Hardness of low delay network scheduling*

Devavrat Shah David N. C. Tse† John N. Tsitsiklis‡

August 8, 2009

Abstract

We consider a communication network and study the problem of designing a high-throughput and low-delay scheduling policy that only requires a polynomial amount of computation at each time step. The well-known maximum weight scheduling policy, proposed by Tassiulas and Ephremides (1992), has favorable performance in terms of throughput and delay but, for general networks, it can be computationally very expensive. A related randomized policy proposed by Tassiulas (1998) provides maximal throughput with only a small amount of computation per step, but seems to induce exponentially large average delay. These considerations raise some natural questions. Is it possible to design a policy with low complexity, high throughput, and low delay for a general network? Does Tassiulas' randomized policy result in low average delay?

In this paper, we answer both of these questions negatively. We consider a wireless network operating under two alternative interference models: (a) a combinatorial model involving independent set constraints; and (b) the standard SINR (signal to interference noise ratio) model. We show that unless **NP**$\subseteq$**BPP** (or **P**$=$**NP** for the case of determistic arrivals and deterministic policies), and even if the required throughput is a very small fraction of the network's capacity, there does not exist a low-delay policy whose computation per time step scales polynomially with the number of queues. In particular, the average delay of Tassiulas' randomized algorithm must grow super-polynomially. To establish our results, we employ a clever graph transformation introduced by Lund and Yannakakis (1994).

Keywords: Scheduling, low-delay, high-throughput, hardness, independent set, SINR model.

*D. Shah and J. Tsitsiklis were supported by NSF grant CCF-0728554; D. Tse was supported by AFOSR grant FA9550-09-1-0317.

†D. Tse is with the department of EECS, UC Berkeley. Email: dtse@eecs.berkeley.edu

‡D. Shah and J. Tsitsiklis are with LIDS, MIT. Emails: {devavrat,jnt}@mit.edu

1 Introduction

Consider a collection of queues operating in discrete-time. At each time slot, a number of packets from different queues can be served, according to a *schedule* constrained to lie within a prespecified set which captures various interference and scheduling constraints. Upon obtaining service, a packet may leave the network or possibly join other queues. New packets may also arrive to each of these queues, according to an exogenous arrival process.

This general model has been used to describe various settings, including: (i) input-queued switches, a key functional component of Internet routers, where simultaneous packet transfers from ingress ports to egress ports are constrained by underlying hardware; and (ii) wireless local area networks, where packet transmissions are constrained by interference.

In all of the above examples, a scheduling algorithm, or *policy*, is employed, which at each time slot chooses the queues that are scheduled for transmission. The nature of the policy typically has a significant effect on the resulting network performance, measured in terms of throughput and delay. In addition, a policy carries out certain computations at each time step, and practical considerations call for low computational requirements.

A particular policy that applies to a broad class of constrained network scheduling problems, and which has received much attention, is the maximum-weight policy introduced by Tassiulas and Ephremides [10]. This policy assigns a weight to each candidate schedule, equal to the sum of the sizes of the queues that are selected for service by that schedule, and chooses a schedule with the largest weight. This policy is known to achieve maximal throughput [10]. Furthermore, it results in low[1] delay under "friendly" arrival traffic. However, this policy needs to find a maximum weight schedule at each time step, which can be computationally burdensome when the constrains involved are of a combinatorial nature. For example, in a wireless network with interference constraints specified in terms of independent set constraints, the problem of finding a maximum weight schedule amounts to solving a maximum weight independent set problem, which is NP-hard.

The above discussion suggests the need for alternative, low-complexity and high-throughput policies. Towards this purpose, Tassiulas [9] proposed a simple throughput-optimal randomized policy for switch scheduling that requires only a polynomial amount of computation at each time slot. This policy is easily extended to networks with general combinatorial scheduling constraints including the independent set constraint. It can even be implemented in a distributed manner, using a *gossip* mechanism [7]. However, the best known bound on the resulting average delay grows exponentially with the number of nodes.

In summary, there do exist throughput-optimal policies for arbitrary networks, with low complexity at each step. However, it is not known whether a low complexity (polynomial per time step), low-delay, and throughput-optimal policy is possible for networks with general combinatorial scheduling constraints. This question is not easy to address because the classical methods from stochastic network or queueing theory do not provide tools that can capture issues of computational complexity.

In this paper, we provide answers to the above question, by combining a very simple queueing theoretic analysis with tools from computational complexity theory. Specifically, we establish that for certain network models, any policy with favorable delay properties and modest throughput requirements must implicitly solve a computationally hard problem. This implies that under certain widely believed computational hypotheses, any such policy will have high computational requirements. Specifically, with a deterministic arrival stream, deterministic policies with the desired properties do not exist unless $\mathbf{P} = \mathbf{NP}$. Furthermore, with either deterministic or Poisson arrival streams, randomized policies with the desired properties do not exist unless $\mathbf{NP} \subseteq \mathbf{BPP}$.

Remarkably, our negative results continue to hold even if the required throughput is very low, equal to a

[1] In this paper, unless otherwise specified, the term *"low"* means *polynomial* in the number of queues in the network.

vanishing fraction, $o(n^{\epsilon-1})$, of the maximum possible throughput. As a corollary, we conclude that unless $\mathbf{NP} \subseteq \mathbf{BPP}$, Tassiulas's randomized policies cannot have low delay.

In what follows, we describe two alternative formulations that have been considered in the literature: one involving independent set constraints, and one involving the popular SINR wireless interference model. We prove our negative results for both of these models, in Sections 4 and 5, respectively. In the process, we also establish that determining whether a given vector of arrival rates belongs to the stability region, even in an approximate sense, is computationally hard, which is interesting in its own right.

2 Models

In this section, we present the two communication network models that we will study, and introduce the required concepts and notation.

2.1 Model I: Independent Set Constraints

We consider a network, modeled as an undirected graph $G = (V, E)$. Here, $V = \{1, \dots, n\}$ is the set of nodes, and E is the set of undirected edges. We consider a discrete-time model and assume that at each time step, any node can transmit a packet (if it has one), subject to the constraint that no two neighbors can transmit simultaneously. This model arises in various contexts, including [6, 1, 3].

At each node i, packets arrive according to an external arrival process with rate λ_i. We let $A_i(\tau)$ be the number of packets that arrive at node i during time slot τ, and let $\bar{A}_i(\tau)$ be the cumulative number of arrivals, so that $\bar{A}_i(\tau) = A_1(1) + \cdots + A_i(\tau)$. Given that we are interested in negative results, it suffices to consider rather simple arrival processes, such as the ones below, because the results then readily apply to more general classes of arrival processes.

(i) *Deterministic arrivals:* The packet arrival processes are deterministic and satisfy

$$|\bar{A}_i(\tau) - \lambda_i \tau| \leq c, \qquad \forall\, i,\ \tau,$$

where c is an absolute constant.

(ii) *Poisson arrivals:* The random variables $A_i(\tau)$, $i = 1, \dots, n$, $\tau \geq 1$, are independent Poisson random variables, with $\mathbb{E}[A_i(\tau)] = \lambda_i$, for all i, τ.

Each node has an associated queue. Let $Q_i(\tau)$ denote the queue-size at node i at the end of slot τ. We assume that the system starts empty, i.e., $Q_i(0) = \mathbf{0}$, for all i. Let $D_i(\tau)$ and $\bar{D}_i(t)$ denote the number of departures during slot τ (respectively, the cumulative number of departures in the first τ time slots) from queue i. Thus,

$$\begin{aligned} Q_i(\tau) &= \bar{A}_i(\tau) - \bar{D}_i(\tau) \\ &= Q_i(\tau - 1) + A_i(\tau) - D_i(\tau), \qquad \forall\, i,\ \tau. \end{aligned} \tag{1}$$

The departures at each time slot are determined according to a scheduling policy. In particular, we define a *deterministic policy* as a mapping that determines each $D_i(\tau)$ as a function of the problem data (the graph G and the arrival rates λ_j) and the history of the arrival and departure processes during the first $\tau - 1$ slots. We also consider *randomized* policies under which the choice of the $D_i(\tau)$ can take into account some exogenous random variables that are independent of the history of the process during the first $\tau - 1$ time slots. We consider policies that satisfy the following conditions:

(i) $D_i(\tau) \in \{0, 1\}$, for every i and τ.

(ii) If $Q_i(\tau - 1) = 0$, then $D_i(\tau) = 0$, that is, departures are not possible from an empty queue. This involves an implicit assumption that a packet arriving during slot τ cannot depart during that same slot.

(iii) The set $S(\tau) = \{i \mid D_i(\tau) = 1\}$ is an *independent set*; that is, if $D_i(\tau) = D_j(\tau) = 1$, and $i \neq j$, then $\{i, j\} \notin E$.

As a concrete example, the popular maximum-weight policy chooses at each time τ an independent set $S(\tau)$ for which $\sum_{i \in S(\tau)} Q_i(\tau - 1)$ is largest.

In the sequel, we identify a set $S \subseteq V$ with a binary vector $I \in \{0, 1\}^n$ whose ith component is one if and only if $i \in S$. We also let $\mathcal{I}$ denote the set of all vectors I associated with independent sets (including the empty set).

We now discuss the notions of throughput and delay that we will be concerned with. Given a particular graph, we let Λ be the convex hull of the vectors $I \in \mathcal{I}$. It is well known, and easy to check, that the set Λ is the stability region of the system. More precisely, there exists a policy which is stable whenever λ lies in the interior of Λ, in the sense that

$$\limsup_{\tau \to \infty} \mathbb{E}[Q_i(\tau)] < \infty, \qquad \forall\, i \in V.$$

Furthermore, if $\lambda \notin \Lambda$, no policy can be stable. This motivates us to define the *load factor* $\rho(\lambda)$ associated to a particular graph G and arrival rate vector λ by

$$\rho(\lambda) = \min\{\alpha \mid \lambda \in \alpha\Lambda\},$$

where $\alpha\Lambda = \{\alpha x \mid x \in \Lambda\}$. Note that $\rho(\lambda) \leq 1$ is equivalent to the condition $\lambda \in \Lambda$.

We are generally interested in maximum-throughput policies (i.e., policies that are stable whenever $\rho(\lambda) < 1$, and with low delay. Equivalently, in view of Little's law, we are interested in policies with low expected queue sizes. On the other hand, a uniform bound on the expected queue sizes is not possible: even for an M/M/1 queue, the steady-state queue size grows at the rate of $1/(1 - \rho(\lambda))$. For this reason, a more meaningful objective is to require an upper bound on the queue sizes whenever $\rho(\lambda) \leq \rho_0$, for some constant $\rho_0 < 1$. Unfortunately, as we will show, this objective is not attainable: even if we restrict λ to lie in a small neighborhood of the zero vector, low expected queue sizes turn out to be impossible with a polynomial amount of computation at each step. In order to state such a result precisely, we need an additional definition. We use $\mathbf{1}$ to denote the vector with all components equal to one, and $\langle \cdot\,, \cdot \rangle$ to denote the usual inner product. We use the notation $Q(\tau) = (Q_1(\tau), \ldots, Q_n(\tau))$. We also let

$$\gamma(n) = \frac{n}{2^{\sqrt{\log n}}}.$$

Definition 1. *A policy has the* poly-queue *property if there exists a polynomial $p(n)$ such that for every n-node graph, and every arrival rate vector λ for which $\rho(\lambda) \leq 1/\gamma(n)$, the resulting queue sizes satisfy*

$$\sup_{\tau} \mathbb{E}[\langle Q(\tau), \mathbf{1} \rangle] \leq p(n). \tag{2}$$

Note that $1/\gamma(n) = 2^{\sqrt{\log n}}/n = O(n^{\epsilon}/n)$, for any $\epsilon > 0$, so that we are only considering load factors that are significantly below the maximum possible load factor (namely, 1). In this sense, the throughput requirements of a poly-queue policy are very modest.

2.2 Model II: SINR Model

In this section, we consider a single-hop wireless network with a somewhat special structure, in which each node is either a transmitter or a receiver. We assume an equal number of transmitters and receivers, and index them by $1, \ldots, n$ and $1', \ldots, n'$, respectively. Because we are interested in negative results, we can furthermore restrict to the case where each transmitter i is associated with a single receiver i' to whom it wishes to transmit. We assume the same packet arrival models as in Section 2.1, but introduce an additional parameter, v, the number of bits in each packet. In keeping with the usual information-theoretic assumptions, which generally lead to bit-capacities that are not integer multiples of the packet size, we will consider a model that allows serving a fraction of a packet during a time slot.

We introduce an interference model, under which the reception capabilities of a receiver within range of multiple active transmitters are jeopardized. We assume that the nodes are laid out on some geographical area and that the ability of signals to propagate from transmitter node i to receiver j' is described by a distance parameter $r_{ij'}$. These parameters may reflect features of the surrounding environment in a complicated manner; thus, they may be quite different than Euclidean distances, and in particular, we do not require them to satisfy the triangle inequality.

At each time slot, a transmitter i can be either active (in which case, we write $x_i(\tau) = 1$) or inactive (in which case, we write $x_i(\tau) = 1$). Every active transmitter transmits at a fixed, common power P. The received signal power at receiver i' due to a transmission by transmitter k is denoted by $P_{ki'}$ and is assumed to be of the form

$$P_{ki'} = P g_{ki'},$$

where $g_{ki'}$ is non-negative valued path loss coefficient. Generally, it is modeled as a descreasing function of distance $r_{ki'}$, e.g. $g_{ki'} = \exp(-\beta r_{ki'})$ with positive parameter β. When transmitter i is active, attempting to transmit data to node i', node i' may also receive interfering signals, due to transmissions by other active nodes. We adopt the popular Signal to Interference Noise Ratio (SINR) model, under which the bit rate ("bit capacity") at which node i can transmit to node i' at time τ, denoted by $\overline{C}_i(\tau)$, is given by[2]

$$\overline{C}_i(\tau) = \log(1 + \mathsf{SNR}_i(\tau)),$$

where

$$\mathsf{SNR}_i(\tau) = \frac{P_{ii'} x_i(\tau)}{1 + \sum_{k \neq i} P_{ki'} x_k(\tau)}.$$

We also define

$$C_i(\tau) = \frac{\overline{C}_i(\tau)}{v},$$

which is the capacity in measured in packets per slot.

As in Section 2.1, a (deterministic or randomized) policy chooses the variables $x_i(\tau)$ at each time slot τ, on the basis of the history of the process. We define the number of packets in queue $Q_i(\tau)$, the number of departing packets $D_i(\tau)$, etc., exactly as in Section 2.1, so that Eq. (1) again applies. At each time slot, the variables $x_i(\tau)$ determine the capacities, and the resulting number of departures $D_i(\tau)$ is given by

$$D_i(\tau) = \min\{x_i(\tau) C_i(\tau),\, Q_i(\tau)\}.$$

That is, the (possibly fractional) number of packets transmitted when transmitter i is active, may be as high as the corresponding capacity, but is limited by the amount of data available in the queue.

Given the network parameters $r_{ij'}$, we define the stability region Λ as the set of all vectors $C(\tau) = (C_1(\tau), \ldots, C_n(\tau))$ resulting from the 2^n possible binary vectors $x(\tau) = (x_1(\tau), \ldots, x_n(\tau))$. We define the load factor $\rho(\lambda)$ and the poly-queue property exactly as in Section 2.1.

[2]Throughout the paper, we use logarithms with base 2.

3 Main result

Our results involve certain computational complexity concepts, which we define here for completeness; see textbooks, such as [8], for further details.

Definition 2. *A policy is said to run in* polynomial time *if at each time τ, the required calculations can be carried out in time which is bounded by a polynomial in n, the number of bits needed to specify a rational vector λ of arrival rates, and the maximum of the queue sizes at that time.*

Note that the above definition allows the required computations to be only pseudopolynomial in the current queue sizes, hence it is weaker than the usual definition of polynomial time. On the other hand, our results will show that even with this weaker definition, desirable policies do not exist.

Definition 3. *The class of Bounded-error Probabilistic Polynomial time languages (***BPP***) consists of all languages L such that there is a randomized algorithm that runs in polynomial time and such that for any input x:*

(i) if $x \in L$, the algorithm returns YES, with probability at least $3/4$;

(ii) $x \notin L$, the algorithm returns NO, with probability at least $3/4$,

Definition 4. *The class of Non-deterministic Polynomial time languages (***NP***) consists of all languages L such that there is a (deterministic) algorithm that runs in polynomial time and such that for any input x:*

(i) if $x \in L$, there exists some y of polynomial size such that given the input (x, y), the algorithm returns YES;

(ii) $x \notin L$, then for every input (x, y), where y is of polynomial size, the algorithm returns NO.

Note that **P** $\subseteq$ **BPP** and **P** $\subseteq$ **NP**. On the other hand, it is widely conjectured that **P**$\neq$**NP** and that **BPP** does not contain all of **NP**. As is customary in complexity theory, our negative results are contingent on such conjectures. Our main result is the following.

Theorem 3.1. *Consider either of the two network models introduced in Section 2 (the model with independent set constraints or the SINR model). For the case of deterministic arrivals, there exists no poly-queue deterministic policy that runs in polynomial time, unless* **P**=**NP**. *For either case of deterministic or Poisson arrivals, there exists no poly-queue randomized policy that runs in polynomial time, unless* **NP**$\subseteq$**BPP**.

It is an immediate corollary of Theorem 3.1 that Tassiulas' low-complexity randomized algorithm [9], even though it is throughput optimal, induces super-polynomial queue size. Theorem 3.1 also indicates that the structure of the SINR model does not make the problem any simpler compared to the independent set model. In fact, the proof shows that the model involving independent set constraints can be viewed as a special case of the SINR model.

In the course of the proof of Theorem 3.1, we also establish that computing the load factor $\rho(\lambda)$ within a multiplicative factor $\gamma(n)$ is also hard, even though our tolerance factor $\gamma(n)$ is quite large, namely, $\Omega(n^{1-\epsilon})$. The problems of computing $\rho(\lambda)$ or of answering the stability question whether λ lies in the interior of Λ cannot be any easier. We note that this result is not an obvious corollary of inapproximability results on the cardinality of a maximum independent set (see the proof of Lemma 4.2), and is interesting in its own right.

4 Proof of Theorem 3.1: Independent Set Constraints Model

We start with some preliminaries, continue with some lemmas, and conclude with the proof of the theorem for the independent set constraints model. The general structure of the argument is as follows. Suppose that a polynomial time poly-queue policy is available. By simulating this policy and observing whether the resulting queue sizes remain small or grow large, we can obtain information on $\rho(\lambda)$, within a certain multiplicative factor. This leads to a polynomial time algorithm for determining (approximately) membership in the set Λ. However, using the combinatorial nature of Λ, and available inapproximability results for the maximum independent set problem, such an algorithm is not possible unless one of the common conjectures in complexity theory fails to hold.

We first note some easy general properties of the load factor ρ, which we will be using without further comment. As far as notation is concerned, inequalities between vectors are meant to hold componentwise.

(a) If $\lambda \leq \lambda'$, then $\rho(\lambda) \leq \rho(\lambda')$.

(b) $\rho(\lambda + \lambda') \leq \rho(\lambda) + \rho(\lambda')$.

(c) $\rho(c\lambda) = c\rho(\lambda)$.

(d) If $\langle \lambda, \mathbf{1} \rangle = \nu$, then $\nu/n \leq \rho(\lambda) \leq \nu$.

(e) $1 \leq \rho(\mathbf{1}) \leq n$.

Lemma 4.1.

(a) Consider either the deterministic or the Poisson arrival model. Suppose that there exists a polynomial time poly-queue randomized policy. Then, there exists a polynomial time randomized algorithm that on input (n, G, λ) determines, with probability at least $1 - O(1/n)$, some $\hat{\rho}$ with the property $\hat{\rho} \leq \rho(\lambda) \leq 4\gamma(n)\hat{\rho}$.

(b) Consider the deterministic arrival model. Suppose that there exists a polynomial time poly-queue deterministic policy. Then, there exists a polynomial time deterministic algorithm that on input (n, G, λ) determines some $\hat{\rho}$ with the property $\hat{\rho} \leq \rho(\lambda) \leq 4\gamma(n)\hat{\rho}$.

Proof. We start with the proof of part (a), for the case of the Poisson arrival model. Consider a polynomial time poly-queue randomized policy π, and let $p(n)$ be the polynomial bound on the resulting queue sizes [cf. Eq. (2)]. Without loss of generality, assume that $p(n) \leq n^K$ for some $K > 0$.

We first show that there exists a polynomial time randomized algorithm $\mathcal{A}$ that produces with probability at least $1 - O(1/n^2)$ the output YES whenever $\rho(\lambda) \leq 1/\gamma(n)$, and NO whenever $\rho(\lambda) > 2$. (The output is arbitrary, otherwise.) The algorithm $\mathcal{A}$ is as follows. If $\lambda_i \geq 1$ for some i, it outputs NO. (In this case, we have $\rho(\lambda) \geq 1 > 1/\gamma(n)$ and this output satisfies our specifications.) If $\lambda_i \leq 1$, for all i, we proceed as follows. Generate independent Poisson arrival processes according to the arrival vector λ, and simulate the network under policy π, starting with empty queues, until time $T = 4n^{K+4}$. The output of the algorithm is as follows:

(i) if $\langle Q(T), \mathbf{1} \rangle \leq n^{K+2}$, then output YES;

(ii) if $\langle Q(T), \mathbf{1} \rangle \geq n^{K+3}$, then output NO;

(iii) otherwise, the output is arbitrary.

Note that this a randomized algorithm that runs in polynomial time. This is because at any time $\tau \le T$, the queue sizes are polynomially bounded, with high probability. Thus, the policy requires a polynomial amount of computation at each step,[3] and there are only polynomially many steps. We now show that algorithm $\mathcal{A}$ has the claimed properties, for the case where $\lambda_i \le 1$ for all i.

Suppose that $\rho(\lambda) \le 1/\gamma(n)$. Since the policy π has the poly-queue property, we have

$$\mathbb{E}[\langle Q(T), \mathbf{1}\rangle] \le p(n) \le n^K. \tag{3}$$

From the Markov inequality, we obtain

$$\Pr\left(\langle Q(T), \mathbf{1}\rangle \le n^{K+2}\right) \ge 1 - \frac{1}{n^2}.$$

Thus, when $\rho(\lambda) \le 1/\gamma(n)$, the output of the algorithm is YES, with probability at least $1 - O(1/n^2)$, as desired.

Suppose now that $\rho(\lambda) > 2$. Let $\mu_i = \bar{D}_i(T)/T$, which is the departure rate observed during the interval $[0, T]$. Note that $(\mu_1, \ldots, \mu_n)$ is the average of the vectors $(D_1(\tau), \ldots, D_n(\tau))$, $\tau = 1, \ldots, T$. The latter vectors all belong to $\mathcal{I}$ because the queues served at each time must correspond to an independent set. It follows that μ belongs to Λ, the convex hull of $\mathcal{I}$, and that $\rho(\mu) \le 1$. Suppose that $\lambda_i \le \mu_i + 1/n$, for all i. Then,

$$\rho(\lambda) \le \rho(\mu) + \rho(\mathbf{1}/n) \le 1 + 1 = 2,$$

which contradicts the assumption $\rho(\lambda) > 2$. Thus, there exists some i for which $\lambda_i > \mu_i + 1/n$. From Chebyshev's inequality, and using the assumption $\lambda_i \le 1$ in the bound $\text{var}(\bar{A}_i(T)) = \lambda_i T \le T$, we have

$$\Pr\left(\bar{A}_i(T) \le \lambda_i T - T/2n\right) \le \frac{4\text{var}(\bar{A}_i(T))n^2}{T^2} \le \frac{4Tn^2}{T^2} = \frac{4n^2}{T} = \frac{1}{n^{K+2}} \le \frac{1}{n^2}.$$

Therefore, with probability at least $1 - O(1/n^2)$, we have $\bar{A}_i(T) \ge \lambda_i T - T/2n$. Using the inequality $\lambda_i > \mu_i + 1/n$, we obtain

$$Q_i(T) = \bar{A}_i(T) - \bar{D}_i(T) \ge \lambda_i T - \frac{T}{2n} - \mu_i T > \frac{T}{n} - \frac{T}{2n} = \frac{T}{2n} = 2n^{K+3} \ge n^{K+3},$$

and the output of the algorithm is NO, as desired.

We now show how to use algorithm $\mathcal{A}$ to determine $\rho(\lambda)$ within a multiplicative factor of $4\gamma(n)$. We are given (n, G, λ), and the corresponding total arrival rate $\nu = \langle \lambda, \mathbf{1}\rangle$. Consider the output of algorithm $\mathcal{A}$ when the input is $(n, G, 2^k\lambda)$, for some integer k. For the purposes of this argument, let us temporarily assume that the probability of error of $\mathcal{A}$ is zero.

If $2^k\nu \le 1/\gamma(n)$, then $\rho(2^k\lambda) \le 2^k\nu \le 1/\gamma(n)$, and the output of the algorithm is YES. If $2^k\nu > 2n$, then $\rho(2^k\lambda) > 2$, and the output of the algorithm is NO. Thus, we can determine the outputs of $\mathcal{A}$, for the inputs $(n, G, 2^k\lambda)$ for all k, by running the algorithm $\mathcal{A}$ only for those k for which $1/\gamma(n) < 2^k\nu \le 2n$, i.e., by running algorithm $\mathcal{A}$ only $\lceil \log(2n\gamma(n)) \rceil = O(n)$ times. Since the output is YES when k is small and NO when k is large, there exists some k^* for which the output associated with k^* and $k^* + 1$ is YES and NO, respectively. Since $\mathcal{A}(n, G, 2^{k^*}) = \text{YES}$, we must have $\rho(2^{k^*}\lambda) \le 2$, i.e., $\rho(\lambda) \le 2/2^{k^*}$. Since $\mathcal{A}(n, G, 2^{k^*+1}) = \text{NO}$, we must have $\rho(2^{k^*+1}\lambda) > 1/\gamma(n)$, so that $\rho(\lambda) \ge \frac{1}{2^{k^*+1}\gamma(n)}$. By taking $\hat{\rho} = \frac{1}{2^{k^*+1}\gamma(n)}$, we have $\hat{\rho} \le \rho(\lambda) \le 4\gamma(n)\hat{\rho}$, as desired.

[3] For an entirely rigorous argument, there are some details to be dealt with here, having to do with the amount of effort required to simulate Poisson random variables. These details are handled formally by truncating the Poisson distribution at a sufficiently high but polynomially bounded threshold, so that a polynomially bounded number of random bits suffices. Furthermore, the truncation will result in only a small probability that the algorithm produces incorrect results, which is acceptable given that we allow for an $1/n^2$ error probability.

We have so far ignored the possibility that algorithm $\mathcal{A}$ makes an error. However, since $\mathcal{A}$ is to be run less than n times, and the probability of error at each run is no more than $O(1/n^2)$, it follows from union bound that the overall probability of error is no more than $O(1/n)$. This completes the proof of the lemma for the case of Poisson arrivals.

If arrivals are deterministic, the same argument goes through. The only difference is that Chebyshev's inequality is replaced by a deterministic inequality. Finally, if arrivals are deterministic and there exists a deterministic poly-queue policy that runs in polynomial time, then the simulation and algorithm $\mathcal{A}$ are all deterministic. In particular, $\rho(\lambda)$ can be determined within a multiplicative factor using a deterministic algorithm. This completes the proof of Lemma 4.1(a) and (b).

□

Let $\alpha(G)$ denote the maximum cardinality of the independent sets in G, and let $\chi(G)$ denote the minimum number of colors required for vertex coloring of G. We will now show that an algorithm that computes $\rho(\lambda)$ approximately can be used to compute $\alpha(G)$ approximately. The difficulty here is that although the value of $\rho(\lambda)$ provides an easy lower bound on $\alpha(G)$, it does not provide in general a comparable upper bound. We handle this difficulty by using a convenient transformation, due to Lund and Yannakakis [4], which transforms a general graph G into a related graph H with the property that $\alpha(H) = \alpha(G)$, and on which there is a fairly tight relation between $\rho(\lambda)$ and $\alpha(H)$. The precise form of the result that we will use is as follows.

Proposition 4.1 ([4], Proposition 4.1)**.** *There exists a polynomial time algorithm which given a graph G with n nodes, a prime number p, and an integer r, with $p \geq r \geq n$, generates a graph H with $m = p^3 r$ nodes such that*

$$\alpha(G) = \alpha(H), \qquad \textit{and} \qquad p^3 r \leq \alpha(H)\chi(H) \leq p^3 r(1 + \alpha(H)/r).$$

Note that for the graph H obtained from this construction, the product $\alpha(H)\chi(H)$ is completely determined within a factor of two. Of course, for this construction to be useful, we need to pick r and p polynomially small. For concreteness, we can choose $r = n$, and let p be the first prime larger than n; such a p is bounded by a polynomial in n, as a consequence of the prime number theorem.

Lemma 4.2. *Suppose that there exists a randomized polynomial time algorithm that computes a $\hat{\rho}$ such that $\hat{\rho} \leq \rho(\lambda) \leq 4\gamma(n)\hat{\rho}$, with probability $1 - O(1/n)$. Then, there exists a randomized polynomial time algorithm that can determine $\alpha(G)$ within a multiplicative factor of $8\gamma(n)$, with probability $1 - O(1/n)$. If the algorithm for $\hat{\rho}$ is deterministic, the algorithm for $\alpha(G)$ can also be taken deterministic.*

Proof. Given a graph G with n nodes, we proceed as follows. We first use the algorithm from Proposition 4.1 to generate a graph H with $m = p^3 r$ nodes, and with m only polynomially larger than n. We then consider an arrival vector λ equal to $\mathbf{1}$, and determine the resulting load factor $\rho(\mathbf{1})$ on the graph H (to be denoted simply by ρ, within a multiplicative factor of $4\gamma(n)$. Let Λ be the stability region associated with H.

Consider a coloring of H with the minimal number $\chi(H)$ of colors. Nodes with the same color form an independent set in H. By serving similarly colored nodes at each time step, and alternating between the different colors, we see that the vector $\mathbf{1}$ of arrivals can be served in $\chi(H)$ time steps. Thus, the vector $\frac{1}{\chi(H)}\mathbf{1}$ belongs to Λ, or $\rho \leq \chi(H)$. Thus,

$$\alpha(H) \leq \frac{2m}{\chi(H)} \leq \frac{2m}{\rho}.$$

From the definition of ρ, we have that $\frac{1}{\rho}\mathbf{1}$ belongs to the convex hull of the set $\mathcal{I}$ of vectors I associated with independent sets. Thus,

$$\frac{1}{\rho}\mathbf{1} = \sum_{I \in \mathcal{I}} \beta_I I,$$

where the coefficients β_I are nonnegative and sum to one. Therefore,

$$\frac{m}{\rho} = \frac{1}{\rho}\langle \mathbf{1}, \mathbf{1} \rangle = \sum_{I \in \mathcal{I}} \beta_I \langle I, \mathbf{1} \rangle \leq \sum_{I \in \mathcal{I}} \beta_I \alpha(H) = \alpha(H).$$

Since $\alpha(G) = \alpha(H)$, we have

$$\frac{m}{\rho} \leq \alpha(G) \leq \frac{2m}{\rho}.$$

Since ρ can be determined within a multiplicative factor of $4\gamma(n)$, it follows that $\alpha(G)$ can be determined within a multiplicative factor of $8\gamma(n)$. □

We now invoke the following result of Trevisan [11] on the nonapproximability of the independent set problem.

Proposition 4.2. *An algorithm that can determine $\alpha(G)$ for any graph with maximum vertex degree B within factor $B/2^{\Omega(\sqrt{\log B})}$, can be used, with a polynomial time reduction, to decide the satisfiability of any 3-CNF formula.*

We can now complete the proof of the theorem. If there exists a polynomial time poly-queue randomized policy (for either deterministic or Poisson arrivals), then (by Lemma 4.1) there exists a polynomial time randomized algorithm for determining ρ within a factor of $4\gamma(n)$ (with a small error probability), and therefore (by Lemma 4.2) a polynomial time randomized algorithm for determining $\alpha(G)$ within a factor of $8\gamma(n)$. Since $8\gamma(n) = O(n/2^{\Omega(\sqrt{\log n})})$, Proposition 4.2 (with $B = n$), this implies that there exists a polynomial time randomized algorithm for satisfiability, and therefore, $\mathbf{NP} \subseteq \mathbf{BPP}$. The argument for the case of deterministic arrivals and deterministic policies is entirely similar, except that we end up with a deterministic algorithm for satisfiability, and therefore **P=NP**.

5 Proof of Theorem 3.1: SINR Model

The main idea of the proof is the following. With a suitable choice of the problem parameters, v, P, ϑ, and $r_{ij'}$, an instance of the model with independent set constraints (Model I) can be closely approximated by an instance of the SINR model (Model II) with almost the same stability region and load factors. Then, a poly-queue policy for Model II that runs in polynomial time results in a policy with similar properties for Model I, contradicting what we have already proved for the latter. A similar reduction from the independent set problem was used in Luo and Zhang [5] in the context of dynamic spectrum allocation.

To this end, consider an instance (n, G, λ) of Model I, with edge set E, and stability region Λ. We say that i and j are neighbors if $i \neq j$ and $\{i, j\} \in E$. We construct an instance of Model II, with n transmitters labeled $1, \ldots, n$, and n receivers, labeled $1', \ldots, n'$ and the same arrival vector λ. We let

$$P = e^{3n} \qquad \text{and} \qquad v = n.$$

We choose distances so that the effective path loss coefficients are such that $g_{ii'} = \exp(-2n)$ for all i, $g_{ij'} = g_{ji'} = \exp(-2n)$ if $\{i, j\} \in E$. Otherwise, we set $g_{ij'} = g_{ji'} = \exp(-4n)$. Consider an arbitrary policy π' for the SINR model. Recall that $x_i(\tau)$ is a binary variable indicating whether transmitter i is active during slot τ. We have the following three possibilities.

(a) If $x_i(\tau) = 0$, then $\mathsf{SNR}_i(\tau) = 0$, and $C_i(\tau) = 0$.

(b) If $x_i(\tau) = 1$ and $x_k(\tau) = 0$ for every neighbor k of i, then interference at i is only due to nodes k for which $g_{ki'} = \exp(-4n)$. Then,

$$\frac{e^n}{1 + ne^{-n}} \leq \frac{P\exp(-2n)}{1 + nP\exp(-4n)} \leq \mathsf{SNR}_i(\tau) \leq P\exp(-2n) = e^n.$$

Therefore, for large enough n, and using the fact $\overline{C}_i(\tau) = \log(1 + \mathsf{SNR}_i(\tau)) \leq \mathsf{SNR}_i(\tau)$,

$$\frac{n}{2} \leq \overline{C}_i(\tau) \leq n.$$

Dividing by $v = n$, we obtain

$$\frac{1}{2} \leq C_i(\tau) \leq 1. \tag{4}$$

(c) If $x_i(\tau) = 1$ and there exists a neighbor k of for which $x_k(\tau) = 1$, then k satisfies $g_{ki'} = \exp(-2n)$, and

$$\mathsf{SNR}_i(\tau) \leq \frac{P\exp(-2n)}{1 + P\exp(-2n)} \leq 1.$$

It then follows that, $\overline{C}_i(\tau) \leq 1$, and

$$C_i(\tau) \leq \frac{1}{n}. \tag{5}$$

We now relate the stability region of our instance of Model II (denoted by Λ') to the stability region Λ of the instance of Model I. Recall that Λ is the convex hull of the set of binary vectors I associated with independent sets in G. Pick an independent set S, and let every transmitter $i \in S$ be active. We are then in case (b) above, and we have $C_i \geq 1/2$ for every $i \in S$. This shows that for each one of the vectors I (which generate the set Λ), the vector $I/2$ is in Λ'. Thus, $\Lambda/2 \subseteq \Lambda'$, which implies that

$$\rho'(\lambda) \leq 2\rho(\lambda), \tag{6}$$

where $\rho'(\lambda)$ is the load factor in Model II. Here, and in the remainder of the proof, we use primes to denote variables associated with Model II.

Suppose that the policy π' for Model II has the poly-queue property. We will show how π' can be used to construct a policy π for the instances of Model I that we have just constructed, with favorable properties, thus contradicting our negative results for Model I.

Policy π operates as follows. The policy π (which is applied to Model I) simulates the operation of policy π' on Model II, under the exact same arrival processes, on a sample path basis, i.e., with $A'_i(\tau) = A_i(\tau)$, for all i, τ. We define binary variables $y_i(\tau)$ by setting $y_i(\tau) = 1$ if and only if $x_i(\tau) = 1$ and $x_k(\tau) = 0$ for every neighbor k of i (this corresponds to case (b) above). Then, $I = \{i \mid y_i(\tau) = 1\}$ is an independent set. We let policy π serve all of the queues in I, to the extent that packets are available. In particular, the departure process for Model I satisfies

$$D_i(\tau) = \min\{y_i(\tau), Q_i(\tau)\}. \tag{7}$$

Recalling the definition of Model II, its departure process satisfies

$$D'_i(\tau) = \min\{x_i(\tau)C_i(\tau),\ Q'_i(\tau)\}. \tag{8}$$

By virtue of Eqs. (4) and (5), we have $C_i(\tau) \leq 1$, and $C_i(\tau) \leq 1/n$ if $y_i(\tau) = 0$. Thus,

$$x_i(\tau)C_i(\tau) \leq y_i(\tau) + \frac{1}{n}.$$

Using this inequality, and by comparing Eqs. (7) and (8), an easy induction shows that

$$Q_i(\tau) \leq Q'_i(\tau) + \frac{\tau}{n}, \qquad \forall\, \tau. \tag{9}$$

Since π' has the poly-queue property, there exists some K such that whenever $\rho'(\lambda) \leq 1/\gamma(n)$, we have $\mathbb{E}[\langle Q'(T), \mathbf{1}\rangle] \leq n^K$, for all T. Whenever $\rho(\lambda) \leq 1/2\gamma(n)$, we have $\rho'(\lambda) \leq 1/\gamma(n)$ [cf. Eq. (6)]. We let $T = 4n^{K+4}$, and then use Eq. (9) to obtain

$$\mathbb{E}[\langle Q(T), \mathbf{1}\rangle] \leq n^K + n \cdot \frac{T}{n} = n^K + 4n^{K+4} \leq 5n^{K+4}.$$

Other than the term $5n^{K+4}$ instead of n^K, and the factor of 2 change from $1/\gamma(n)$ to $1/2\gamma(n)$, none of which are essential, this is the same as the property (3) on which the proof in Section 2.1 was based. According to that proof, with essentially no change, this property leads to a contradiction. This completes the proof of our negative result for Model II.

6 Discussion

Consider a variant of the randomized algorithm proposed by Tassiulas [9] in the context of the model with independent set constraints. Specifically, we have a network graph G, and choose at each time slot τ an independent set in G, and associated binary vector $I(\tau)$, prescribing the queues to be served at that slot. Initially, at $\tau = 1$, we let the schedule $I(1)$ be arbitrary. Given the schedule $I(\tau)$ used at time τ, we choose $I(\tau+1)$ as follows. Select an independent set $R(\tau+1)$ at random. Based on the current queue-size vector $Q(\tau)$, compare the weights of $I(\tau)$ and $R(\tau+1)$, and let $I(t+1)$ to be the vector with the larger weight, with respect to the current queue-size vector, $Q(\tau)$. That is,

$$I(\tau+1) = \arg\max\{\langle Q(\tau), I(\tau)\rangle,\ \langle Q(\tau), R(\tau+1)\rangle\}.$$

In [9], Tassiulas showed that if there is $\delta > 0$, such that for any $\tau \geq 0$,

$$\Pr(R(\tau+1) = \sigma) \geq \delta, \qquad \forall\, \sigma \in \mathcal{I},$$

then the algorithm is stable. Furthermore, it is shown in [7, 2] that this algorithm can be implemented in a distributed manner with a small polynomial computational effort per time slot. The standard Lyapunov function based technique provides a $2^{O(n)}$ upper bound on the expected queue-size. However, it is not apparent whether this is a loose upper bound or whether the average queue-sizes can be exponentially large in n. In particular, the standard network theoretic techniques fail to provide a useful lower bound on the average queue-size. Our Theorem 3.1 indicates that under the common computational hypotheses on **BPP** and **NP**, this algorithm will indeed induce average queue-sizes that are super-polynomial in n, in the worst case over all possible systems with n nodes. In other words, the objective of a computationally efficient algorithm/policy that guarantees low average delay in the interior of the stability region appears to be unachievable. This result, which provides a computational complexity based approach to obtaining lower bounds on average queue-size for large network problems, is fundamentally different from the classical approaches to stochastic networks.

One might be tempted to argue that wireless networks are different from models involving independent set constraints, and that there might be additional structure that could be exploited, rendering the negative results inapplicable. To address this possible objection, we showed that, at least in some regimes, natural interference models are very close to independent set problems, so that the negative results also apply to more realistic wireless network models. One can argue that the regimes used in our proofs are not representative of real-world models. This is a generic issue, always present when using complexity theoretic tools, which are geared towards a worst case analysis. It is interesting direction for future research to identify classes of network graphs and models that allow for both computationally tractable policies and low delay. On the other hand, our results indicate that a search for policies that would be universally applicable is likely to be futile.

One special feature of our SINR model is that the power P at each transmitter was held constant. It would be interesting to derive similar negative results for formulations that involve tunable power levels.

Acknowledgments

Devavrat Shah would like to acknowledge a fruitful discussion with Mihalis Yannakakis on this problem.

References

[1] J H Davis (ed.). Special issue on mobile radio communications. *IEEE Journal on selected areas in communications*, 1984.

[2] K. Jung and D. Shah. Low delay scheduling in wireless network. In *Proceedings of IEEE ISIT*, 2007.

[3] FP Kelly. Stochastic models of computer communication systems. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 379–395, 1985.

[4] C. Lund and M. Yannakakis. On the hardness of approximating minization problems. *Journal of ACM*, 41(5):960–981, 1994.

[5] Z. Luo and S. Zhang. Dynamic spectrum management: Complexity and duality. *IEEE Journal of Selected Topics in Signal Processing*, 2(1):57, 2008.

[6] D. Mitra and PJ Weinberger. Probabilistic models of database locking: solutions, computational algorithms, and asymptotics. *Journal of the ACM (JACM)*, 31(4):855–878, 1984.

[7] E. Modiano, D. Shah, and G. Zussman. Maximizing throughput in wireless network via gossiping. In *ACM SIGMETRICS/Performance*, 2006.

[8] R. Motwani and P. Raghavan. *Randomized algorithms*. Cambridge University Press, 1995.

[9] L. Tassiulas. Linear complexity algorithms for maximum throughput in radio networks and input queued switches. In *IEEE INFOCOM*, volume 2, pages 533–539, 1998.

[10] 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*, 37:1936–1948, 1992.

[11] L. Trevisan. Non-approximability results for optimization problems on bounded degree instances. In *ACM STOC*, 2001.