_{sym}}, \tag{5.43}$$

$$\liminf_{t\to\infty} \sum_{n,c} g_n^{(c)}\left(\overline{R}_n^{(c)}(t)\right) \geq \sum_{n,c} g_n^{(c)}\left(r_n^{*(c)}\right) - \frac{C_2}{V}. \tag{5.44}$$

We note that in the special case when the input rate matrix is *inside* the capacity region, a tighter bound than (5.43) can be computed that does not depend on the V parameter and has a form similar to the bound derived in Theorem 4.5 in Section 4.5 for the differential backlog policy without flow control.

Proof. The proof follows because, given a particular queue state $\boldsymbol{\Theta}(t) = [\boldsymbol{U}(t), \boldsymbol{Y}(t)]$ at time t, the CLC2b algorithm maximizes the right hand side of (5.42) over all alternate choices of the decision variables $R_n^{(c)}(t), \mu_{ab}^{(c)}(t)$, $\gamma_n^{(c)}(t)$. It is not difficult to construct alternate

polices that choose $R_n^{*(c)}(t), \mu_{ab}^{*(c)}(t), \gamma_n^{*(c)}(t)$ to yield for all $(n,c) \in \mathcal{D}$ (see [108]):

$$\mathbb{E}\left\{\sum_b \mu_{nb}^{*(c)}(t) - \sum_a \mu_{an}^{*(c)}(t) \,|\boldsymbol{\Theta}(t)\right\} = r_n^{*(c)}(\epsilon) + \epsilon 1_n^{(c)}$$
$$\gamma_n^{*(c)}(t) = r_n^{*(c)}, \;\; \mathbb{E}\left\{R_n^{*(c)}(t)\,|\boldsymbol{\Theta}\,(t)\right\} = r_n^{*(c)}$$

Plugging these particular decision variables into the right hand side of (5.42) thus preserves the inequality and creates many terms that can be cancelled, yielding:

$$\Delta(\boldsymbol{\Theta}(t)) - V\mathbb{E}\left\{\sum_{n,c} g_n^{(c)}\left(\gamma_n^{(c)}(t)\right)|\boldsymbol{\Theta}(t)\right\} \leq C_2$$

$$-\epsilon\sum_{n,c} U_n^{(c)}(t) - V\sum_{n,c} g_n^{(c)}\left(r_n^{*(c)}(\epsilon)\right).$$

The above inequality is in the exact for for application of the Lyapunov Optimization Theorem (Theorem 5.4), and holds for any value ϵ such that $0 < \epsilon \leq \mu_{sym}$. Applying the theorem and optimizing over all ϵ shows that the queues $\boldsymbol{U}(t)$ are strongly stable with the congestion bound (5.43), and that a utility bound similar to (5.44) holds. A similar argument can be used to prove strong stability of the $\boldsymbol{Y}(t)$ queues, which relates this utility bound directly to (5.44) and proves the theorem (see [108] for details). □

We note that if the constants R_n^{max} and $\hat{R}_n^{(c)}$ are chosen to be equal to some fixed constant $\hat{R}$, then the C_2 constant in Theorem 5.8 is $O(M\hat{R})$, where M is the number of active sessions (n,c) throughout the network. The constant can be reduced by replacing the Type 2 flow control constraints with Type 1 constraints, so that every timeslot and for each node n the $R_n^{(c)}(t)$ variables are chosen to maximize $\sum_{c=1}^K R_n^{(c)}(t)[\eta Y_n^{(c)}(t) - U_n^{(c)}(t)]$ subject to $\sum_c R_n^{(c)}(t) \leq \hat{R}$, $0 \leq R_n^{(c)}(t) \leq L_n^{(c)}(t) + A_n^{(c)}(t)$. The solution of this optimization is still quite simple, as it amounts to admitting as much data as possible from the commodities c with the largest (positive) values of

$\eta Y_n^{(c)}(t) - U_n^{(c)}(t)$. This modification would yield a constant C_2 that is $O((\eta M + N)\hat{R})$, which can be pushed to $O(N\hat{R})$ by choosing η appropriately small.

The average backlog in the virtual queues can also be decreased if, for each source node n, the $\gamma_n^{(c)}(t)$ optimization were replaced by the more complex optimization of maximizing $\sum_c g_n^{(c)}(\gamma_n^{(c)})$ subject to the simplex constraint $\gamma_n^{(c)} \geq 0$, $\sum_c \gamma_n^{(c)} \leq \hat{R}$. In this case, we could choose $\eta = 1$ and still have $C_2 = O(N\hat{R})$. Reducing the average virtual queue backlog is advantageous as it reduces the time required for the network to adapt to possible changes in traffic rates or channel statistics.

5.4.3 An alternative construction on flow state queues

Our presentation of the CLC2b algorithm is somewhat different than the original CLC2 algorithm developed in [108]. The original CLC2 algorithm used Type 1 flow control constraints, and used a different transformation. Specifically, for each utility function $g_n^{(c)}\left(r_n^{(c)}\right)$, a new function $h_n^{(c)}\left(\alpha_n^{(c)}\right)$ was formed:

$$h_n^{(c)}\left(\alpha_n^{(c)}\right) \triangleq g_n^{(c)}(R_{max}^{in}) - g_n^{(c)}\left(R_{max}^{in} - \alpha_n^{(c)}\right),$$

where $\alpha_n^{(c)}(t)$ represents a process of auxiliary variables. Letting $\overline{r}_n^{(c)}$ represent the time average of the flow control decisions $R_n^{(c)}(t)$ and letting $\overline{\alpha}_n^{(c)}$ represent the time average of the auxiliary processes $\alpha_n^{(c)}(t)$, we observe the following:

Minimizing $\sum_{n,c} h_n^{(c)}\left(\overline{\alpha}_n^{(c)}\right)$ *subject to network stability and to the additional constraint* $\overline{\alpha}_n^{(c)} \geq R_{max}^{in} - \overline{r}_n^{(c)}$ *for all* (n,c) *is equivalent to maximizing* $\sum_{n,c} g_n^{(c)}\left(\overline{r}_n^{(c)}\right)$ *subject to network stability.*

To ensure that $\overline{\alpha}_n^{(c)} \geq R_{max}^{in} - \overline{r}_n^{(c)}$, the algorithm of [108] uses a set of flow state queues $\tilde{Y}_n^{(c)}(t)$ with update equations as follows:

$$\tilde{Y}_n^{(c)}(t+1) = \max\left[\tilde{Y}_n^{(c)}(t) - \alpha_n^{(c)}(t), 0\right] + R_{max}^{in} - R_n^{(c)}(t). \qquad (5.45)$$

The CLC2 algorithm performs resource allocation and routing in the same manner as in CLC2b, but chooses flow control values $R_n^{(c)}(t)$ for each source node n to maximize $\sum_{c=1}^{K}\left[\eta\tilde{Y}_n^{(c)}(t) - U_n^{(c)}(t)\right]R_n^{(c)}(t)$ subject to the Type 1 flow control constraints. The auxiliary variables $\alpha_n^{(c)}(t)$ are then computed by maximizing $Vg_n^{(c)}\left(R_{max}^{in} - \alpha_n^{(c)}\right) + \eta\tilde{Y}_n^{(c)}(t)\alpha_n^{(c)}$ subject to $0 \leq \alpha_n^{(c)} \leq R_{max}^{in}$.

5.4.4 Notes

- Simulation results of the CLC2 algorithm for downlinks, $N \times N$ packet switches, and multi-hop networks are found in [108].
- The method of introducing auxiliary variables and flow state queues to solve the flow control problem was developed in [108]. The transformation (5.37)–(5.40) was later considered in [121, 120], and a related use of auxiliary variables is presented in [93]. An alternative approach to utility optimization is developed in [136] using fluid model transformations. It is interesting to note that the algorithm in [136] also keeps additional variables to solve the nonlinear optimization problem, similar to the auxiliary variables and flow state queues developed in this section. A more direct comparison of the two methods is presented in the next section.
- The $[O(1/V), O(V)]$ utility-delay tradeoff achieved by CLC2b is not the optimal tradeoff. A recent result in [120] demonstrates that an improved tradeoff $[O(1/V), O(\log(V))]$ is achievable using a more sophisticated flow control algorithm. Further, [120] shows that, for the special case of one-hop networks, this logarithmic structure of the utility-delay tradeoff cannot be improved by any alternative control strategy. Related work in [15, 119] considers the fundamental *energy-delay* tradeoff for single-user and multi-user wireless systems, where a square-root tradeoff law is established. Energy optimal networking is considered in more detail in the next section.

- When the exogenous traffic arrival rates are outside the capacity region, it may be of interest to control the rate of increase of node queues in order to delay buffer fill-up as long as possible. It turns out that the dynamic back-pressure policy has desirable properties in this respect as well. The study of this problem for networks with fixed link capacities is presented in [53]. For networks with changing link capacities the problem can be dealt with using the methodologies developed in this text. In this case, instead of flow control of exogenous traffic each of the nodes of the network can exercise control of its queues by adding extra "overflow" buffers. A step towards this direction has been taken in [52].

6

Networking with General Costs and Rewards

Here we use Lyapunov optimization theory to develop a framework for optimizing stochastic networks with general cost and reward metrics. The results in this section are well suited to solve problems of *energy optimal networking*, including problems of minimizing average power expenditure in mobile ad-hoc networks, and problems of maximizing network throughput utility subject to average power constraints. The general solution to these problems integrates the basic Lyapunov stability and optimization concepts developed in previous sections, including the use of auxiliary variables and virtual cost queues.

6.1 The network model assumptions

We consider the same network as in the previous section, with N nodes, K commodities (denoted by a node set $\mathcal{N}$ and a commodity set $\mathcal{K}$), a topology state process $S(t)$, and a link transmission rate function $\boldsymbol{C}(I(t), S(t))$. Exogenous arrivals are given by $\boldsymbol{A}(t) = (A_n^{(c)}(t))$. For simplicity, we continue to assume that the pair $[S(t); \boldsymbol{A}(t)]$ is i.i.d. over timeslots, with the understanding that similar results can be extended to non-i.i.d. systems using T-slot analysis.

Control decision variables $I(t)$, $\mu_{ab}^{(c)}(t)$, and $R_n^{(c)}(t)$ for resource allocation, routing, and flow control are also the same as in previous sections, with the exception that we restrict attention to the "Type 2" flow control constraints (see Section 5.1) for simplicity, so that

$$R_n^{(c)}(t) \leq \min[A_n^{(c)}(t) + L_n^{(c)}(t), \hat{R}_n^{(c)}],$$

for suitably large constants $\hat{R}_n^{(c)}$. Specifically, for simplicity we continue to assume that arrivals are deterministically bounded, and choose $\hat{R}_n^{(c)}$ so that $A_n^{(c)}(t) \leq \hat{R}_n^{(c)}$ for all t. The $I(t)$ decisions satisfy $I(t) \in \mathcal{I}_{S(t)}$ for all t, and the $\mu_{ab}^{(c)}(t)$ variables satisfy (5.1) and (5.2). Transport layer storage reservoirs have arbitrary storage space (infinite, finite, or zero). Network layer queueing dynamics are given in (5.3) of Section 5. This general framework can also be used to treat networks *without* flow control by simply adding the additional constraint $A_n^{(c)}(t) = R_n^{(c)}(t)$ for all t. Recall that the sets $\mathcal{L}_c$ restrict routing decisions, and the set $\mathcal{D}$ consists of all (n, c) pairs for which there is a valid queue $U_n^{(c)}(t)$.

6.1.1 Network penalties and rewards

Let $\boldsymbol{x}(t) = (x_1(t), \ldots, x_{M_x}(t))$ represent a vector of M_x *penalties* incurred by the network control decisions $I(t)$, $\mu_{ab}^{(c)}(t)$, $R_n^{(c)}(t)$ at timeslot t, and let $\boldsymbol{y}(t) = (y_1(t), \ldots, y_{M_y}(t))$ represent a vector of M_y *rewards* earned at timeslot t (where M_x and M_y are arbitrary integers). For example, in a network with power allocation decisions, a penalty $x_m(t)$ might represent an arbitrary function of the power expended at one or more nodes during slot t, such as:

- $x_m(t) = \sum_b P_{nb}(t)$ for a given node $n \in \mathcal{N}$ associated with penalty m.
- $x_m(t) = \sum_b (P_{nb}(t))^2 + P_{ab}(t)P_{c,d}(t)$ for a given node $n \in \mathcal{N}$ and some given links (a, b), (c, d) associated with penalty m.
- $x_m(t) = e^{P_{ab}(t)} + \mu_{ab}^{(c)}(t)$ for some link (a, b) and some commodity c associated with penalty m.

Likewise, a reward $y_m(t)$ can be defined arbitrarily, and is usually associated with the flow control variables $R_n^{(c)}(t)$. This abstract

use of network penalties was introduced in [136], where a fluid model transformation was used to analyze performance (the result of [136] is considered in more detail in Section 6.4). Here, we demonstrate how these general penalty and reward functions can be treated using the Lyapunov optimization techniques developed in the previous sections.

We assume that penalties and rewards are non-negative and upper bounded by positive vectors $\boldsymbol{X}^{max}$ and $\boldsymbol{Y}^{max}$, so that $\boldsymbol{0} \leq \boldsymbol{x}(t) \leq \boldsymbol{X}^{max}$ and $\boldsymbol{0} \leq \boldsymbol{y}(t) \leq \boldsymbol{Y}^{max}$ for all time t (inequalities taken entrywise). Let $\overline{\boldsymbol{x}} = (\overline{x}_1, \ldots, \overline{x}_{M_x})$ and $\overline{\boldsymbol{y}} = (\overline{y}_1, \ldots, \overline{y}_{M_y})$ represent the vector of time average penalties and rewards (assuming for now that such time averages exist). We consider the problem of minimizing the sum of a convex increasing function of the long term average penalty vector and a convex decreasing function of the long term average reward vector, subject to network stability and also subject to an additional set of convex constraints on the long term average penalties and rewards.

Specifically, let $f(\boldsymbol{x})$ represent a scalar valued *cost function* associated with the penalty vector $\boldsymbol{x}$. We assume that $f(\boldsymbol{x})$ is non-negative, continuous, convex in the multi-dimensional vector $\boldsymbol{x}$, and entrywise non-decreasing (so that $f(\boldsymbol{x}) \leq f(\boldsymbol{y})$ whenever $\boldsymbol{x} \leq \boldsymbol{y}$, where inequality is taken entrywise). We assume there is a value f_{max} such that $f(\boldsymbol{x}(t)) \leq f_{max}$ for all t. Let the vector valued function $\boldsymbol{q}(\boldsymbol{x}) = (q_1(\boldsymbol{x}), \ldots, q_{J_x}(\boldsymbol{x}))$ represent an additional set of cost functions, where each component function $q_j(\boldsymbol{x})$ is similarly non-negative, continuous, bounded, convex, and entrywise non-decreasing.

Similarly, we let $g(\boldsymbol{y})$, $\boldsymbol{h}(\boldsymbol{y}) = (h_1(\boldsymbol{y}), \ldots, h_{J_y}(\boldsymbol{y}))$, represent *utility functions* associated with the rewards $\boldsymbol{y}$. These functions are assumed to be non-negative, continuous, bounded, concave, and entrywise non-decreasing. The generalized stochastic optimal networking problem is:

$$\begin{aligned} \text{Minimize: } & f(\overline{\boldsymbol{x}}) - g(\overline{\boldsymbol{y}}) \\ \text{Subject to: } & 1)\ \boldsymbol{q}(\overline{\boldsymbol{x}}) \leq \boldsymbol{Q}, \\ & 2)\ \boldsymbol{h}(\overline{\boldsymbol{y}}) \geq \boldsymbol{H}, \\ & 3)\ \text{Network Stability}, \end{aligned} \tag{6.1}$$

where $\boldsymbol{Q} = (Q_1, \ldots, Q_{J_x})$ is a vector of required upper bounds on the cost functions $\boldsymbol{q}(\boldsymbol{x})$, and likewise $\boldsymbol{H}$ is a vector of lower bounds on the

utility functions $\boldsymbol{h}(\boldsymbol{y})$. We define Λ_Q as the modified capacity region for this network, consisting of all arrival rate matrices that can be stably supported by the network layer under the additional cost constraints $\boldsymbol{q}(\overline{\boldsymbol{x}}) \leq \boldsymbol{Q}$, $\boldsymbol{h}(\overline{\boldsymbol{y}}) \geq \boldsymbol{H}$ imposed on the network penalties and rewards.

Note that the general problem can be stated purely in terms of penalties. Indeed, the reward objective is equivalent to minimizing the function $\tilde{f}(\tilde{\boldsymbol{x}}(t))$, where $\tilde{\boldsymbol{x}}(t) \triangleq \boldsymbol{Y}^{max} - \boldsymbol{y}(t)$, and where:

$$\tilde{f}(\tilde{\boldsymbol{x}}(t)) \triangleq g(\boldsymbol{Y}^{max}) - g(\boldsymbol{Y}^{max} - \tilde{\boldsymbol{x}}(t)). \tag{6.2}$$

Because $g(\boldsymbol{y})$ is concave and entrywise non-decreasing, the function $\tilde{f}(\tilde{\boldsymbol{x}})$ is convex and entrywise non-decreasing, and therefore fits into our general framework. This transformation is used to treat the problem of fair flow control in [108] (as discussed in Section 5, Section 5.4.3). Likewise, penalties can be changed into rewards through a similar transformation. However, it is often useful to maintain a distinction between penalties and rewards, especially for implementation purposes.

6.1.2 Assumptions on optimal stationary control

Let $c^* = f^* - g^*$ represent the optimal cost associated with the problem (6.1), and let $\boldsymbol{x}^*$, $\boldsymbol{y}^*$ represent the optimal time average penalty and reward vectors, so that $f(\boldsymbol{x}^*) = f^*$, $g(\boldsymbol{y}^*) = g^*$. We consider the class of systems for which optimality can be achieved within the class of *S-only* algorithms, defined as follows:

Definition 6.1. An *S-only* algorithm is a stationary randomized algorithm that chooses control variables $I(t)$, $\{R_n^{(c)}(t)\}$, $\{\mu_{ab}^{(c)}(t)\}$ based only on the current topology state $S(t)$.

While our dynamic control algorithm will base decisions on current queue backlog and hence is not S-only, its performance will be evaluated by comparison to S-only algorithms. Specifically, we assume:

Assumption 1: (Optimality of an S-only Policy) There exists an S-only algorithm such that for all t and all node-commodity pairs

$(n,c) \in \mathcal{D}$ we have:

$$\sum_a \mathbb{E}\left\{\mu_{an}^{(c)}(t)\right\} + \mathbb{E}\left\{R_n^{(c)}(t)\right\} = \sum_b \mathbb{E}\left\{\mu_{nb}^{(c)}(t)\right\}, \tag{6.3}$$

$$\boldsymbol{q}(\mathbb{E}\{\boldsymbol{x}(t)\}) \leq \boldsymbol{Q}, \quad \boldsymbol{h}(\mathbb{E}\{\boldsymbol{y}(t)\}) \geq \boldsymbol{H},$$

$$f(\mathbb{E}\{\boldsymbol{x}(t)\}) - g(\mathbb{E}\{\boldsymbol{y}(t)\}) = f^* - g^*, \tag{6.4}$$

where the expectations above are taken over the random topology state $S(t)$ and the potentially randomized control decisions.

Further, we make the following *interior point* and *slackness* assumptions (which hold for all of the networks considered in this text):

Assumption 2: (Interior Point) There exists a value $\epsilon_{max} > 0$ together with an S-only algorithm such that for all t and all node-commodity pairs $(n,c) \in \mathcal{D}$ we have:

$$\sum_a \mathbb{E}\left\{\mu_{an}^{(c)}(t)\right\} + \epsilon_{max} + \mathbb{E}\left\{R_n^{(c)}(t)\right\} = \sum_b \mathbb{E}\left\{\mu_{nb}^{(c)}(t)\right\},$$

$$\boldsymbol{q}(\mathbb{E}\{\boldsymbol{x}(t)\}) \leq \boldsymbol{Q}, \quad \boldsymbol{h}(\mathbb{E}\{\boldsymbol{y}(t)\}) \geq \boldsymbol{H}.$$

Assumption 3: (Slackness 1) There exist positive values δ_1, δ_2 together with an S-only algorithm that yields (6.3) and that simultaneously yields:

$$\boldsymbol{q}(\mathbb{E}\{\boldsymbol{x}(t)\}) \leq \boldsymbol{Q} - \delta_1 \mathbf{1}_q, \quad \boldsymbol{h}(\mathbb{E}\{\boldsymbol{y}(t)\}) \geq \boldsymbol{H} + \delta_2 \mathbf{1}_h.$$

where $\mathbf{1}_q$ is a vector with all entries equal to 0 or 1, with entry i equal to 1 if and only if function $q_i(\cdot)$ is not identically zero, and $\mathbf{1}_h$ is a 0/1 vector with entry j equal to 1 if and only of $h_j(\cdot)$ is not identically zero.

Assumption 4: (Slackness 2) There exist positive values δ_3, δ_4 and vectors $\boldsymbol{\alpha}$, $\boldsymbol{\gamma}$ such that $\boldsymbol{q}(\boldsymbol{\alpha}) \leq \boldsymbol{Q}$, $\boldsymbol{h}(\boldsymbol{\gamma}) \geq \boldsymbol{H}$ and $\mathbf{0} \leq \boldsymbol{\alpha} \leq \boldsymbol{X}^{max}$, $\mathbf{0} \leq \boldsymbol{\gamma} \leq \boldsymbol{Y}_{max}$, such that there exists an S-only algorithm yielding (6.3) while simultaneously yielding:

$$\mathbb{E}\{\boldsymbol{x}(t)\} \leq \boldsymbol{\alpha} - \delta_3 \mathbf{1}_x, \quad \mathbb{E}\{\boldsymbol{y}(t)\} \geq \boldsymbol{\gamma} + \delta_4 \mathbf{1}_y.$$

where $\mathbf{1}_x$ is a 0/1 vector with entry i equal to 1 if and only if penalty $x_i(t)$ is not identically zero for all t, and $\mathbf{1}_y$ is a 0/1 vector with entry j equal to 1 if and only if reward $y_j(t)$ is not identically zero for all t.

Note that the quantities $\mathbb{E}\{\mu_{nb}^{(c)}(t)\}$ can be viewed as flows $f_{nb}^{(c)}$, and hence the control decisions of the S-only policy of Assumption 1

yield flows that satisfy the flow conservation constraints, while also satisfying the cost and utility inequality constraints and yielding time average penalties and rewards that achieve the minimum network cost $f(\overline{\boldsymbol{x}}) - g(\overline{\boldsymbol{y}}) = f^* - g^*$. The value $c^* = f^* - g^*$ can thus be interpreted as the optimal cost over all S-only policies. For a large class of network flow problems, including the fairness problems of the previous section and the energy problems to be presented in this section, c^* is also the optimal cost over *all possible* (perhaps not S-only) policies [116].

Assumption 2 effectively states that there exists an S-only policy that meets all inequality constraints while supporting a traffic matrix $(\mathbb{E}\{R_n^{(c)}(t)\} + \epsilon 1_n^{(c)})$ (where $1_n^{(c)}$ is equal to 1 if $(n,c) \in \mathcal{D}$, and zero else). Suppose S-only policies can be used to support any traffic matrix within the relative interior of the network capacity region Λ_Q, and define μ_{sym} as the edge size of the largest D-dimensional hypercube that can fit inside the effective dimensions of Λ_Q (with one vertex at the origin). If $\mu_{sym} > 0$ and if the functions $\boldsymbol{q}(\boldsymbol{x})$ and $\boldsymbol{h}(\boldsymbol{y})$ do not depend on penalties or rewards associated with the flow control decision variables $R_n^{(c)}(t)$, then Assumption 2 is satisfied. This can be seen by setting $R_n^{(c)}(t) = 0$ for all (n,c) and all t, so that ϵ_{max} is equal to μ_{sym}. In the case when there are no flow controllers (so that $A_n^{(c)}(t) = R_n^{(c)}(t)$ for all (n,c) and all t), the value of ϵ_{max} is the largest value of ϵ such that $(\lambda_n^{(c)} + \epsilon 1_n^{(c)}) \in \Lambda_Q$.

Assumption 3 states that there exists an S-only algorithm that satisfies the flow constraints while also yielding the strict inequalities $q_i(\mathbb{E}\{\boldsymbol{x}(t)\}) < Q_i$ and $h_i(\mathbb{E}\{\boldsymbol{y}(t)\}) > H_i$ whenever the corresponding $q_i(\cdot)$ and $h_i(\cdot)$ functions are not identically zero and hence define legitimate constraints. Assumption 4 is similar. In many networks with flow control and/or without cost or utility constraint functions $\boldsymbol{q}(\cdot), \boldsymbol{h}(\cdot)$, assumptions 3 and 4 can be shown to *trivially hold* simply by considering the policy that sets all control variables to zero or near-zero.

6.1.3 Example: A wireless uplink with power constraints

Consider a wireless uplink where L users transmit to a base-station over L time varying channels. Let $\boldsymbol{C}(\boldsymbol{P}(t), S(t)) = (C_1(\boldsymbol{P}(t), S(t)), \ldots, C_L(\boldsymbol{P}(t), S(t)))$ represent the link transmission rate vector

as a function of the current topology state $S(t)$ and the power allocation vector $\boldsymbol{P}(t) = (P_1(t), \ldots, P_L(t))$. The particular form of the $\boldsymbol{C}(\boldsymbol{P}(t), S(t))$ function depends on the physical layer transmission and multi-user detection schemes used by this system. Suppose that power allocation is restricted so that $0 \leq P_i(t) \leq P_{max}$ for all users i and for all t. Separate queues are maintained at each user, and we let $R_i(t)$ represent the amount of data user i decides to add to its queue on timeslot t. We assume that $0 \leq R_i(t) \leq R_{max}$ for some constant R_{max}. Suppose that the transport layer reservoirs at each user are infinite with infinite backlog.

Consider the following network parameters, with penalties $\boldsymbol{x}(t)$ and rewards $\boldsymbol{y}(t)$ defined according to power allocation and flow control decisions:

- $\boldsymbol{x}(t) = (P_1(t), \ldots, P_L(t))$, $\boldsymbol{y}(t) = (R_1(t), \ldots, R_L(t))$
- $\boldsymbol{q}(\boldsymbol{x}) = \boldsymbol{x}$, $\boldsymbol{Q} = (.1, .1, \ldots, .1)$ Watts; $\boldsymbol{h}(\boldsymbol{y}) = \boldsymbol{H} = \boldsymbol{0}$
- $f(\boldsymbol{x}) - g(\boldsymbol{y}) = \beta_1 \sum_{i=1}^{L} x_i^2 - \beta_2 \sum_{i=1}^{L} \log(1 + y_i)$

where β_1, β_2 are non-negative constants that weight the relative importance of energy cost and throughput utility. Thus, in this simple example, the penalties $\boldsymbol{x}(t)$ correspond to power allocations and the rewards $\boldsymbol{y}(t)$ correspond to transport layer admission decisions. The inequality constraint $\boldsymbol{q}(\overline{\boldsymbol{x}}) \leq \boldsymbol{Q}$ ensures that the average power expended by each user is no more than $0.1\, Watts$. The global cost function $f(\boldsymbol{x}) - g(\boldsymbol{y})$ contains a negative throughput utility term and a positive term that is quadratic in the power penalties. Such a quadratic cost on power expenditure might be defined to provide a measure of "energy fairness" among the different users, so that the resulting average power costs are balanced more evenly across users.

It can be shown that Assumption 1 holds for this example, and that Assumption 2 holds whenever the capacity region Λ_Q contains a positive vector $(\epsilon, \ldots, \epsilon)$ [116]. If $\boldsymbol{C}(\boldsymbol{0}, S(t)) = \boldsymbol{0}$ for all $S(t)$ (so that zero power yields zero transmission rate), then Assumption 3 trivially holds by assigning $\boldsymbol{R}(t) = \boldsymbol{0}$, $\boldsymbol{P}(t) = \boldsymbol{0}$, $(\mu_{ab}^{(c)}(t)) = (0)$ for all t, so that (6.3) holds, and

$$\boldsymbol{q}(\mathbb{E}\{\boldsymbol{x}(t)\}) = \boldsymbol{q}(\boldsymbol{0}) = \boldsymbol{0} < \boldsymbol{Q} = (0.1, \ldots, 0.1),$$

and hence Assumption 3 is satisfied for $\delta_1 = 0.1$. Further, choosing $\boldsymbol{\alpha} = \boldsymbol{Q}$, $\boldsymbol{\gamma} = \boldsymbol{0}$, and choosing positive δ_3, δ_4 values such that there exists an S-only algorithm to support a throughput vector $(\delta_4, \ldots, \delta_4)$ subject to the reduced average power constraint $\boldsymbol{Q} - \delta_3 \boldsymbol{1}_x$ ensures that Assumption 4 is satisfied.

6.2 Algorithm design

Here we specify a dynamic control algorithm that does not require knowledge of traffic statistics or topology state probabilities, yet meets all network constraints while yielding a total network cost that is arbitrarily close to the minimum cost $c^* = f^* - g^*$, with a corresponding trade-off in average end-to-end network delay. To motivate the control algorithm, note that the optimization problem (6.1) is equivalent to the following modified problem that introduces new variables $\boldsymbol{\alpha}$ and $\boldsymbol{\gamma}$:

$$\begin{aligned} \text{Minimize: } & f(\boldsymbol{\alpha}) - g(\boldsymbol{\gamma}) \\ \text{Subject to: } & 1)\ \boldsymbol{q}(\boldsymbol{\alpha}) \leq \boldsymbol{Q}, \quad \boldsymbol{h}(\boldsymbol{\gamma}) \geq \boldsymbol{H}, \\ & 2)\ \overline{\boldsymbol{x}} \leq \boldsymbol{\alpha}, \quad \overline{\boldsymbol{y}} \geq \boldsymbol{\gamma}, \\ & 3)\ \text{Network Stability.} \end{aligned}$$

As in Section 5, we have introduced a new vector $\boldsymbol{\alpha}$ to decouple the penalty variables $\boldsymbol{x}$ from the cost functions $f(\cdot)$ and $\boldsymbol{q}(\cdot)$, and have similarly introduced a new vector $\boldsymbol{\gamma}$ to decouple the rewards from the utility functions. The decoupling vector $\boldsymbol{\alpha}$ is not necessary in the special case when cost functions $f(\cdot)$ and $\boldsymbol{q}(\cdot)$ are both *linear*.[1] Likewise, the vector $\boldsymbol{\gamma}$ is not necessary when $\boldsymbol{h}(\cdot)$ and $g(\cdot)$ are both linear. However, these extra vectors are important for the treatment of general (potentially non-linear) convex cost functions and concave utility functions. Let $\boldsymbol{\alpha}(t)$ and $\boldsymbol{\gamma}(t)$ represent a process of these new vector variables, to be chosen as control parameters on each timeslot. To ensure that the first set of inequality constraints in the above problem are satisfied,

[1] Using decoupling vectors $\boldsymbol{\alpha}$ in the case of linear costs cannot hurt, but linearity can be exploited to design a somewhat simpler algorithm (see Section 6.3).

we define *virtual cost queues* $\boldsymbol{D}_q(t)$, $\boldsymbol{D}_h(t)$, with update equations as follows:

$$\boldsymbol{D}_q(t+1) = \max[\boldsymbol{D}_q(t) - \boldsymbol{Q}, \boldsymbol{0}] + \boldsymbol{q}(\boldsymbol{\alpha}(t)), \tag{6.5}$$

$$\boldsymbol{D}_h(t+1) = \max[\boldsymbol{D}_h(t) - \boldsymbol{h}(\boldsymbol{\gamma}(t)), \boldsymbol{0}] + \boldsymbol{H}. \tag{6.6}$$

Likewise, to ensure the second set of inequality constraints are satisfied, we define virtual queues $\boldsymbol{Z}_x(t)$ and $\boldsymbol{Z}_y(t)$, which are generalized versions of the *flow state queues* developed in the previous section, and have update equations:

$$\boldsymbol{Z}_x(t+1) = \max[\boldsymbol{Z}_x(t) - \boldsymbol{\alpha}(t), \boldsymbol{0}] + \boldsymbol{x}(t), \tag{6.7}$$

$$\boldsymbol{Z}_y(t+1) = \max[\boldsymbol{Z}_y(t) - \boldsymbol{y}(t), \boldsymbol{0}] + \boldsymbol{\gamma}(t). \tag{6.8}$$

Note that the $\boldsymbol{D}_q(t)$ variables can be viewed as backlogs in a virtual queue with input process $\boldsymbol{q}(\boldsymbol{\alpha}(t))$ and constant service rate $\boldsymbol{Q}$. Thus, if the $\boldsymbol{D}_q(t)$ process is strongly stable then the time average expected arrival rate is less than or equal to $\boldsymbol{Q}$ (ensuring the inequality constraint $\boldsymbol{Q} \geq \overline{\boldsymbol{q}(\boldsymbol{\alpha})} \geq \boldsymbol{q}(\overline{\boldsymbol{\alpha}})$). Similarly, stabilizing the other queues ensures the other inequality constraints are satisfied. This technique of transforming stochastic inequality constraints into queueing stability problems generalizes the *virtual power queue* technique introduced in [116] for stabilizing wireless networks subject to average power constraints.

The goal of the network controller is to minimize $f(\overline{\boldsymbol{\alpha}}) - g(\overline{\boldsymbol{\gamma}})$ while stabilizing all actual and virtual queues in the network. To this end, let $\boldsymbol{\Theta}(t) = [\boldsymbol{U}(t), \boldsymbol{D}_q(t), \boldsymbol{D}_h(t), \boldsymbol{Z}_x(t), \boldsymbol{Z}_y(t)]$ represent the combined network state vector, and define the following quadratic Lyapunov function:

$$L(\boldsymbol{\Theta}) = \frac{1}{2}\left[\sum_{n,c}(U_n^{(c)})^2 + \sum_{i=1}^{J_q} D_{i,q}^2 + \sum_{j=1}^{J_h} D_{j,h}^2 + \sum_{m=1}^{M_x} Z_{m,x}^2 + \sum_{l=1}^{M_y} Z_{l,y}^2\right], \tag{6.9}$$

where $D_{i,q}, D_{j,h}$ represent the i and j entries of vectors $\boldsymbol{D}_q$ and $\boldsymbol{D}_h$, respectively, and $Z_{m,x}, Z_{l,y}$ represent the m and l entries of vectors $\boldsymbol{Z}_x$

and $\boldsymbol{Z}_y$, respectively. Define the one-step Lyapunov drift $\Delta(\boldsymbol{\Theta}(t))$ as follows:

$$\Delta(\boldsymbol{\Theta}(t)) \triangleq \mathbb{E}\{L(\boldsymbol{\Theta}(t+1)) - L(\boldsymbol{\Theta}(t))|\boldsymbol{\Theta}(t)\}.$$

Motivated by the Lyapunov Optimization Theorem (Theorem 5.4 of Section 5), we proceed by designing a controller that, every timeslot, minimizes a bound on the following metric:

$$\text{Minimize:} \quad \Delta(\boldsymbol{\Theta}(t)) + V\mathbb{E}\{f(\boldsymbol{\alpha}(t)) - g(\boldsymbol{\gamma}(t))|\boldsymbol{\Theta}(t)\}, \tag{6.10}$$

where $V > 0$ is a control parameter that affects a performance-delay trade-off.

6.2.1 The generalized CLC algorithm (GCLC)

An expression for the drift metric (6.10) can be computed using the dynamic queueing laws (5.3), (6.5)–(6.8) in the same manner as in previous sections. Below we state the result, omitting the details for brevity:

$$\begin{aligned}
&\Delta(\boldsymbol{\Theta}(t)) + V\mathbb{E}\{f(\boldsymbol{\alpha}(t)) - g(\boldsymbol{\gamma}(t))|\boldsymbol{\Theta}(t)\} \\
&\quad \leq B - \sum_{n,c} U_n^{(c)}(t)\mathbb{E}\left\{\sum_b \mu_{nb}^{(c)}(t) - \sum_a \mu_{an}^{(c)}(t) - R_n^{(c)}(t)\middle|\boldsymbol{\Theta}(t)\right\} \\
&\qquad - \sum_{i=1}^{J_q} D_{i,q}(t)\left[Q_i - \mathbb{E}\{q_i(\boldsymbol{\alpha}(t))|\boldsymbol{\Theta}(t)\}\right] \\
&\qquad - \sum_{j=1}^{J_h} D_{j,h}(t)\left[\mathbb{E}\{h_j(\boldsymbol{\gamma}(t))|\boldsymbol{\Theta}(t)\} - H_j\right] \\
&\qquad - \sum_{m=1}^{M_x} Z_{m,x}(t)\mathbb{E}\{\alpha_m(t) - x_m(t)|\boldsymbol{\Theta}(t)\} \\
&\qquad - \sum_{l=1}^{M_y} Z_{l,y}(t)\mathbb{E}\{y_l(t) - \gamma_l(t)|\boldsymbol{\Theta}(t)\} \\
&\qquad + V\mathbb{E}\{f(\boldsymbol{\alpha}(t)) - g(\boldsymbol{\gamma}(t))|\boldsymbol{\Theta}(t)\},
\end{aligned} \tag{6.11}$$

where B is a finite and positive constant. Every timeslot, the control decision variables are chosen to minimize the right hand side of the above inequality.

In order to express the algorithm as a decoupled set of flow control decisions and routing/resource allocation decisions, we assume the following *separability criteria*:

- The penalty vector $\boldsymbol{x}(t)$ depends only on the routing and resource allocation variables $I(t)$ and $\mu_{ab}^{(c)}(t)$ (and not on the flow control variables).
- The reward vector $\boldsymbol{y}(t)$ depends only on the flow control variables $R_n^{(c)}(t)$ (and not on the resource allocation and routing variables).

The above separability criteria leads to the following dynamic control algorithm.

Generalized Cross-Layer Control Algorithm (GCLC):

Flow Control: The flow controllers for each active input (n, c) observe the virtual and actual queue backlogs, and variables $R_n^{(c)}(t)$ are chosen to solve the following optimization problem:

$$\text{Minimize: } \sum_{n,c} U_n^{(c)}(t) R_n^{(c)}(t) - \sum_{l=1}^{M_y} y_l(t) Z_{l,y}(t),$$
$$\text{Subject to: } 0 \leq R_n^{(c)}(t) \leq \min[A_n^{(c)}(t) + L_n^{(c)}(t), \hat{R}_n^{(c)}],$$

The values $\gamma_l(t)$ are then computed as follows:

$$\text{Minimize: } -Vg(\boldsymbol{\gamma}) + \sum_{l=1}^{M_y} \gamma_l Z_{l,y}(t) - \sum_{j=1}^{J_h} D_{j,h}(t) h_j(\boldsymbol{\gamma}),$$
$$\text{Subject to: } 0 \leq \gamma_l \leq Y_l^{max} \quad \text{for all } l \in \{1, \ldots, M_y\}.$$

The virtual queues $\boldsymbol{Z}_y(t)$ and $\boldsymbol{D}_h(t)$ are then updated according to (6.8) and (6.6), using the $\boldsymbol{\gamma}(t)$ values computed above, and using the $\boldsymbol{y}(t)$ rewards associated with the flow control decisions computed above.

Routing/Resource Allocation: The topology state $S(t)$ and queue backlogs are observed, and the control input $I(t) \in \mathcal{I}_{S(t)}$ and routing

variables $\mu_{ab}^{(c)}(t)$ are chosen according to the following optimization:

$$\begin{aligned}\text{Minimize: } & -\sum_{n,b,c} W_{nb}^{(c)}(t)\mu_{nb}^{(c)}(t) + \sum_{m=1}^{M_x} Z_{m,x}(t)x_m(t),\\ \text{Subject to: } & \sum_c \mu_{nb}^{(c)}(t) \leq C_{nb}(I(t),S(t)) \quad \text{for all } (n,b),\\ & I \in \mathcal{I}_{S(t)}, \quad \mu_{nb}^{(c)}(t) = 0 \quad \text{if } (n,b) \notin \mathcal{L}_c,\end{aligned}$$

where:

$$W_{nb}^{(c)}(t) \triangleq \max\left[U_n^{(c)}(t) - U_b^{(c)}(t), 0\right].$$

Each link (n,b) then transmits $\mu_{nb}^{(c)}(t)$ units of commodity c data (using idle fill if necessary), provided that $W_{nb}^{(c)}(t) > 0$ and $(n,b) \in \mathcal{L}_c$.

The $\boldsymbol{\alpha}(t)$ values are then computed as solutions of the following optimization problem:

$$\begin{aligned}\text{Minimize: } & Vf(\boldsymbol{\alpha}) - \sum_{m=1}^{M_x} \alpha_m Z_{m,x}(t) + \sum_{i=1}^{J_q} D_{i,q}(t)q_i(\boldsymbol{\alpha}),\\ \text{Subject to: } & 0 \leq \alpha_m \leq X_m^{max} \quad \text{for all } m \in \{1,\ldots,M_x\}.\end{aligned}$$

The virtual queues $\boldsymbol{D}_q(t)$ and $\boldsymbol{Z}_x(t)$ are then updated according to (6.5) and (6.7).

Assuming the flow control rewards $y_l(t)$ are associated with distinct nodes, and that the utility functions $g(\boldsymbol{y})$ and $\boldsymbol{h}(\boldsymbol{y})$ are sums of individual utilities for each node, it is not difficult to show that the flow control optimization can be distributed node by node, using only the local (virtual and actual) queue backlog information for that node. The routing algorithm can be further decoupled from resource allocation if penalties depend only on $I(t)$ (and not on the variables $\mu_{nb}^{(c)}(t)$ if $I(t)$ is given), in which case an optimal commodity c_{nb}^* can be found as in previous sections, and the resulting resource allocation problem reduces to the backpressure algorithm of maximizing:

$$\sum_{nb} W_{nb}^*(t)C_{nb}(I(t),S(t)) - \sum_{m=1}^{M_x} Z_{m,x}(t)x_m(t). \tag{6.12}$$

The algorithm for choosing $\boldsymbol{\alpha}(t)$ can be distributed node-by-node provided that cost functions are separable. The resource allocation algorithm for choosing $I(t)$ to maximize (6.12) is the most complex part of the algorithm, but is easily distributed in the case when transmission penalty functions are separable and each node transmits over orthogonal frequency bands, or is approximated via methods discussed in previous sections.

6.2.2 Algorithm performance

Let Assumptions 1-4 from Section 6.1.2 hold. It is important to note that Assumption 1 is valid if we assume either of the following two cases:

(1) The flow control rewards $\boldsymbol{y}(t)$ are *linear functions* of the decision variables $\boldsymbol{R}(t)$ (the utilities $\boldsymbol{h}(\cdot)$ and $g(\cdot)$ can still be nonlinear). The reservoir buffer size is arbitrary (infinite, finite, or zero).
(2) The flow control rewards $\boldsymbol{y}(t)$ are *arbitrary functions* of the decision variables $\boldsymbol{R}(t)$. In this case, the reservoir buffer size *possibly affects optimality*, but Assumption 1 holds *if we assume zero reservoir space* (so that all admission/rejection decisions are made immediately upon arrival) or assume *infinite and always full reservoir space.*

Theorem 6.2. (Algorithm Performance) Under Assumptions 1-4, the GCLC algorithm stabilizes all actual and virtual queues in the network. Furthermore, there exists a finite constant B (which can be explicitly found by computing the Lyapunov drift of the system), such that:

$$\limsup_{t\to\infty} \frac{1}{t}\sum_{\tau=0}^{t-1}\sum_{n,c}\mathbb{E}\left\{U_n^{(c)}(\tau)\right\} \leq \frac{B+V(f_{max}+g_{max})}{\epsilon_{max}},$$

$$\limsup_{t\to\infty} \boldsymbol{q}(\overline{\boldsymbol{x}}(t)) \leq \boldsymbol{Q},$$

$$\liminf_{t\to\infty} \boldsymbol{h}(\overline{\boldsymbol{y}}(t)) \geq \boldsymbol{H},$$

$$\limsup_{t\to\infty}[f(\overline{\boldsymbol{x}}(t)) - g(\overline{\boldsymbol{y}}(t))] \leq f^* - g^* + B/V,$$

where $\overline{x}_m(t) \triangleq \frac{1}{t}\sum_{\tau=0}^{t-1}\mathbb{E}\{x_m(\tau)\}$ and $\overline{y}_l(t) \triangleq \frac{1}{t}\sum_{\tau=0}^{t-1}\mathbb{E}\{y_l(\tau)\}$, and where f_{max} and g_{max} are constants such that $f(x(t)) \leq f_{max}$ and $g(y(t)) \leq g_{max}$ for all t.

The proof of the above theorem uses techniques similar to those presented in the previous section and developed in [108, 115, 116], and is omitted for brevity. It is important to note that the proof works for any system that satisfies Assumptions 1-4 (for a given, potentially sub-optimal constant $c^* = f^* - g^*$ on the right hand side of (6.4)), provided that the virtual queues are maintained as in (6.5)–(6.8) and that control decision variables are chosen every timeslot to minimize the right hand side of (6.11).

6.3 Energy optimal networking examples

Here we apply the general framework to several important problems in the area of energy-aware wireless networking.

6.3.1 Max throughput with average power constraints

Consider a multi-user wireless downlink that transmits to L users over L distinct channels. Let $S(t)$ describe the collective channel state process, and let $\boldsymbol{C}(\boldsymbol{P}(t), S(t)) = (C_1(\boldsymbol{P}(t), S(t)), \ldots, C_L(\boldsymbol{P}(t), S(t)))$ represent the link transmission rate function, where $\boldsymbol{P}(t) = (P_1(t), \ldots, P_L(t))$ is the vector of power allocations for each link. Power is constrained so that $\sum_{i=1}^{L} P_i(t) \leq P_{max}$ for all t. The goal is to maximize throughput subject to an average power constraint P_{av} (where $P_{av} < P_{max}$). Arrival processes $A_i(t)$ satisfy $A_i(t) \leq \hat{R}$ for all i. Assume there are no transport layer storage reservoirs, so that all arriving data is either admitted or dropped. Let $R_i(t)$ represent the flow control decision for queue i at time t, and let $U_i(t)$ represent the current backlog in queue i. In this case, we have:

- $\boldsymbol{x}(t) = (P_1(t), \ldots, P_L(t))$, $\boldsymbol{y}(t) = (R_1(t), \ldots, R_L(t))$
- $q(\boldsymbol{P}) = \sum_{i=1}^{L} P_i$, $Q = P_{av}$; $h(\boldsymbol{R}) = H = 0$
- $f(\boldsymbol{P}) = 0$, $g(\boldsymbol{R}) = \sum_{i=1}^{L} R_i$

Because $h(\boldsymbol{R}) = 0$, there is no utility constraint and hence we do not use any $\boldsymbol{D}_h(t)$ queues. A further simplification results by noticing that the cost functions $q(\boldsymbol{P}), f(\boldsymbol{P})$ are *linear*. Hence, we can avoid the virtual queues $\boldsymbol{Z}_x(t)$ and simply use $\boldsymbol{\alpha}(t) = \boldsymbol{x}(t) = \boldsymbol{P}(t)$ everywhere (including the virtual queue definitions, and in the right hand side of the drift bound (6.11)), without loss of optimality. Likewise, the utility function $g(\boldsymbol{R})$ is linear, and hence we can avoid the virtual queues $\boldsymbol{Z}_y(t)$ and simply use $\boldsymbol{\gamma}(t) = \boldsymbol{y}(t) = \boldsymbol{R}(t)$.

For the average power constraint, we define a *virtual power queue* $D_q(t)$ (we shall use $D(t)$ for simplicity here) with a queueing law (6.5) that reduces to the following by setting $\boldsymbol{\alpha}(t) = \boldsymbol{P}(t)$:

$$D(t+1) = \max[D(t) - P_{av}, 0] + \sum_{i=1}^{L} P_i(t). \tag{6.13}$$

with initial condition $D(0) = 0$. In this context, $D(t)$ has the intuitive interpretation of being the accumulated excess power expenditure over and above the average power constraint. With these simplifications, minimizing the right hand side of (6.11) leads to the following Energy Constrained Control Algorithm (ECCA) from [116]:

Energy Constrained Control Algorithm (ECCA) [116]:

Flow Control: Minimize $\sum_{i=1}^{L}[U_i(t) - V]R_i(t)$ such that $0 \leq R_i(t) \leq A_i(t)$ for all i. That is, every timeslot and for each queue i, we allow the full set of new arrivals $A_i(t)$ into the queue whenever $U_i(t) \leq V$. Else, we drop all new arrivals for queue i entering on that timeslot.

Power Allocation: Every timeslot, observe the current topology state $S(t)$ and the current queue backlogs $\boldsymbol{U}(t)$ and $D(t)$, and allocate power $\boldsymbol{P}(t) = (P_1(t), \ldots, P_L(t))$ according to the following optimization:

$$\text{Maximize: } \sum_{i=1}^{L}[U_i(t)C_i(\boldsymbol{P}, S(t)) - D(t)P_i(t)] \tag{6.14}$$

$$\text{Subject to: } \sum_{i=1}^{L} P_i(t) \leq P_{max}.$$

The virtual power queue $D(t)$ is then updated via (6.13).

From the nature of the flow control algorithm, we have that $U_i(t) \leq V + \hat{R}$ for all i and all t, and hence the above algorithm trivially stabilizes all actual queues. Using this fact, it is not difficult to show that the power allocation algorithm (6.14) *allocates zero power on any timeslot in which the virtual queue $D(t)$ is sufficiently large.* Specifically, assume there exists a constant $\beta > 0$ such that for any link i, any topology state $S(t)$, and any power vector $\boldsymbol{P}$ such that $\sum_i P_i \leq P_{max}$, we have:

$$C_i(\boldsymbol{P}, S(t)) \leq C_i(\boldsymbol{P}[i], S(t)) + \beta P_i,$$

where $\boldsymbol{P} = (P_1, \ldots, P_L)$, and $\boldsymbol{P}[i]$ is equal to $\boldsymbol{P}$ with the exception that the ith entry is set to zero. It follows that if $D(t) > U_i(t)\beta$, then $P_i(t) = 0$. This leads to the following theorem.

Theorem 6.3. (ECCA Performance [116]) For any topology state process $S(t)$ and any input process $\boldsymbol{A}(t)$ that satisfies $A_i(t) \leq \hat{R}$ for all t, the ECCA algorithm ensures:

$$U_i(t) \leq U_{max} \triangleq V + \hat{R} \quad \text{for all } i \text{ and all } t,$$
$$D(t) \leq D_{max} \triangleq \beta V + \beta \hat{R} + P_{max} \quad \text{for all } t.$$

Consequently, the energy expended over any interval of T slots is never more than $P_{av}T + D_{max}$ (and so the average power constraint clearly holds). Further, if the arrival vector $\boldsymbol{A}(t)$ and topology state $S(t)$ is i.i.d. over timeslots, then achieved throughput satisfies:

$$\liminf_{t\to\infty} \frac{1}{t} \sum_{\tau=0}^{t-1} \sum_{i=1}^{L} \mathbb{E}\{R_i(\tau)\} \geq \sum_i r_i^* - B/V,$$

where $\sum_i r_i^*$ is the optimal throughput over all possible policies, and B is a constant that can be computed from the Lyapunov drift of the system [116].

The maximum throughput objective can be replaced by a concave fairness objective, such as $g(\boldsymbol{R}) = \sum_i \log(1 + R_i)$, in which case the flow control portion of the ECCA algorithm can be modified by using auxiliary variables $(\gamma_1(t), \ldots, \gamma_L(t))$ and flow state queues $\boldsymbol{Z}_y(t) = (Z_{1,y}(t), \ldots, Z_{L,y}(t))$ according to the GCLC algorithm.

6.3.2 Minimizing power expenditure in mobile networks

Consider a mobile wireless network with topology state $S(t)$ and link transmission rate function $\boldsymbol{C}(\boldsymbol{P}(t), S(t)) = (C_{ab}(\boldsymbol{P}(t), S(t)))$, where $\boldsymbol{P}(t) = (P_{ab}(t))$ is the matrix of power allocations over each link. Let $\mathcal{P}$ represent the set of feasible power allocation vectors, so that $\boldsymbol{P}(t) \in \mathcal{P}$ for all t. Suppose that, without average power constraints on each node, the network capacity region is given by the set Λ.

Suppose that there is no flow control, so that all arriving data is admitted immediately into the network layer (so that $A_n^{(c)}(t) = R_n^{(c)}(t)$ for all t). Assume that the resulting traffic matrix $(\lambda_n^{(c)})$ is in the relative interior of Λ, and define ϵ_{max} to be the largest scalar such that $(\lambda_n^{(c)} + \epsilon_{max} 1_n^{(c)}) \in \Lambda$ (where $1_n^{(c)}$ is an indicator function that takes the value "1" when $(n,c) \in \mathcal{D}$, and "0" else). The goal is to design a joint strategy for resource allocation, scheduling, and routing, so that the network is stable and total average power expenditure is minimized.

This objective can be stated within our general framework by defining penalties to be the power expended on each link: $x_{ab}(t) = P_{ab}(t)$. There are no rewards, and there is only a single cost function $f(\boldsymbol{P}) = \sum_{ab} P_{ab}$. Because this function is *linear* and there are no additional cost constraints, we do not require any virtual queues for this system. Setting the virtual queue backlogs to zero, assigning $\boldsymbol{y}(t) = \boldsymbol{\gamma}(t) = \boldsymbol{0}$, and letting $\boldsymbol{\alpha}(t) = \boldsymbol{P}(t)$ in (6.11) leads to the following Energy Efficient Control Algorithm (EECA) from [116]:

Multi-hop EECA for Minimizing Average Power [116]: Every timeslot, the current $\boldsymbol{U}(t)$ backlog and the current topology state $S(t)$ is observed. Then:

(1) For all links (a,b), find the commodity $c_{ab}^*(t)$ such that:

$$c_{ab}^*(t) \triangleq \arg \max_{c \in \{1,\ldots,K\}} \left\{ U_a^{(c)}(t) - U_b^{(c)}(t) \right\},$$

and define:

$$W_{ab}^*(t) \triangleq \max[U_a^{(c_{ab}^*)}(t) - U_b^{(c_{ab}^*)}(t), 0].$$

(2) *Power Allocation:* Choose $\boldsymbol{P}(t) \in \mathcal{P}$ to maximize:

$$\sum_{ab} \left[W_{ab}^*(t) C_{ab}(\boldsymbol{P}, S(t)) - V P_{ab}(t) \right].$$

(3) *Routing:* Over each link (a, b) such that $W^*_{ab} > 0$, transmit $C_{ab}(\boldsymbol{P}(t), S(t))$ units of commodity $c^*_{ab}(t)$ data (using idle fill if necessary).

The above algorithm stabilizes the network whenever $\epsilon_{max} > 0$, and yields a total average congestion of $\overline{\sum_{n,c} U_n^{(c)}} \leq (B + VP_{max})/\epsilon_{max}$ and ensures total average power expenditure is within B/V of the minimum power required for stability. If power reception costs are also significant, the algorithm can easily be modified by augmenting the penalty functions to account for power costs expended by each receiver.

6.3.2.1 Simulation of a 2-queue downlink under EECA

Consider the special case of a 2-queue downlink with time varying channels and "Good," "Medium," and "Bad" states on each of the two links. At most one link can be activated for transmission every slot, and exactly 1 unit of power is used on each activation. Every timeslot the network controller must choose to activate either link 1, link 2, or to remain idle (in order to save power). A single packet can be transmitted when a link is in the "Bad" state, two packets can be transmitted in the "Medium" state, and three can be transmitted in the "Good" state. An example set of arrivals over the course of 9 timeslots is given in Fig. 6.1, where the choices associated with the MWM policy described in Section 4.1 are shown, along with a more energy efficient set of choices, both of which leave the system empty on timeslot $t = 9$.

The EECA algorithm in this case reduces to serving the queue with the largest value of $U_i(t)\mu_i(t) - VP$ (where $P = 1$ unit) whenever this value is positive, and remaining idle otherwise. In our simulation, packets arrived according to Poisson processes with rates $\lambda_1 = 8/9, \lambda_2 = 5/9$, which are the same as the empirical rates obtained by averaging over the first 9 timeslots of the example in Fig. 6.1. Channel states arise as i.i.d. vectors $(S_1(t), S_2(t))$ every slot. The probability of each vector state is matched to the empirical occurrence frequency in the example, so that $Pr[(G, M)] = 3/9, Pr[(M, B)] = 2/9, Pr[(M, M)] = 1/9$, etc.

We simulated the EECA algorithm for 20 different values of the control parameter V, ranging from 0 to 10^4. Each simulation was run

	t	0	1	2	3	4	5	6	7	8
Arrivals	$A_1(t)$	3	0	3	0	0	1	0	1	0
	$A_2(t)$	2	0	1	0	1	1	0	0	0
Channels	$S_1(t)$	G	G	M	M	G	G	M	M	G
	$S_2(t)$	M	M	B	M	B	M	B	G	B
MWM Policy	0	3	0	3	1	0	1	1	2	
	$U_2(t)$	0	2	2	2	2	3	2	1	0
Better Choices	$U_1(t)$	0	3	3	6	6	3	1	1	2
	$U_2(t)$	0	2	2	3	1	2	3	3	0

Fig. 6.1 An example set of arrivals, channel conditions, and queue backlogs for a two queue wireless downlink under two different scheduling algorithms, illustrating the power efficiency gains enabled by having full knowledge of future arrivals and channel states.

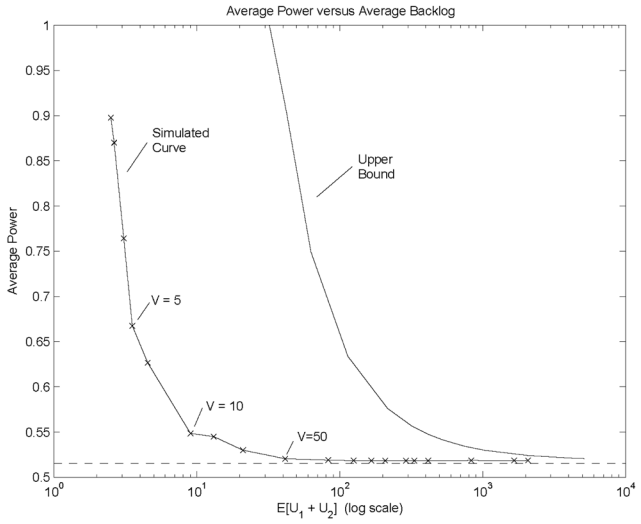

Fig. 6.2 Average power versus average backlog for a two queue downlink under the EECA algorithm (from [116]).

for 10 million timeslots. In Fig. 6.2 the resulting average power is plotted against the time average backlog. The corresponding upper bound derived in [116] is also shown in the figure. We find that average power is equal to 0.898 Watts and average sum backlog is 2.50 packets when $V = 0$ (corresponding to the original MWM policy). Average

power decreases to its minimum value of 0.518 Watts as the control parameter V is increased, with a corresponding tradeoff in average delay. As a point of reference, we note that at $V = 50$, the average power is 0.53 Watts and the average sum backlog in the system is 21.0 packets.

6.3.2.2 Minimum energy scheduling for mobile networks

Here we consider an ad-hoc mobile network with 28 users and a cell structure arranged as a 4×4 grid, as shown in Fig. 6.3. For simplicity, we assume there can be at most one transmission per cell per timeslot, and that all transmissions use full power of 1 Watt. We assume transmission rates are adaptive, and that 3 packets can be transferred if the receiver is in the same cell as the transmitter, while only 1 packet can be transferred if the receiver is in one of the adjacent cells to the North, South, East, or West. Data arrives to each node according to a Bernoulli arrival process with rate $\lambda = 0.5$ packets/slot (so that a single packet arrives with probability 0.5, else no packet arrives). We assume source-destination pairs are given by the grouping $1 \leftrightarrow 2$, $3 \leftrightarrow 4$, ..., $27 \leftrightarrow 28$, so that node 1 packets are destined for node 2 and node 2

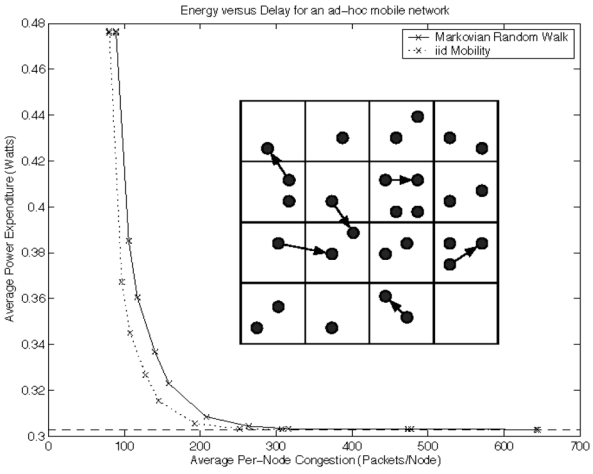

Fig. 6.3 An ad-hoc mobile network with adaptive transmission rates, and the resulting per node average power expenditure versus average node congestion for V between 0 and 200.

packets are destined for node 1, node 3 packets are destined for node 4 and node 4 packets are destined for node 3, etc.

We simulate the multi-hop EECA algorithm for both a Markovian random walk model and an i.i.d. mobility model, with the objective of minimizing total power expenditure. In the Markovian mobility model, every timeslot nodes independently move to a neighboring cell either to the North, South, East, or West, with equal probability. In the case when a node on the edge of the network attempts to move in an infeasible direction, it simply stays in its current cell. In the i.i.d. mobility model, nodes randomly choose new cell locations every timeslot independently and uniformly over the set of all 16 cells. It is not difficult to show that both mobility models have the same steady state node location distribution. Hence, the network capacity region and the minimum average power required for stability are exactly the same for both mobility models [115, 116]. In this case, the minimum average power for stability under the given traffic load can be exactly computed, and is equal to 0.303 Watts [160].

Simulations were conducted using control parameters V in the range from 0 to 200, and the results are given in Fig. 6.3. In the figure, each data point represents an independent simulation for a particular value of V over the course of 4 million timeslots. The resulting per-node average power is plotted against the resulting per-node average queue congestion. From the figure, it is clear that under both mobility models, average power expenditure quickly converges to the minimum power level as the control parameter V is increased (and hence, delay is increased). The average delay under Markovian mobility is slightly larger than the delay under i.i.d. mobility. As an example set of data points, we note that for the Markovian mobility model at $V = 0$ (corresponding to the Dynamic Backpressure strategy of Section 4), the per-node average backlog is 89.2 packets (about 3.3 packets on average in each of the 27 internal queues), and per-node average power expenditure is 0.477 Watts. At $V = 40$, the per-node average backlog is 263.6 packets, and per-node average power expenditure is 0.305 Watts. For values of V beyond 50, average power expenditure differs from the optimal value of 0.303 only in the fourth or fifth significant digits, while average congestion continues to increase.

6.3.3 Notes

- Related work by Liu, Chong, and Shroff in [95] considers maximizing a time average utility function for an infinitely backlogged downlink in the case when exactly one user must transmit on each and every timeslot. The technique can be used to address the minimum average power problem if all transmissions consist of single packets and "utility" is measured by a negative power cost associated with transmitting a packet in the current channel state. In this case, the algorithm of [95] chooses to transmit over the link i that maximizes $v_i^* - P_i$, where $\{v_i^*\}_{i=1}^L$ are pre-computed indices determined by the system constraints and the channel state probabilities. The EECA algorithm applied to this setting would choose the link i that maximizes the index $U_i(t)/V - P_i$, so that the pre-computed index v_i^* is replaced by a time varying index $U_i(t)/V$ that does not require any pre-computation or prior knowledge of channel probabilities. Note also that performance can be improved if the system is allowed to be idle during some timeslots, in which case EECA would enter idle mode whenever $U_i(t)/V - P_i$ is non-positive for all links i.
- The EECA and ECCA algorithms presented in this section generalize several related results. Work in [155] describes the information theoretic capacity region for multiple access fading channels, and a dynamic strategy is given there for the case when all users are infinitely backlogged and all channel probabilities are a-priori known. This can be viewed as a special case of our framework, where rate-power curves $\boldsymbol{C}(\boldsymbol{P}, S)$ are determined by an optimal receiver algorithm that uses successive noise cancellation every slot. Broadcast capacities for power limited downlinks are developed in [64, 96] assuming infinite backlog and known channel probabilities, and a capacity achieving queueing strategy with average energy constraints is developed in [168] under the assumption that channel probabilities are known.

- Average power expenditure in an ad-hoc network with no channel or topology state variation is considered in [20, 36, 77]. In [36], a periodic scheduling framework is developed for minimizing energy expenditure subject to fixed time average rate requirements on each link. The solution is based on an off-line computation of an optimization problem. The work in [77] considers complexity issues for a similar problem, and the work in [20] treats a network with simpler interference constraints and provides an algorithm for computing schedules that are within a constant factor of the minimum energy solution. The EECA algorithm provides an on-line solution strategy for all of these problems, which can also be used as an off-line computation method for the scenarios of [20, 36, 77].
- The $[O(1/V), O(V)]$ energy-delay tradeoff achieved by the EECA algorithm is not the optimal tradeoff. Work by Berry and Gallager in [15] considers the energy-delay tradeoff for a single wireless link with a random arrival process, a fading channel, and a concave rate-power curve. The fundamental tradeoff law is shown to have a square-root structure with energy-delay parameters $[O(1/V), O(\sqrt{V})]$. This result is recently extended to multi-user networks in [119]. An improved tradeoff $[O(1/V), O(\log(V))]$ is achievable in the exceptional cases when the system has a piecewise linear structure [119] or when a small fraction of packets can be dropped [117]. Related notions of *delay limited capacities* are developed in [154] for the case when tolerable delay is no more than one timeslot.
- Energy and delay issues for static wireless links are considered in [30, 161, 170], and a problem with a static link but stochastic arrivals is treated using filter theory in [73]. Similar problems of minimizing energy subject to delay constraints or minimizing delay subject to energy constraints are treated for single link satellite and wireless systems in [49, 50, 56, 170] using stochastic differential equations, dynamic programming, and Markov decision theory, and multi-link systems

are considered in [141]. Related problems for wireless sensor networks are considered in [37, 169], and delay efficient sleep scheduling is considered in [99]. Asymptotic energy results as network size is scaled to infinity are considered in [89].

6.4 A related algorithm

In this section we describe another approach for the solution to the general optimization problem (6.1), proposed by Stolyar [136], under modeling assumptions similar to those described above. We describe next the assumptions in [136]. For consistency we employ the notation used in the current text.

(1) The network model and the set of penalties x_m, $m \in \mathcal{M}$ are the same as those described in Section 6.1.1.
(2) The network topology state $S(t)$ constitutes an irreducible finite Markov Chain with state space $\mathcal{S}$.
(3) For each state $s \in \mathcal{S}$, the set of available controls $\mathcal{I}_s$ is finite.
(4) When $S(t) = s$ and a control $I \in \mathcal{I}_s$ is chosen, the following occurs.
    (a) A penalty $x_m(I)$ is generated for all $m \in \mathcal{M}$.
    (b) An amount $R_n(I)$ of exogenous traffic is admitted to enter node $i \in \mathcal{N}$.
    (c) Each node $i \in \mathcal{N}$ selects up to $\mu_i(I)$ units of traffic from its queue (if there are fewer units than $\mu_i(I)$, the whole queue is emptied). Each of these units is routed to the rest of the nodes in the network with probabilities $p_{ij}(I)$, $j \in \mathcal{N}$.
    (d) When a traffic unit reaches its destination, it is removed from the network.

The penalties $x_m(I)$, the admitted traffic $R_n(I)$, and the traffic transmitted by each node $\mu_i(I)$ may also be random variables that depend on the control chosen, but are independent of anything else. The routing probabilities $p_{ik}(I)$ may depend on the control but are also

independent of anything else. Multicommodity flows can be taken care of by considering multiple queues at each node, one for each commodity along the lines described in earlier sections.

The Markovian assumption on the $S(t)$ topology state is similar in both our general framework and the framework of [136]. The assumption that the available control sets $\mathcal{I}_s$ ($s \in \mathcal{S}$) are finite does not include the case where controls may take continuous values, as for example, when the control refers to transmission power of a node which may be continuously varied. The assumption that $R_n(I)$ depends only on the control chosen excludes the case of using a reservoir for holding traffic that has not been admitted to the network. These assumptions are not very restrictive and they can probably be relaxed without affecting the optimality of the proposed algorithm to be described below. The assumption imposed on routing, i.e., the use of routing probabilities, may be more restrictive as it does not seem easy to include in a simple manner the practical case where a deterministic part of the traffic $\mu_i(I)$ selected by node i is routed to node j, rather than a random amount with average $\mu_{ij}(I) = \mu_i(I)p_{ij}(I)$. However in several specific problems, if one formally uses the average amount as the amounts actually routed, the algorithm seems to work. Therefore, it seems that this assumption is mainly needed for the proofs in [136] and may also be relaxed without affecting the optimality of the algorithm. Further, in the special case when each node is restricted to sending over a single outgoing link, the probabilistic routing can be designed to choose the desired link with probability 1.

The system is slotted. The proposed algorithm performs the following at the beginning of each slot.

(1) The weighted averages of penalties are measured,

$$\overline{x}_m(t) = (1-\beta)\overline{x}_m(t-1) + \beta x_m(I(t-1)), \quad m \in \mathcal{M}, \tag{6.15}$$

where $0 < \beta < 1$ is a typically small number. The smaller β is, the closer the performance of the algorithm approaches the optimum, at the expense of less adaptivity of the algorithm in case of parameter changes.

(2) The queue sizes, $U_n(t)$, $n \in \mathcal{N}$ are measured and the network state $S(t)$ is observed. The control to be employed is obtained as the solution

of the following optimization problem.

$$\text{Max: } \beta \sum_{i,j} (U_i(t) - U_j(t))\,\mu_{ij}(I) - \beta \sum_i U_i(t)\,R_n(I) \qquad (6.16)$$
$$- \sum_{m \in \mathcal{M}} (\vartheta f(\overline{x}_m(t))/\vartheta \overline{x}_m(t))\,x_m(I),$$

subject to: $I \in \mathcal{I}_{S(t)}$.

Here we are assuming that the algorithm transmits a deterministic amount of bits $\mu_{ij}(I)$ over each link (i,j), rather than the probabilistic amount formally used in the framework of [136].

There are similarities as well as differences between this algorithm and the algorithm presented in Section 6.2.1. These are best explained by an example which we discuss below.

Example 6.1. Consider a wireless network with L transmission links denoted by a link set $\mathcal{L}$. The transmission rate for each link l depends on a power allocation vector according to the function $C_l(\boldsymbol{P}(t), S(t))$. Suppose that power must satisfy $\boldsymbol{P}(t) \in \mathcal{P}$ for all t, where $\mathcal{P}$ is a finite collection of acceptable power allocation vectors. We consider single-hop transmissions where exogenous arrivals $A_l(t)$ arrive to the network at the source node of link l and must be transmitted over link l. The exogenous arrival vectors $(A_l(t))_{l \in \mathcal{L}}$ are i.i.d. over timeslots, and we assume that $A_l(t) \leq A^{\max}$. At each timeslot, $R_l(t)$ bits from this exogenous traffic may enter the network, where $R_l(t) \leq A_l(t)$. Traffic that does not enter the network is dropped, i.e., we assume zero reservoir space. The arrival rate vector may take arbitrary values, and in particular may lie outside the capacity region of the system. The goal is to design a joint flow control and resource allocation algorithm that stabilizes the network and yields an optimal throughput utility $\sum_{l \in \mathcal{L}} g_l(\overline{r}_l)$, where $g_l(r)$ denotes a concave utility function of the throughput over link l, and $\overline{r}_l$ represents the long term average admitted rate into link l (assuming for now such time averages exist). In the following we will refer to the GCLC algorithm presented in Section 6.2.1 as ALG1, and to the Algorithm presented in [136] as ALG2. According to the description in Section 6.2.1, ALG1 consist of the following operations.

Description of ALG1: In this example, the GCLC algorithm uses rewards $\boldsymbol{y}(t) = \boldsymbol{R}(t)$ (where $0 \leq R_l(t) \leq A^{max}$), with utility function $g(\boldsymbol{R}) = \sum_{l \in \mathcal{L}} g_l(R_l)$. There are no utility constraints, cost constraints, or penalties, and so queues $D_h(t)$, $D_q(t)$, $Z_x(t)$ are not required. As the utility function $g(\boldsymbol{R})$ is non-linear and concave, the auxiliary variables and flow state queues $\boldsymbol{\gamma}(t)$ and $\boldsymbol{Z}_y(t)$ are used. For notational simplicity, we drop the subscript on queue $\boldsymbol{Z}_y(t)$ and use instead $\boldsymbol{Z}(t)$. The initial condition is $\boldsymbol{Z}(0) = \boldsymbol{0}$, and the queue update equation (6.8) in this example is given by:

$$Z(t+1) = \max[Z(t) - \boldsymbol{R}(t), 0] + \boldsymbol{\gamma}(t) \tag{6.17}$$

This algorithm thus reduces to the CLC2b algorithm of Section 5. In particular, every slot the flow control variables $R_l(t)$ are chosen such that $R_l(t) = A_l(t)$ whenever $U_l(t) \leq Z_l(t)$, and $R_l(t) = 0$ else. The values $\gamma_l(t)$ are then computed as the solution to the following optimization:

$$\begin{aligned} &\text{Minimize: } -Vg_l(\gamma_l) + \gamma_l Z_l(t) \\ &\text{Subject to: } 0 \leq \gamma_l \leq A^{max}. \end{aligned} \tag{6.18}$$

The virtual queue $Z(t)$ is then updated according to (6.17). The resource allocation strategy is then the same as the Dynamic Backpressure strategy of Section 4.3, where $\boldsymbol{P}(t) \in \mathcal{P}$ is chosen to maximize $\sum_{l \in \mathcal{L}} U_l(t) C_l(\boldsymbol{P}(t), S(t))$.

Description of ALG2: To bring the problem into a cost-minimization form we define penalty variables $x_l(t) = A_{\max} - R_l(t)$ and consider the equivalent problem of minimizing

$$\sum_{l \in \mathcal{L}} f_l(\overline{x}_l), \tag{6.19}$$

where

$$f_l(x) = g_l(A^{\max}) - g_l(A^{\max} - x). \tag{6.20}$$

Hence the "penalties" are now $(x_l(t))_{l \in \mathcal{L}}$. Since $x_l(t) \leq A^{\max}$ we may set $X_l^{\max} = A^{\max}$, $l \in \mathcal{L}$. With the zero-reservoir assumption, and the condition that $A_l(t)$ takes finite values, we may consider that the vector

$(A_l(t))_{l\in\mathcal{L}}$ is part of network state $S(t)$ and that $R_l(t)$ is part of the exercised control $\widehat{\mathcal{I}}_{\widehat{s}(t)}$ (interpreted as a control vector augmented with the flow control choice and with a state that includes the new arrivals). The ALG2 then keeps track of the weighted averages:

$$\overline{x}_l(t+1) = (1-\beta)\overline{x}_l(t) + \beta\left(A^{\max} - R_l(t)\right),\ l \in \mathcal{L}, \tag{6.21}$$

Taking into account the form of the function (6.19), the optimization problem (6.16) for determining the optimal control $\widehat{\mathcal{I}}_{\widehat{s}(t)}$ becomes in this case:

$$\begin{aligned}\text{Max: } &\beta\sum_{l\in\mathcal{L}} U_l(t)C_l(\boldsymbol{P}(t),S(t)) - \beta\sum_{l\in\mathcal{L}} U_l(t)\,r_l \\ &-\sum_{l\in\mathcal{L}} f_l'\left(\overline{x}_l(t)\right)\left(A^{\max} - r_l\right), \\ &0 \le R_l \le A_l(t),\ l\in\mathcal{L}, \quad \boldsymbol{P}(t)\in\mathcal{P},\end{aligned} \tag{6.22}$$

where $f_l'(x)$ denotes the derivative of $f_l(x)$. Observing the structure of this problem it can be seen that it is decomposable into the following subproblems.

Flow Control: The accepted traffic on link l, $R_l(t)$, is given by the solution to the following optimization problem,

$$\begin{aligned}\text{Minimize: } &\beta\sum_{l\in\mathcal{L}} U_l(t)r_l - \sum_{l\in\mathcal{L}} f_l'\left(\overline{x}_l(t)\right)r_l, \\ &0 \le r_l \le A_l(t), \quad l\in\mathcal{L}.\end{aligned}$$

The solution to this problem is simply to accept all traffic $A_l(t)$ if $U_l(t) < f_l'\left(\overline{x}_l(t)\right)/\beta$ and reject it otherwise. Using the computed values of $(R_l(t))_{l\in\mathcal{L}}$, the auxiliary queues are updated according to (6.21).

Resource Allocation: Same as in ALG1.

The main difference between the two algorithms is that ALG1 uses the auxiliary queue updates (6.17), while ALG2 uses the weighted averages updates (6.21). It is remarkable that both algorithms introduce additional state variables to solve the non-linear optimization. The updates performed by either algorithm have an effect on the adaptivity of the system when channel state or arrival statistics change. It would be interesting to assess the delay properties of both algorithms,

and the adaptability capabilities of the two algorithms when statistical parameters change. The ALG2 requires utility functions to have computable derivatives in order to implement the flow control algorithm. This is not a major assumption, as most utilities of interest are simple differentiable functions. The ALG1 does not require the existence of a derivative, but requires the solution of problem (6.18) to implement the flow control. In this case, the problem is a simple concave maximization over a single variable, and so the solution is either one of the two endpoints, or a local maximum in the interior. In the case when a derivative exists, any such local maximum would satisfy $g_l'(\gamma_l) = Z_l(t)/V$, which can be solved in closed form if the inverse of the derivative is known. In both cases, the flow control algorithms are quite simple. The main computational complexity is in the solution to the Link Power Control problem in an efficient and preferably distributed manner.

7

Final Remarks

The cross-layer resource allocation model presented in this text was motivated by wireless communication networks. Nevertheless it has within its scope a number of other application areas and it can be extended in various ways. Some are outlined below.

High speed switches with input queueing is an area that attracted a lot of attention [40, 72, 86, 88, 102, 103]. Packet scheduling in such a switch falls within the scope of our resource allocation model. The switch architecture imposes restrictions of a single packet transmission per slot for each input and output port, a restriction that is captured by the transmission conflict constraints of the resource allocation model. The capacity region characterization as well as the optimal control policies results apply in that case. Different applications pose different requirements on the scheduling policy regarding tolerable complexity and distributed versus centralized implementations. In the case of the switch for instance, the computational burden of scheduling is severely constrained by the time for packet transmission and the latter is shrinking as the bandwidth increases. Another approach to deal with high scheduling complexity is to resort to randomized scheduling policies [54, 126, 150]. In these polices a randomized algorithm computes the access schedule at each time and it updates the one used previously only

if it is better. The randomized algorithm being of low computational complexity simplifies the computational requirements, without sacrificing any throughput but only with some increase of the delay. When the network is geographically distributed, collecting state information for the access controller might be cumbersome and might result in outdated information available to the controller. Recently there have been several efforts to identify scheduling algorithms with low computational complexity and amenable to distributed implementation. The impact of imperfect scheduling on the throughput in cases where suboptimal algorithms perform the scheduling is studied in [29, 45, 91] while fully distributed scheduling policies are given as well.

Resource allocation problems in manufacturing and transportation that have been considered recently fall within the scope of the model we consider here as well. An extension of the back pressure policy that is applicable in systems with random service times and non-preemptive service is presented in [142]. The maximum pressure policy proposed in [38, 39] follows similar principles with the adaptive back pressure policy while it was shown to possess certain optimality properties. Other service provisioning structures that fall within the scope of the presented model have been considered in [9, 13, 14]. The analysis of the system in the latter cases was done under general stationary ergodic assumptions about the statistics.

Various load balancing and routing problems studied by the theoretical computer science community fall within the scope of the model we consider here, while various policies proposed in that context rely on the differential backlog rule for traffic forwarding. A policy similar to adaptive back pressure policy was proposed and studied in [2, 8, 11], in the context of adversarial queueing theory. That is, its performance was analyzed under arrival traffic patterns that might be the worst possible within a certain family of arrival patterns, for instance all possible arrival patterns at the output of a traffic regulator that produces traffic constrained bursts. It was shown that the policy achieves maximum throughput in that context as well.

The problem of paging a mobile user may be cast in the context of our model and the adaptive back pressure strategy provides optimal paging policies [6, 7].

Multicasting can be viewed as a generalization of unicast information transport, where the information generated at source node s_l needs to be transported to all nodes of the set of destinations D_l. The capacity and maximum throughput results presented in the text can be extended to the case of multicasting as is reported in [131, 132]. Information flows in multicasting are identified by the source node and the group of receivers and typically there are several of them flowing through the network $(s_1, D_1), .., (s_L, D_L)$. A multicast information flow might be served by an eligible multicast tree, i.e. a directed tree rooted at s_l and including in its leaf or intermediate nodes, all nodes of D_l. The information flows from the source to the destination nodes through the unique paths designated by the multicast tree. In that sense information transport in multicasting resembles more virtual circuit forwarding rather than datagram. The traffic of an information flow might be split across all multicast trees eligible for the specific flow that are designated to carry traffic from that flow. A vector of the traffic loads of each information flow is feasible, if there is a way to split the load of each information flow across the eligible trees for the flow such that the aggregate load of the traffic of all trees results in traffic loads for the links that do not exceed their capacities. The transport capacity region is defined to include all feasible traffic load vectors as above. A maximum throughput policy for the multicast case can be obtained by a variation of the adaptive back pressure policy as follows. Transmission priorities among the different trees crossing a certain link are given according to the sizes of the differential backlogs, where the differential backlog of a tree in a certain link is the difference of the upstream node backlog minus the maximum of the backlogs at each one of the links where the tree branches out at the downstream node. The above rule, in combination with a load balancing rule for allocating the traffic of an information flow to the various eligible trees that may support it, gives a maximum throughput transmission control policy.

In the above discussion about multicasting the information of the different flows after it enters the network remains intact until it reaches the destinations. An alternative that promises capacity gains as well as significant simplification of the mechanisms involved for multicasting in particular, is to allow for combining of the information arriving

at intermediate network nodes through linear or nonlinear operations. That approach, referred to as network coding [1], has been intensely investigated recently. It is possible to use an approach along the lines of the adaptive back pressure control in order to achieve maximum throughput in the case where network coding of information corresponding to the same multicast session is allowed. That approach is pursued in [62] where it is shown how the capacity region including network coding can be achieved.

Acknowledgements

This work was supported by the European networks of excellence NEWCOMM and EURONGI, the ARO grant W911NF-04-1-0306, the Greek Secretariat of Research and Development grant PENED, and by the USA National Science Foundation grant OCE 0520324.

References

[1] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Transactions on Information Theory*, vol. 46, pp. 1204–1216, July 2000.

[2] W. Aiello, E. Kushilevitz, R. Ostrovsky, and A. Rosen, "Adaptive packet routing for bursty adversarial traffic," in *Proceedings of ACM Symposium on Theory of Computing*, 1998.

[3] T. E. Anderson, S. S. Owicki, J. B. Saxe, and C. P. Thacker, "High speed switch scheduling for local area networks," *ACM Transactions on Computer Systems*, vol. 11, pp. 319–352, November 1993.

[4] M. Andrews, K. Kumaran, K. Ramanan, A. Stolyar, and P. Whiting, "Providing quality of service over a shared wireless link," *IEEE Communications Magazine*, vol. 39, no. 2, pp. 150–154, 2001.

[5] M. Andrews, "Maximizing profit in overloaded networks," in *Proceedings of IEEE INFOCOM*, March 2005.

[6] F. Anjum, M. Shayman, and L. Tassiulas, "On maximum throughput paging policies in wireless networks," in *Proceedings of 15th International Teletraffic Congress*, June 1997.

[7] F. Anjum, L. Tassiulas, and M. Shayman, "Optimal paging for mobile location tracking," *Advances in Performance Analysis*, vol. 3, pp. 153–178, 2002.

[8] E. Anshelevich, D. Kempe, and J. Kleinberg, "Stability of load balancing algorithms in dynamic adversarial systems," in *Proceedings of ACM Symposium on Theory of Computing*, 2002.

[9] M. Armony and N. Bambos, "Queueing dynamics and maximal throughput scheduling in switched processing systems," *Queuing Systems*, vol. 44, pp. 209–252, July 2003.

[10] S. Asmussen, *Applied Probability and Queues*. New York: Spring-Verlag, Second ed., 2003.

[11] B. Awerbuch, P. Berenbrink, A. Brinkmann, and C. Scheideler, "Simple routing strategies for adversarial systems," in *Proceedings of IEEE Symposium on Foundations of Computer Science*, 2001.

[12] F. Baccelli and P. Bremaud, *Elements of Queueing Theory*. Berlin: Springer, 2nd ed., 2003.

[13] N. Bambos and G. Michailidis, "On parallel queuing with random server connectivity and routing constraints," *Probability in the Engineering and Information Sciences*, vol. 16, pp. 185–203, 2002.

[14] N. Bambos and G. Michailidis, "Queueing and scheduling in random environments," *Adv. Applied Prob*, vol. 36, pp. 293–317, 2004.

[15] R. Berry and R. Gallager, "Communication over fading channels with delay constraints," *IEEE Transactions on Information Theory*, vol. 48, pp. 1135–1149, May 2002.

[16] R. Berry, P. Liu, and M. Honig, "Design and analysis of downlink utility-based schedulers," in *Proceedings of the 40th Allerton Conference on Communication, Control and Computing*, October 2002.

[17] D. P. Bertsekas, A. Nedic, and A. E. Ozdaglar, *Convex Analysis and Optimization*. Boston: Athena Scientific, 2003.

[18] D. Bertsekas and R. Gallager, *Data Networks*. Englewood Cliffs, New Jersey 07632, Prentice Hall, 1992.

[19] D. Bertsekas, *Nonlinear Programming*. Belmont, Massachusetts, Athena Scientific, 1995.

[20] R. Bhatia and M. Kodialam, "On power efficient communication over multihop wireless networks: Joint routing, scheduling and power control," in *Proceedings of IEEE INFOCOM*, March 2004.

[21] P. P. Bhattacharya, L. Georgiadis, P. Tsoucas, and I. Viniotis, "Adaptive lexicographic optimization in multi-class M/GI/1 queues," *Mathematics of Operations Research*, vol. 18, no. 3, pp. 705–740, 1993.

[22] P. P. Bhattacharya, L. Georgiadis, and P. Tsoucas, "Problems of adaptive optimization in multiclass M/GI/1 queues with bernoulli feedback," *Mathematics of Operations Research*, vol. 20, pp. 356–380, May 1995.

[23] E. Biglieri, Proakis, and S. Shamai, "Fading channels: Information-theoretic and communications aspects," *IEEE Transactions on Information Theory*, vol. 44(6), Oct. 1998.

[24] S. Borst, "User-level performance of channel-aware scheduling algorithms in wireless data networks," *IEEE/ACM Transactions on Networking*, vol. 13, pp. 636–647, June 2005.

[25] J. Broch, D. A. Maltz, D. B. Johnson, Y. Hu, and J. Jetcheva, "A performance comparison of multi-hop wireless ad hoc network routing protocols," in *Proceedings of Mobile Computing and Networking*, pp. 85–97, 1998.

[26] L. Bui, E. Eryilmaz, R. Srikant, and X. Wu, "Joint asynchronous congestion control and distributed scheduling for multi-hop wireless networks," in *Proceedings of IEEE INFOCOM*, April 2006.

[27] C. S. Chang, *Performance Guarantees in Communication Networks*. New York: Spinger-Verlag, 2000.

[28] J. H. Chang, L. Tassiulas, and F. Farrokhi, "Beamforming for maximum capacity in wireless networks with transmitter and receiver antenna arrays," *IEEE Transactions on Wireless Communications*, vol. 1, pp. 16–27, January 2002.

[29] P. Chaporkar, K. Kar, and S. Sarkar, "Throughput guarantees through maximal scheduling in wireless networks," in *Proceedings of 43rd Annual Allerton Conference on Communication Control and Computing*, Allerton, Monticello, IL, September 2005.

[30] W. Chen and U. Mitra, "Delay-constrained energy-efficient packet transmissions," in *Proceedings of IEEE INFOCOM*, April 2006.

[31] M. Chiang, "To layer or not to layer: Balancing transport and physical layers in wireless multihop networks," in *Proceedings of IEEE INFOCOM*, March 2004.

[32] J. P. Choi and V. W. S. Chan, "Predicting and adapting satellite channels with weather-induced impairments," *IEEE Transactions on Aerospace and Electronics Systems*, July 2002.

[33] J. P. Choi, "Channel prediction and adaptation over satellite channels with weather-induced impairments". Master's thesis, Massachusets Institute of Technology, Cambridge, MA, 2000.

[34] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. New York: John Wiley & Sons, Inc., 1991.

[35] R. L. Cruz, "A calculus for network delay. i. network elements in isolation," *IEEE Transactions on Information Theory*, pp. 114–131, 37:1, 1991.

[36] R. Cruz and A. Santhanam, "Optimal routing, link scheduling and power control in multi-hop wireless networks," in *Proceedings of IEEE INFOCOM*, March 2003.

[37] S. Cui, R. Madan, A. J. Goldsmith, and S. Lall, "Energy-delay tradeoff for data collection in TDMA-based sensor networks," in *Proceedings of IEEE ICC*, May 2005.

[38] G. Dai, W. Lin, R. Moorthy, and C. P. Teo, "Berth allocation planning optimization in container terminals,". preprint.

[39] G. Dai and W. Lin, "Maximum pressure policies in stochastic processing networks," *Operations Research*, vol. 53, pp. 197–218, March-April 2005.

[40] J. G. Dai and B. Prabhakar, "The throughput of data switches with and without speedup," in *Proceedings of IEEE INFOCOM*, March 2000.

[41] W. J. Dally and B. P. Towles, *Principles and Practices of Interconnection Networks*. Elsevier, Inc., 2004.

[42] R. Elbatt and A. Ephremides, "Joint scheduling and power control for wireless ad-hoc networks," in *Proceedings of IEEE INFOCOM*, June 2002.

[43] M. El-Taha and S. Stidham Jr., *Sample-Path Analysis of Queueing Systems*. Boston: Kluwer Academic Publishers, 1999.

[44] Y. Ermoliev, "Stochastic quasigradient methods and their application to system optimization," *Stochastics*, vol. 9, pp. 1–36, 1983.

[45] A. Eryilmaz, R. Srikant, and J. Perkins, "Stable scheduling policies for fading wireless channels," *IEEE/ACM Transactions on Networking*, pp. 411–424, April 2005.

[46] A. Eryilmaz and R. Srikant, "Fair resource allocation in wireless networks using queue-length based scheduling and congestion control," in *Proceedings of IEEE INFOCOM*, March 2005.

[47] F. Farrokhi, K. J. R. Liu, and L. Tassiulas, "Transmit beamforming and power control for cellular wireless systems," *IEEE Journal on Selected Areas in Communications, Special issue on signal processing in wireless communications*, vol. 16, no. 8, pp. 1437–1450, 1998.

[48] F. Farrokhi, L. Tassiulas, and K. J. R. Liu, "Joint optimal power control and beamforming in wireless networks using antenna arrays," *IEEE Transactions On Communications*, vol. 46, pp. 1313–1324, October 1998.

[49] A. Fu, E. Modiano, and J. Tsitsiklis, "Optimal energy allocation and admission control for communication satellites," *IEEE Transactions on Networking*, vol. 11, pp. 448–501, June 2003.

[50] A. Fu, E. Modiano, and J. Tsitsiklis, "Optimal energy allocation for delay-constrained data transmission over a time-varying channel," in *Proceedings of IEEE INFOCOM*, March 2003.

[51] R. Gallager, "A minimum delay routing algorithm using distributed computation," *IEEE Transactions on Communications*, vol. COM-25, pp. 73–85, 1977.

[52] L. Georgiadis and L. Tassiulas, "Robust network response to overload traffic fluctuations," in *44th Allerton Conference on Communication, Control and Computing*, Illinois, USA, September 2005.

[53] L. Georgiadis and L. Tassiulas, "Optimal overload response in sensor networks," *IEEE Transactions on Information Theory, to be published*.

[54] P. Giaccone, D. Shah, and B. Prabhakar, "Randomized scheduling algorithms for high-aggregate bandwidth switches," *IEEE Journal on Selected Areas in Communications*, vol. 21, pp. 546–559, May 2003.

[55] A. Goldsmith, *Wireless Communications*. Cambridge University Press, 2005.

[56] V. Goyal, A. Kumar, and V. Sharma, "Power constrained and delay optimal policies for scheduling transmission over a fading channel," in *Proceedings of IEEE INFOCOM*, March 2003.

[57] M. Grossglauser and D. Tse, "Mobility increases the capacity of ad-hoc wireless networks," *IEEE/ACM Transactions on Networking*, vol. 48, pp. 477–486, August 2002.

[58] F. M. Guillemin and R. R. Mazumdar, "On pathwise analysis and existence of empirical distributions for G/G/1 queues," *Stochastic Processes and their Applications*, vol. 67, pp. 55–67, April 1997.

[59] P. Gupta and P. R. Kumar, "The capacity of wireless networks," *IEEE Transactions on Information Theory*, vol. 46, pp. 388–404, March 2000.

[60] P. Gupta and P. R. Kumar, "Internets in the sky: The capacity of three dimensional wireless networks," *Communications in Information and Systems*, vol. 1, pp. 33–49, January 2001.

[61] B. Hajek and G. Sasaki, "Link scheduling in polynomial time," *IEEE Transactions on Information Theory*, vol. 34, pp. 910–917, September 1998.

[62] T. Ho and H. Viswanathan, "Dynamic algorithms for multicast with intra-session network coding," in *Proceedings of 43rd Allerton Conference on Comunication, Control and Computing*, September 2005.

[63] A. Jalali, R. Padovani, and R. Pankaj, "Data throughput of cdma-hdr a high efficiency data rate personal communication wireless system," in *IEEE Vehicular Technology Conference*, May 2000.

[64] N. Jindal and A. Goldsmith, "Capacity and optimal power allocation for fading broadcast channels with minimum rates," *IEEE Transactions on Information Theory*, vol. 49, Nov. 2003.

[65] R. Johari and J. N. Tsitsiklis, "Efficient loss in a network resource allocation game," *Mathematics of Operations Research*, vol. 29, August 2004.

[66] D. Julian, M. Chiang, D. O'Neil, and S. Boyd, "QoS and fairness constrained convex optimization of resource allocation for wireless cellular and ad-hoc networks," in *Proceedings of IEEE INFOCOM*, June 2002.

[67] N. Kahale and P. E. Wright, "Dynamic global packet routing in wireless networks," in *Proceedings of IEEE INFOCOM*, 1997.

[68] P. Kall and S. W. Wallace, *Stochastic Programming*. Wiley, 1994.

[69] Y. Karasawa, M. Yamada, and J. E. Alnett, "A new prediction method for tropospheric scintillation on earth-space paths," *IEEE Transactions on Antennas and Propagation*, November 1988.

[70] F. Kelly, A. Maulloo, and D. Tan, "Rate control for communication networks: Shadow prices, proportional fairness, and stability," *Journal of the Operational Research Society*, vol. 49, pp. 237–252, 1998.

[71] F. Kelly, "Charging and rate control for elastic traffic," *European Transactions on Telecommunications*, vol. 8, pp. 33–37, 1997.

[72] I. Keslassy and N. McKeown, "Analysis of scheduling algorithms that provide 100% throughput in input-queued switches," in *Proceedings of the 39th Annual Allerton Conf. on Communication, Control, and Computing*, Oct. 2001.

[73] M. A. Khojastepour and A. Sabharwal, "Delay-constrained scheduling: Power efficiency, filter design and bounds," in *Proceedings of IEEE INFOCOM*, March 2004.

[74] M. Kobayashi, G. Caire, and D. Gesbert, "Impact of multiple transmit antennas in a queued SDMA/TDMA downlink," in *Proceedings of 6th IEEE Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, June 2005.

[75] M. Kobayashi, "On the use of multiple antennas for the downlink of wireless systems," Ph.D. dissertation, Ecole Nationale Superieure des Telecommunications, Paris, 2005.

[76] U. C. Kozat, I. Koutsopoulos, and L. Tassiulas, "A framework for cross-layer design of energy-efficient communication with qos provisioning in multi-hop wireless networks," in *Proceedings of IEEE INFOCOM*, March 2004.

[77] U. C. Kozat, I. Koutsopoulos, and L. Tassiulas, "Cross-layer design and power-efficiency considerations for QoS provisioning in multi-hop wireless networks," IEEE Transactions on Wireless Communications, to appear.

[78] U. C. Kozat and L. Tassiulas, "Throughput scalability of wireless hybrid networks over a random geometric graph," *Wireless Networks (WINET) Journal*, vol. 11, no. 4, pp. 435–449, 2005.

[79] B. Krishnamachari and F. Ordonez, "Analysis of energy-efficient, fair routing in wireless sensor networks through non-linear optimization," *IEEE Vehicular Technology Conference*, Oct. 2003.

[80] P. Kumar and S. Meyn, "Stability of queueing networks and scheduling policies," *IEEE Trans. on Automatic Control*, Feb. 1995.

[81] H. J. Kushner and P. A. Whiting, "Convergence of proportional-fair sharing algorithms under general conditions," *IEEE Transactions on Wireless Communications*, vol. 3, pp. 1250–1259, July 2004.

[82] J. Y. LeBoudec and P. Thiran, *Network Calculus*. Berlin, Germany: Springer-Verlag, 2001.

[83] J.-W. Lee, R. R. Mazumdar, and N. B. Shroff, "Downlink power allocation for multi-class cdma wireless networks," in *Proceedings of IEEE INFOCOM*, June 2002.

[84] J.-W. Lee, R. R. Mazumdar, and N. B. Shroff, "Opportunistic power scheduling for dynamic multi-server wireless systems," *IEEE Transactions on Wireless Systems*, to appear.

[85] T. Leighton, F. Makedon, S. Plotkin, C. Stein, E. Tardos, and S. Tragoudas, "Fast approximation algorithms for multicommodity flow problems," *Journal of Computer and System Sciences*, vol. 50(2), pp. 228–243, April 1995.

[86] E. Leonardi, M. Mellia, M. A. Marsan, and F. Neri, "On the throughput achievable by isolated and interconnected input-queueing switches under multiclass traffic," in *Proceedings of IEEE INFOCOM*, June 2002.

[87] E. Leonardi, M. Mellia, F. Neri, and M. A. Marsan, "Bounds on average delays and queue size averages and variances in input-queued cell-based switches," in *Proceedings of IEEE INFOCOM*, April 2001.

[88] E. Leonardi, M. Mellia, F. Neri, and M. A. Marsan, "On the stability of input-queued switches with speedup," *IEEE/ACM Transactions on Networking*, vol. 9, pp. 104–118, February 2001.

[89] L. Lin, N. B. Shroff, and R. Srikant, "Asymptotically optimal power-aware routing for multihop wireless networks with renewable energy sources," in *Proceedings of IEEE INFOCOM*, March 2005.

[90] X. Lin and N. Shroff, "The fundamental capacity-delay tradeoff in large mobile ad-hoc networks," Purdue University, Tech. Rep., 2004.

[91] X. Lin and N. Shroff, "The impact of imperfect scheduling on cross-layer rate control in wireless networks," in *Proceedings of IEEE INFOCOM*, March 2005.

[92] Y.-H. Lin and R. Cruz, "Power control and scheduling for interfering links," in *Proceedings of Information Theory Workshop*, pp. 24–29, San Antonio, Texas, USA, October 2004.

[93] Y. H. Lin and R. Cruz, "Opportunistic link scheduling, power control, and routing for multi-hop wireless networks over time-varying channels," in *Proceedings of 43rd Annual Allerton Conference on Communication, Control, and Computing*, September 2005.

[94] B. Liu, Z. Liu, and D. Towsley, "On the capacity of hybrid wireless networks," in *Proceedings of IEEE INFOCOM*, March 2003.

[95] X. Liu, E. K. P. Chong, and N. B. Shroff, "A framework for opportunistic scheduling in wireless networks," *Computer Networks*, vol. 41, pp. 451–474, March 2003.

[96] L. Li and A. Goldsmith, "Capacity and optimal resource allocation for fading broadcast channels, part i: Ergodic capacity," *IEEE Transactions on Information Theory*, vol. 47, no. 3, pp. 1083–1102, 2001.

[97] S. H. Low, "A duality model of tcp and queue management algorithms," *IEEE Trans. on Networking*, vol. 11(4), August 2003.

[98] J. Luo and A. Ephremides, "On the throughput, capacity and stability regions of random multiple access," in *IEEE Transactions on Information Theory*. to be published.

[99] G. Lu, N. Sadagopan, B. Krishnamachari, and A. Goel, "Delay efficient sleep scheduling in wireless sensor networks," in *Proceedings of IEEE INFOCOM*, March 2005.

[100] P. Marbach and R. Berry, "Downlink resource allocation and pricing for wireless networks," in *Proceedings of IEEE INFOCOM*, June 2002.

[101] P. Marbach, "Priority service and max-min fairness," *IEEE/ACM Transactions on Networking*, vol. 11, pp. 733–746, October 2003.

[102] M. A. Marsan, P. Giaccone, E. Leonardi, and F. Neri, "On the stability of local scheduling policies in networks of packet switches with input queues," *IEEE Journal on Selected Areas in Communications*, vol. 21, pp. 642–655, May 2003.

[103] N. McKeown, A. Mekkittikul, V. Anantharam, and J. Walrand, "Achieving 100% throughput in an input-queued switch," *IEEE Transactions on Communications*, vol. 47, pp. 1260–1272, August 1999.

[104] N. McKeown, "The iSLIP scheduling algorithm for input-queued switches," *IEEE/ACM Transactions on Networking*, vol. 7, pp. 188–201, April 1999.

[105] M. Médard, "Channel uncertainty in communications," *IEEE Information Theory Society Newsletter*, vol. 53, June 2003.

[106] S. Meyn and R. Tweedie, *Markov Chains and Stochastic Stability*. NY: Springer-Verlag.

[107] B. A. Movsichoff, C. M. Lagoa, and H. Che, "Decentralized optimal traffic engineering in connectionless networks," *IEEE Journal on Selected Areas in Communications*, vol. 23, pp. 293–303, February 2005.

[108] M. J. Neely, E. Modiano, and C.-P. Li, "Fairness and optimal stochastic control for heterogeneous networks," in *Proceedings of IEEE INFOCOM*, March 2005.

[109] M. J. Neely, E. Modiano, and C. E. Rohrs, "Tradeoffs in delay guarantees and computation complexity in nxn packet switches," in *Proceedings of the Conference on Information Sciences and Systems, Princeton University*, 2002.

[110] M. J. Neely, E. Modiano, and C. E. Rohrs, "Power allocation and routing in multibeam satellites with time-varying channels," *IEEE/ACM Transactions on Networking*, vol. 11, pp. 138–152, February 2003.

[111] M. J. Neely, E. Modiano, and C. E. Rohrs, "Dynamic power allocation and routing for time varying wireless networks," *IEEE Journal on Selected Areas*

in Communications, Special Issue on Wireless Ad-hoc Networks, vol. 23, No. 1, pp. 89–103, January 2005.

[112] M. J. Neely and E. Modiano, "Improving delay in ad-hoc mobile networks via redundant packet transfers," in *Proceedings of Conference on Information Sciences and Systems*, March 2003.

[113] M. J. Neely and E. Modiano, "Logarithmic delay for nxn packet switches," *IEEE Workshop on High Performance Switching and Routing*, April 2004.

[114] M. J. Neely and E. Modiano, "Capacity and delay tradeoffs for ad-hoc mobile networks," *IEEE Transactions on Information Theory*, vol. 51, No. 6, pp. 1917–1937, June 2005.

[115] M. J. Neely, "Dynamic power allocation and routing for satellite and wireless networks with time varying channels," Ph.D. dissertation, Massachusetts Institute of Technology, LIDS, 2003.

[116] M. J. Neely, "Energy optimal control for time varying wireless networks," in *Proceedings of IEEE INFOCOM*, March 2005.

[117] M. J. Neely, "Intelligent packet dropping for optimal energy-delay tradeoffs in wireless downlinks," in *Proceedings of the 4th International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt)*, April 2006.

[118] M. J. Neely, "Optimal backpressure routing for wireless networks with multi-receiver diversity," in *Proceedings of Conference on Information Sciences and Systems*, March 2006.

[119] M. J. Neely, "Optimal energy and delay tradeoffs for multi-user wireless downlinks," in *Proceedings of IEEE INFOCOM*, April 2006.

[120] M. J. Neely, "Super-fast delay trade-offs for utility optimal fair scheduling in wireless networks," in *Proceedings of IEEE INFOCOM*, April 2006.

[121] M. J. Neely, "Super-fast convergence for utility optimal scheduling in stochastic networks," Tech. Rep. CSI-05-07-01, University of Southern California, July 2005.

[122] M. J. Neely, "Optimal energy and delay tradeoffs for multi-user wireless downlinks," Tech. Rep. CSI-05-06-01, University of Southern California, June 2005.

[123] S. Papavassiliou and L. Tassiulas, "Joint optimal channel, base station and power assignment for wireless access," *IEEE/ACM Transactions on Networking*, vol. 4, No. 6, pp. 857–872, December 1996.

[124] S. Papavassiliou and L. Tassiulas, "Improving the capacity of wireless networks through integrated channel base station and power assignment," *IEEE Transactions on Vehicular Technology*, vol. 47, No. 2, pp. 417–427, May 1998.

[125] L. L. Peterson and B. S. Davie, *Computer Networks: A Systems Approach*. Morgan Kaufman Publishers, 2000.

[126] K. Psounis and B. Prabhakar, "A randomized web-cache replacement scheme," in *Proceedings of IEEE INFOCOM*, April 2001.

[127] B. Radunovic and J.-Y. LeBoudec, "Optimal power control, scheduling and routing in UWB networks," *IEEE Journal on Selected Areas in Communications*, vol. 22, No. 7, pp. 1252–1270, September 2004.

[128] B. Radunovic and J.-Y. LeBoudec, "Rate performance objectives of multihop wireless networks," *IEEE Transactions on Mobile Computing*, vol. 3, No. 4, pp. 334–349, October-December 2004.

[129] B. Radunovic and J.-Y. LeBoudec, "Power control is not required for wireless networks in the linear regime," in *Proceedings of WoWMoM*, (Taormina, Italy), June 2005.

[130] T. Ren, R. La, and L. Tassiulas, "Optimal transmission scheduling with base station antenna array in cellular networks," in *Proceedings of IEEE INFOCOM*, March 2004.

[131] S. Sarkar and L. Tassiulas, "A framework for routing and congestion control for multicast information flows," *IEEE Transactions on Information Theory*, vol. 48, No. 10, pp. 2690–2708, October 2002.

[132] S. Sarkar and L. Tassiulas, "Back pressure based multicast scheduling for fair bandwidth allocation," *IEEE Transactions on Neural Networks, Special issue on Adaptive Learning Systems in Communication Networks*, vol. 49, No. 10, pp. 1858–1863, 2005.

[133] S. Sarkar and L. Tassiulas, "End-to-end bandwidth guarantees through fair local spectrum share in wireless ad-hoc networks," *IEEE Transactions on Automatic Control*, vol. 50, No. 9, pp. 1246–1259, September 2005.

[134] D. Shah and M. Kopikare, "Delay bounds for the approximate maximum weight matching algorithm for input queued switches," in *Proceedings of IEEE INFOCOM*, June 2002.

[135] A. L. Stolyar, "MaxWeight scheduling in a generalized switch: State space collapse and workload minimization in heavy traffic," *Annals of Applied Probability*, vol. 14, no. 1, pp. 1–53, 2004.

[136] A. L. Stolyar, "Maximizing queueing network utility subject to stability: Greedy primal-dual algorithm," *Queueing Systems*, vol. 50, pp. 401–457, 2005.

[137] A. L. Stolyar, "On the asymptotic optimality of the gradient scheduling algorithm for multi-user throughput allocation," *Operations Research*, vol. 53, No. 1, pp. 12–25, 2005.

[138] J. Sun, E. Modiano, and L. Zheng, "A novel auction algorithm for fair allocation of a wireless fading channel," in *Proceedings of Conference on Information Science and Systems*, March 2004.

[139] Y. Tamir and G. L. Frazier, "Dynamically-allocated multi-queue buffers for VLSI communication switches," *IEEE Transactions on Computers*, vol. 41, No. 6, pp. 725–737, June 1992.

[140] A. Tang, J. Wang, and S. Low, "Is fair allocation always inefficient," in *Proceedings of IEEE INFOCOM*, March 2004.

[141] A. Tarello, E. Modiano, J. Sun, and M. Zafer, "Minimum energy transmission scheduling subject to deadline constraints," in *Proceedings of IEEE WiOpt*, April 2005.

[142] L. Tassiulas and P. Bhattacharya, "Allocation of interdependent resources for maximum throughput," *Stochastic Models*, vol. 16, No. 1, 2000.

[143] L. Tassiulas and A. Ephremides, "Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop

radio networks," *IEEE Transactions on Automatic Control*, vol. 37, No. 12, pp. 1936–1949, December 1992.

[144] L. Tassiulas and A. Ephremides, "Dynamic server allocation to parallel queues with randomly varying connectivity," *IEEE Transactions on Information Theory*, vol. 39, No. 2, pp. 466–478, 1993.

[145] L. Tassiulas and S. Papavassiliou, "Optimal anticipative scheduling with asynchronous transmission opportunities," *IEEE Transactions on Automatic Control*, vol. 40, No. 12, pp. 2052–2062, December 1995.

[146] L. Tassiulas and S. Sarkar, "Maxmin fair scheduling in wireless adhoc networks," *IEEE Journal on Selected Areas in Communications*, vol. 23, No. 1, pp. 163–173, January 2005.

[147] L. Tassiulas, "Dynamic link activation scheduling in multihop radio networks with fixed or changing topology," Ph.D. dissertation, University of Maryland, College Park, 1991.

[148] L. Tassiulas, "Adaptive back-pressure congestion control based on local information," *IEEE Transactions on Automatic Control*, vol. 40, No. 2, pp. 236–250, February 1995.

[149] L. Tassiulas, "Scheduling and performance limits of networks with constantly changing topology," *IEEE Transactions on Information Theory*, vol. 43, No. 3, pp. 1067–1073, 1997.

[150] L. Tassiulas, "Linear complexity algorithms for maximum throughput in radio networks and input queued switches," in *Proceedings of IEEE INFOCOM*, April 1998.

[151] S. Toumpis and A. J. Goldsmith, "Capacity bounds for large wireless networks under fading and node mobility," in *Proceedings of 41st Allerton Conference on Communications, Control and Computing*, (Allerton IL), October 2003.

[152] L. Tsaur and D. C. Lee, "Closed-loop architecture and protocols for rapid dynamic spreading gain adaptation in cdma networks," in *Proceedings of IEEE INFOCOM*, March 2004.

[153] D. N. Tse, "Optimal power allocation over parallel broadcast channels," in *Proceedings of International Symposium on Information Theory*, June 1997.

[154] D. Tse and S. Hanly, "Multi-access fading channels: Part ii: Delay-limited capacities," *IEEE Transactions on Information Theory*, vol. 44, No. 7, pp. 2816–2831, November 1998.

[155] D. Tse and S. Hanly, "Multi-access fading channels: Part i: Polymatroid structure, optimal resource allocation and throughput capacities," *IEEE Transactions on Information Theory*, vol. 44, pp. 2796–2815, 1998.

[156] D. Tse and P. Viswanath, *Fundamentals of Wireless Communication*. Cambridge University Press, 2005.

[157] V. Tsibonis, L. Georgiadis, and L. Tassiulas, "Exploiting wireless channel state information for throughput maximization," *IEEE Transactions on Information Theory*, vol. 50, No. 11, pp. 2566–2582, November 2004.

[158] V. Tsibonis and L. Georgiadis, "An adaptive framework for addressing fairness issues in wireless networks," *Computer Communications*, vol. 28, pp. 1167–1178, 2005.

[159] V. Tsibonis and L. Georgiadis, "Optimal downlink scheduling policies for slotted wireless time-varying channels," *IEEE Transactions on Wireless Communications*, vol. 4, No. 3, pp. 1808–1817, July 2005.

[160] R. Urgaonkar and M. J. Neely, "Capacity region, minimum energy and delay for a mobile ad-hoc network," in *Proceedings of the 4th International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt)*, April 2006.

[161] E. Uysal-Biyikoglu, B. Prabhakar, and A. E. Gamal, "Energy-efficient packet transmission over a wireless link," *IEEE/ACM Transactions on Networking*, vol. 10, pp. 487–499, August 2002.

[162] T. Weller and B. Hajek, "Scheduling nonuniform traffic in a packet switching system with small propagation delay," *IEEE/ACM Transactions on Networking*, vol. 5, No. 6, pp. 558–598, 1998.

[163] X. Wu and R. Srikant, "Regulated maximal matchings: A distributed scheduling algorithm for multi-hop wireless networks with node exclusive spectrum sharing," in *IEEE Conference on Decision and Control*, 2005.

[164] X. Wu and R. Srikant, "Bounds on the capacity region of multi-hop wireless networks under distributed greedy scheduling," in *Proceedings of IEEE INFOCOM*, April 2006.

[165] L. Xiao, M. Johansson, and S. Boyd, "Simultaneous routing and resource allocation for wireless networks," in *Proceedings of 39th Annual Allerton Conference on Communications, Control and Computing*, October 2001.

[166] L.-L. Xie and P. R. Kumar, "A network information theory for wireless communication: Scaling laws and optimal operation," *IEEE Transactions on Information Theory*, vol. 50, No. 5, pp. 784–767, May 2004.

[167] R. Yates, "A framework for uplink power control in cellular radio systems," *IEEE Journal of Selected Areas in Communications*, vol. 13, No. 7, pp. 1341–1448, September 1995.

[168] E. Yeh and A. Cohen, "Throughput optimal power and rate control for queued multiaccess and broadband communications," in *Proceedings of International Symposium on Information Theory*, 2004.

[169] Y. Yu, B. Krishnamachari, and V. K. Prasanna, "Energy-latency tradeoffs for data gathering in wireless sensor networks," in *Proceedings of IEEE INFOCOM*, March 2004.

[170] M. Zafer and E. Modiano, "A calculus approach to minimum energy transmission policies with quality of service guarantees," in *Proceedings of IEEE INFOCOM*, March 2005.

[171] J. Zander, "Distributed co-channel interference control in cellular radio systems," *IEEE Transactions on Vehicular Technology*, vol. 41, No. 1, February 1992.