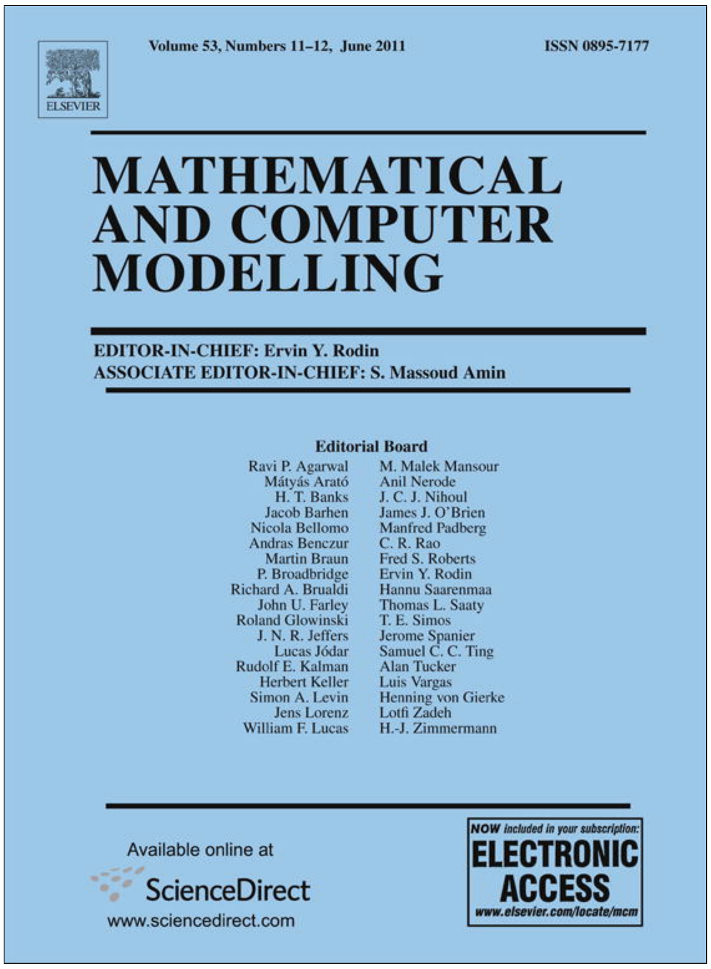


Volume 53, Numbers 11–12, June 2011 ISSN 0895-7177

ELSEVIER

MATHEMATICAL AND COMPUTER MODELLING

Mathematical and Computer Modelling 53 (2011) 2149–2161

Contents lists available at ScienceDirect

Mathematical and Computer Modelling

journal homepage: www.elsevier.com/locate/mcm

Distributed throughput-optimal scheduling framework with delay analysis in multi-hop wireless networks

Nguyen H. Tran, Choong Seon Hong *, Sungwon Lee

Department of Computer Engineering, Kyung Hee University, Republic of Korea

A R T I C L E I N F O

Article history:
Received 1 January 2010
Received in revised form 21 July 2010
Accepted 5 August 2010

Keywords:
Wireless networks
Scheduling algorithms
Optimal-Throughput
Resource allocation

A B S T R A C T

In wireless networks, due to the shared medium, we require sophisticated algorithms to schedule concurrent transmissions that meet the interference constraint where two nodes cannot transmit simultaneously in a guaranteed interference range of each other. For the general (K-hop) interference model,[1] especially with $K \geq 2$, the throughput-optimal centralized scheduler needs to solve a NP-Hard problem. It leads to the desire of a distributed, low-complexity but throughput-optimal scheduling algorithm. Inspired by this problem, we generalize a randomized scheduling framework for a K-hop interference model and prove that any scheduling algorithm can achieve the capacity of the system if it satisfies the constraints of this framework. Then, we develop two randomized distributed scheduling algorithms which can be integrated into this framework. In spite of having the constant time-complexity, the first proposed algorithm can lead to the exponential growth of the network delay. The second one is a maximal matching algorithm having better delay performance with polynomial growth of delay in the network size. We also show that the whole time-complexity of this randomized distributed scheduling framework is polynomial in network size with any finite value of K.

1. Introduction

Compared to wireline networks, wireless networks have a number of distinguishing characteristics like topology dependence, interference, time-varying channels and limited resources. This leads to the demand for designing a highly efficient resource utilization for wireless networks. There have been many papers addressing resource allocation schemes while taking into account those features. The seminal paper of Tassiulas and Ephemides [1] on Max Weight scheduling specified that by using the queue length information, their scheduling algorithms can stabilize the networks provided that the set of user arrival rates fall within the so called stability region of the network. The stability region is also referred to as the capacity region of a constrained queueing system and such a scheduling policy is called throughput-optimal when it stabilizes the queue length for any set of user flow rates which lies within the capacity region. A number of such optimal scheduling algorithms [1–3] proposed to achieve maximum throughput require solving a very difficult global optimization problem in every slot, taking into account the global network topology and queue length information of every link in the network. However, due to the heavy communication overhead of central control in typical wireless networks, it seems infeasible to obtain a centralized solution of this problem. This attracts researchers' attention to the distributed scheduling policies with low-complexity computation, but it comes at the cost of sacrificing a significant portion of the network capacity.

* Corresponding author.
E-mail addresses: nguyenth@khu.ac.kr (N.H. Tran), cshong@khu.ac.kr (C.S. Hong), drsungwon@khu.ac.kr (S. Lee).

[1] This is the generalization of the interference model where two nodes cannot transmit simultaneously if they are within K-hop neighbors of each other due to the shared wireless medium.

0895-7177/$ – see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.mcm.2010.08.010

Here, distributed implementations mean each node needs only to communicate with its neighbors so that the computation to gather the necessary information is performed locally.

Generally, distributed scheduling schemes can be classified into three categories as in [4]. One of them is the Greedy Maximal scheduling algorithm [5–7], which requires only local message exchange and provides a very appealing throughput achieved in practice [7]. A new schedule of this class is constructed in a greedy way that when the largest queue-length link is chosen, no other links in its interference range can be scheduled. In this policy, the set of links chosen for a transmission is a maximal schedule.[2] Another distributed approach is the Constant-Time algorithm [8–10] in which the number of computations and local message exchanges at every slot is a constant, independent of the number of nodes in wireless networks. While both of these approaches guarantee only a fraction of the capacity region, most of Pick-and-Compare algorithms [11–13], where the original one was first proposed and analyzed in [14], can provide the maximum throughput. In this policy, a transmission schedule in time-slot t is constructed by choosing the one with larger total weight between the schedule in time-slot $t-1$ and a newly generated schedule in time-slot t.

Most of the above scheduling policies have been investigated under some specific types of interference models, specifically 1-hop interference model [1,15,16,13,7], (used for Bluetooth and FH-CDMA networks) and 2-hop interference model [6,12] (used for 802.11 networks). Both these models are only specific instances of the class of K-hop interference models, first generalized in [5]. When increasing K, more and more stringent interference constraints will be imposed on links in the network, and wider applicability of the wireless networks is covered. In [5], the authors showed that the centralized optimal scheduling is polynomial time solvable under the 1-hop interference model, but it is NP-Hard and Non-Approximable in case of $K \geq 2$. In the class of distributed scheduling, many algorithms guarantee a $\frac{1}{2}$ capacity region [15,6] for the primary interference model ($K = 1$). As K increases, even more of the network capacity needs to be sacrificed for the distributed implementation [7]. By using the Pick-and-Compare approach, the authors in [12,13] developed distributed scheduling algorithms working for 1-hop and 2-hop interference models. Even though it can achieve maximum throughput, this work comes at the cost of polynomial computation.

From all of the discussions above, in this paper, we consider the problem of how to achieve the maximum throughput in wireless networks with distributed scheduling algorithms under the K-hop interference constraint. We choose the Pick-and-Compare approach as the randomized scheduling framework for our proposed algorithms. Our main contributions can be summarized as the following.

- In Section 3, we generalize the throughput-optimal distributed scheduling schemes into a randomized scheduling framework and we prove that an algorithm can achieve the optimal performance if it satisfies a set of required constraints of this framework.
- In Section 4, for the Pick operation, we propose two randomized distributed scheduling algorithms that can be integrated into the randomized scheduling framework. Both of them can work for the K-hop interference model. While the first algorithm has constant time-complexity, the second one can return a maximal matching with $O\left(e^{4\Delta^K}\log^2 N\right)$ time-complexity, where N is the number of nodes and Δ is the maximum node degree of the wireless network, i.e. maximum number of links incident to a node. We also show that the probabilities of picking them are guaranteed.
- We also investigate the delay characteristic of the throughput-optimal randomized scheduling framework corresponding to two proposed distributed scheduling algorithms. While the first algorithm provides the network delay bound increasing exponentially, the second algorithm guarantees the delay bound that increases polynomially with network size.
- For the Compare operation, based on the results in [12], which apply for the 2-hop interference constraint, we show that this algorithm can combine with our algorithms into the framework for achieving optimal performance and the total time-complexity of the whole framework is still polynomial for any finite K.

The remainder of this paper is organized as follows. We describe the system models in Section 2. In Section 3, we generalize a randomized scheduling framework, also analyze its throughput and delay properties. In Section 4, we propose and exhibit the performance of two distributed algorithms designed for this framework and working under the K-hop interference model. Simulation results are provided in Section 5. We state the conclusion in Section 6.

2. System models

The multi-hop wireless network is represented by a graph $\boldsymbol{G}(\mathcal{N}, \mathcal{L})$, which has the node set $\mathcal{N}$ and the link set $\mathcal{L}$. Denote the number of links and nodes by L and N. There is a set of sessions $\mathcal{S}$ in the network, which represents the network traffic. Sessions (i.e., flows) are identified by a set of source and destination nodes. Denote S as the number of sessions. We assume that each node $n \in \mathcal{N}$ maintains per-session queues with finite buffer. In the network, exogenous arrivals are generated at each source node. We assume that the system is time-slotted and denote $A_n^s(t)$ as the exogenous traffic (in packets/slot) generated by session s during the time-slot t at node n. The arrival process $A_n^s(t)$ is assumed i.i.d. with mean

[2] A maximal schedule is a set of links to which no new link that does not interfere with any of the existing links can be added.

rate $\lambda_n^s = \lim_{T\to\infty} \frac{1}{T}\sum_{t=0}^{T-1} A_n^s(t)$. Maximum arrival rate is assumed to be bounded by a constant $A_{\max}$ and we denote the arrival vector by $\boldsymbol{\lambda} = \left(\lambda_n^s\right)_{n\in\mathcal{N},s\in\mathcal{S}}$.

We denote network resources as a finite set $\mathcal{R}$ of the feasible allocation vectors standing for the achievable rates of network links. Here a feasible allocation means a set of links in which no two links interfere with each other. A resource allocation scheme then aims at choosing in each time-slot a rate schedule $\mathbf{R}(t) = (R_1(t), \ldots, R_L(t)) \in \mathcal{R}$, where $R_l(t)$ is the rate (in packets/slot) of link l. We assume that each time-slot is long enough to accommodate a single packet transmission over each link in $\mathcal{L}$ unless there is interference ($R_l(t) = 0$ or 1). We also assume that if $\mathbf{R} \in \mathcal{R}$ then $\mathbf{R}' \in \mathcal{R}$ if $\mathbf{R}' \preceq \mathbf{R}$ coordinate-wise. The rate allocated to session s over link l at time t is introduced by the notation $R_l^s(t)$, and in a given slot, $R_l^s(t) \in \{0, 1\}$ is 1 if link l serves a packet of session s, and 0 otherwise. This means that $\sum_{s\in\mathcal{S}} R_l^s(t) = R_l(t)$. We denote a constant $\Phi = A_{\max} + 2\Delta$, which will be used later in our algorithm analysis. We assume that the arrival rates always fall within the stability region so that the scheduling algorithms only try to pick the schedules that maximize network throughput. We denote $Q_n^s(t)$ the queue length of session s at node n at time t. We introduce $\mathcal{I}(n)$ and $\mathcal{O}(n)$ as the notation for the set of incoming and outgoing links of node n, respectively. Furthermore, we assume that the scheduling algorithm is aware of the queue lengths and does not schedule empty queues. Then, the queueing dynamics are described by the following recursion:

$$Q_n^s(t+1) = Q_n^s(t) - \sum_{l\in\mathcal{O}(n)} R_l^s(t) + \sum_{l\in\mathcal{I}(n)} R_l^s(t) + A_n^s(t). \tag{1}$$

Next, we introduce the notion of network stability and capacity region, which have already been well-characterized in the literature [1,3].

Definition 1 (*Stability*)**.** The queueing constrained network system is stable, if

$$\limsup_{T\to\infty} \frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}\left[\sum_{s\in\mathcal{S},n\in\mathcal{N}} Q_n^s(t)\right] < \infty. \tag{2}$$

We assume that the queue length process $\mathbf{Q}(t)$ is an ergodic Markov chain. So the network system is stable when the queue length process is positive recurrent.

Definition 2 (*Capacity Region*)**.** The capacity region $\boldsymbol{\Lambda}$ is the closure of the set of all arrival rate vectors $\left(\lambda_n^s\right)_{n\in\mathcal{N},s\in\mathcal{S}}$ for which there exists a resource allocation scheme that can stabilize the network.

Now we restate some main properties relating to the flow conservation and feasibility constraints. A user arrival rate vector $\boldsymbol{\lambda} = \left(\lambda_n^s\right)_{n\in\mathcal{N},s\in\mathcal{S}}$ is in the capacity region $\boldsymbol{\Lambda}$ when for all s, if n is not in the set of destination nodes of session s, there exists a vector $\left(\bar{R}_l^s\right)_{l\in\mathcal{L},s\in\mathcal{S}}$ satisfying:

$$\begin{aligned}
&\bar{R}_l^s \geq 0, \quad \forall l \in \mathcal{L}\\
&\lambda_n^s + \sum_{l\in\mathcal{I}(n)} \bar{R}_l^s = \sum_{l\in\mathcal{O}(n)} \bar{R}_l^s, \quad \forall n \in \mathcal{N}\\
&\left(\sum_s \bar{R}_l^s\right)_{l\in\mathcal{L}} \in Co(\mathcal{R}).
\end{aligned} \tag{3}$$

Here $Co(\mathcal{R})$ denotes the convex hull of set $\mathcal{R}$, which is the smallest convex set that includes $\mathcal{R}$. We can consider $\bar{R}_l^s$ as the long-term rate allocated to session s along link l.

3. Throughput and delay performance of a randomized scheduling framework

3.1. Randomized scheduling framework

In this section, we describe the general throughput-optimal randomized scheduling framework. This randomized scheduling strategy was presented first in the seminal work [14], and then some improvements on relaxation conditions were introduced to cover various types of errors and to adapt with distributed implementations in [13]. We will extend this framework to the K-hop interference model and apply it to multi-hop wireless networks.

Definition 3 (*Randomized Scheduling Framework*)**.** Any algorithm of the randomized scheduling framework aiming at choosing a feasible rate schedule $\mathbf{R}(t)$ must meet the following two constraints in its operations. First, it generates a new random schedule $\mathbf{N}(t) \in \mathcal{R}$ satisfying the first constraint:

- **C1** : $\mathbb{P}[\mathbf{N}(t) = \mathbf{R}^\star(t)] \geq \delta$, for some $0 < \delta < 1$ and for all t.

And then, the real schedule at time-slot t, $\mathbf{R}(t)$, is decided by the second constraint:

- **C2** : $\mathbb{P}\left[W_{\mathbf{R}(t)}(t) \geq \max\{W_{\mathbf{R}(t-1)}(t), (1-\gamma)W_{\mathbf{N}(t)}(t)\}\right] \geq 1-\phi$.

In what follows, for an arbitrary resource allocation scheme, the weight of the rate schedule $\mathbf{R}(t)$ at time-slot t is denoted by $W_{\mathbf{R}(t)}(t)$. Let *tx*(l) and *rx*(l) be the transmitting and receiving nodes respectively of link l, we have:

$$W_{\mathbf{R}(t)}(t) = \sum_{l\in\mathcal{L}} R_l(t) \max_{s\in l} |Q^s_{tx(l)}(t) - Q^s_{rx(l)}(t)|.$$

And the *optimal rate schedule vector* at time-slot t is defined as:

$$\mathbf{R}^\star(t) = \arg\max_{\mathbf{R}(t)\in\mathcal{R}} W_{\mathbf{R}(t)}(t). \tag{4}$$

This randomized scheduling framework provides a probabilistic guarantee of finding an optimal schedule in **C1** and selects the better schedule in **C2** after comparing at each time-slot. The parameter δ in **C1** allows us to design an algorithm in a low-complexity manner, which simplifies substantially the heavy computation of the optimal schedule at every time-slot. In **C2**, while γ represents the error probability in weight computation of the new randomly generated schedule $\mathbf{N}(t)$, ϕ draws inaccurate estimates of the difference between the weights of the previous schedule $\mathbf{R}(t-1)$ and the new schedule $\mathbf{N}(t)$. Note that the above framework generalizes various scheduling algorithms in the literature, from throughput-optimal with polynomial complexity (i.e., [12,13]) to fractional capacity with constant overhead (i.e., [9,10]) achieving, and from 1-hop (i.e., [9,13]) to 2-hop (i.e., [12]) interference models, respectively. Nevertheless, none of these works can operate under the K-hop interference model.

Before presenting the throughput and delay performance of the framework, we introduce an important concept which is used frequently in later sections.

Definition 4 (*Network Load Parameter*).

$$\theta_{\max} = \sup\{\theta \mid \boldsymbol{\lambda} \in (1-\theta)\boldsymbol{\Lambda}\}.$$

Here $\theta_{\max}$ reflects how heavily the system is loaded (e.g., when $\theta_{\max} = 0$, the system is fully loaded). We can also consider it as the distance between $\boldsymbol{\lambda}$ and the boundary of $\boldsymbol{\Lambda}$. Thus, the set $\boldsymbol{\lambda} \in (1-\theta)\boldsymbol{\Lambda}$ can be viewed as the resulting set of arrival rates within the network capacity region when a $\theta\boldsymbol{\Lambda}$-layer is removed from the boundary.

In the following theorem, we investigate the trade-off between parameters δ, γ, ϕ and the obtained throughput. We assume that the mean arrival rate vector $\boldsymbol{\lambda}$ is fixed over time and lies in the interior of the $(1-\theta)\boldsymbol{\Lambda}$.

Theorem 1. *Any algorithm of the randomized scheduling framework can stabilize the network system if the mean arrival rate vector* $\boldsymbol{\lambda} \in (1-\theta)\boldsymbol{\Lambda}$ *falls within the region* $\left(1-\gamma-2\sqrt{\frac{\phi}{\delta}}\right)\boldsymbol{\Lambda}$.

Proof. The proof is provided in Appendix A. □

From the above theorem, we clearly see how parameters γ, ϕ can affect the loss of network throughput in that the larger γ, ϕ, the more the throughput loss. In order to maximize throughput, the first requirement is $\delta > 0$, and the second one is designing a distributed algorithm with careful communication between nodes to eliminate potential errors when computing and comparing the schedule weights (i.e., $\gamma = 0$ and $\phi = 0$). With this goal, a deterministic algorithm was proposed for the Compare operation for a 1-hop interference model in [13]. Another recent work [12] also introduced an algorithm that works for the 2-hop interference model by transforming the conflict graph and using the spanning tree technique.

Next, we discuss the delay performance of the framework. The delay analysis has already been studied in literature [3,17,18,10] with different system models. The idea is finding the upper bound on the mean queue length of the whole system (i.e., mean delay by Little's formula) based on telescoping the Lyapunov function over times. We extend the analysis on this randomized scheduling framework to show how parameter δ effects the delay property.

Theorem 2. *If any algorithms of the randomized scheduling framework can achieve throughput-optimal performance with* $\delta > 0$, $\gamma = \phi = 0$, *and* $\boldsymbol{\lambda} \in (1-\theta)\boldsymbol{\Lambda}$, *we have:*

$$\limsup_{\tau\to\infty} \frac{1}{\tau}\sum_{t=0}^{\tau-1}\sum_{s,n} Q^s_n(t) \leq \frac{SN^2\Phi}{\theta_{\max}}\left(\frac{2}{\delta}+\frac{\Phi}{2}\right). \tag{5}$$

Proof. The proof is provided in Appendix B. □

4. Algorithm description and analysis

In this section, we propose two algorithms following the randomized scheduling framework. At first, we present the computation model for designing our distributed algorithms.

4.1. Distributed computation model

Each scheduling time-slot is divided into a control phase and a data transmission phase. For scheduling purposes, we focus on the details of time-slot structure of the control phase which include a number of rounds for the sequence Pick-and-Compare operations. We assume that all the nodes are synchronized, which can be relaxed as in [12], and a distributed communication unit takes place in one round. The general structure is shown in Fig. 1.

We define the *time-complexity* of an algorithm as the number of *rounds* required by the algorithm for computing a schedule. We also use the term *with high probability* (w.h.p.) to indicate “with probability at least $1 - \frac{1}{N^{C'}}$ for some $C' > 1$”.

4.2. Randomized Feasible Allocation Algorithm

Algorithm 1 Randomized Feasible Allocation Algorithm

```
1:  At each time-slot, each node $n \in \mathcal{N}$ does
2:  if $n \in N_K(n') \cup N_K(m')$ senses ACK$(n', m')$, $\forall n', m' \in \mathcal{N}$ then
3:      disabled$(n) = 1$
4:  if disabled$(n) \neq 1$ then
5:      $n$ chooses an arbitrary $m \in N(n)$ and sends
6:      RTS with probability $p$
7:  if $n$ senses other RTS's or senses COL then
8:      $n$ sends COL
9:  if $m$ senses $n'$s RTS and senses no COL then
10:     $m$ sends CTS
11: if $m$ senses other's RTS's when sending its CTS then
12:     $m$ sends COL
13: if $n$ senses RTS from $m$ and senses no COL then
14:     $n$ sends ACK$(n, m)$
15: if $m$ senses ACK$(n, m)$ successfully then
16:     $\mathbf{R}(t) := \mathbf{R}(t) \cup (n, m)$
17:
```

We first propose an algorithm named the Randomized Feasible Allocation Algorithm aiming at generating a feasible allocation at the end of the Pick operation. In contrast to existing algorithms based on queue lengths information such as those in [19,4,20,8], in our algorithm each node with backlogged links does not need to exchange queue length information for constructing a feasible schedule. Also, it is fully distributed with constant time-complexity, which is much more flexible and general than existing ones in the literature.

Here, we denote the local (i.e., 1-hop) and K-hop neighbors of node n by $N(n)$ and $N_K(n)$ respectively. We say that a node *senses* a control packet if it can decode a control packet or receive a non-decodable collision packet. At the beginning, a node that wishes to request a matching, sends a *ready-to-send* (RTS) packet to a chosen neighbor node. If it senses another ongoing transmission or a *collision* (COL) packet from that transmission, it sends a COL in the next round. If the receiver can decode its RTS successful and sense no COL packet, it means that no other transmissions are within K hops of the transmitter side at that time.

Subsequently, the receiver responds with a *clear-to-send* (CTS) packet. While sending its CTS, if the receiver detects an ongoing transmission, it will send a COL packet in the next round. Thus, no COL packets being sent by receiver guarantees that no other transmissions are within K hops of the receiver side.

Next, if no COL packet is sensed from both sides, the transmitter broadcasts the *acknowledge* packet (i.e., ACK(n, m)) to announce that their matching is successful. Every node within K hops neighborhood of transmitter n and receiver m after realizing the ID of this link through sensing this ACK(n, m) packet will be *disabled*, preventing them from requesting their matchings in the subsequent rounds. And after receiver senses ACK(n, m), this link is ready to be active in the data transmission phase (i.e., knowing that it belongs to $\mathbf{R}(t)$). Therefore, with the important feature of exchanging the COL packets, the algorithm guarantees that two link matching requests cannot exist within K hops of each other.

Proposition 1. *If $p = \frac{1}{|N(n)|+1}$, $\forall n \in \mathcal{N}$, the above algorithm has $O(1)$ time-complexity and returns a feasible schedule with a positive probability of at least δ satisfying* **C1** *of the randomized scheduling framework as below:*

$$\mathbb{P}[\mathbf{N}(t) = \mathbf{R}^{\star}(t)] \geq \delta = \left(\frac{1}{2e^{4\Delta^K}}\right)^N. \tag{6}$$

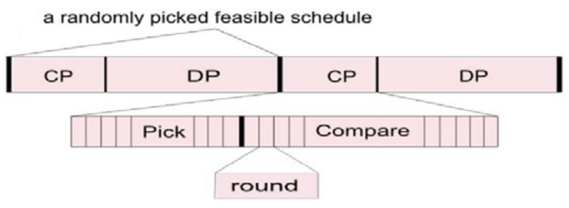


Fig. 1. Time-slot structure of a feasible schedule.

Proof. From the above discussion of the algorithm operation, we clearly see that the maximum number of rounds for each node to terminate is 6, so the time complexity is $O(1)$.

We realize that the link (n, m) will be matched successfully if during the whole operation of this algorithm, provided that nodes n and m are exchanging control packets, all nodes in K-hop neighbors of n

$$N_K(n) = (m_1|m_1 \in N(n)) \cup (m_2|m_2 \in N(m_1)) \cup \cdots \cup (m_K|m_K \in N(m_{K-1})),$$

and all nodes in K-hop neighbors of m

$$N_K(m) = (n_1|n_1 \in N(m) \setminus n) \cup (n_2|n_2 \in N(n_1)) \cup \cdots \cup (n_K|n_K \in N(n_{K-1}))$$

keep silent. We first consider the transmitter side, where node n sends all of its control packets successfully (i.e., without sending COL packets) with a probability $\mathbb{P}_{n\to m}$:

$$\begin{aligned}\mathbb{P}_{n\to m} &= \frac{1}{|N(n)|+1}\left(1-\frac{1}{|N(n)|+1}\right)^{|N(n)|} \underbrace{\left(1-\frac{1}{|N(m_1)|+1}\right)^{|N(m_1)|}\cdots}_{|N(n)|\text{ times}} \\ &\quad \times \underbrace{\left(1-\frac{1}{|N(m_2)|+1}\right)^{|N(m_2)|}\cdots}_{|N(n)|\times|N(m_1)|\text{ times}} \times \cdots \times \underbrace{\left(1-\frac{1}{|N(m_K)|+1}\right)^{|N(m_K)|}\cdots}_{|N(n)|\times|N(m_1)|\times\cdots\times|N(m_{K-1})|\text{ times}} \\ &\geq \frac{1}{|N(n)|+1}\mathrm{e}^{-1}\cdot \mathrm{e}^{-|N(n)|}\cdot \mathrm{e}^{-|N(n)||N(m_1)|}\cdots \mathrm{e}^{-|N(n)||N(m_1)|\cdots|N(m_{K-1})|} \\ &\geq \frac{\mathrm{e}^{-2\Delta^K}}{|N(n)|+1} \quad \text{when } \Delta > 1.\end{aligned}$$

The final inequality follows from the fact that $\Delta = \max_{n\in\mathcal{N}} |N(n)|$ and $(1+\Delta+\Delta^2+\cdots+\Delta^K) \leq 2\Delta^K$ if $\Delta > 1$. We note that the condition $\Delta > 1$ is absolutely satisfied for a standard wireless network with more than one link. By using the same analysis for the receiver side, when node m responds all of its control packets successfully, we have the probability $\mathbb{P}_{n\leftarrow m} \geq \mathrm{e}^{-2\Delta^K}$. So the link (n, m) is matched with probability $\mathbb{P}_{(n,m)} \geq \frac{\mathrm{e}^{-4\Delta^K}}{|N(n)|+1}$. Since the event node n is matched to node m or m' and the event $m \neq m'$ are disjoint, the probability that node n is matched is:

$$\begin{aligned}\mathbb{P}[n \text{ is matched}] &= \sum_{m\in N(n)} \mathbb{P}_{(n,m)} \geq \frac{|N(n)|\,\mathrm{e}^{-4\Delta^K}}{|N(n)|+1} \\ &\geq \frac{\mathrm{e}^{-4\Delta^K}}{2}. \end{aligned} \tag{7}$$

Now, our goal is to find a lower bound on the probability of picking a feasible allocation. Suppose that we have a feasible allocation $F \in \mathcal{R}$, so in order to generate F successfully, exactly $|F|$ nodes are matched, and the remaining nodes are inactive. Therefore, the probability of picking an arbitrary feasible allocation F is:

$$\begin{aligned}\mathbb{P}[F] &\geq \left(\frac{\mathrm{e}^{-4\Delta^K}}{2}\right)^{|F|}\left(1-\frac{\mathrm{e}^{-4\Delta^K}}{2}\right)^{N-|F|} \\ &\geq \left(\min\left\{\frac{\mathrm{e}^{-4\Delta^K}}{2}, 1-\frac{\mathrm{e}^{-4\Delta^K}}{2}\right\}\right)^N \\ &\geq \left(\frac{1}{2\mathrm{e}^{4\Delta^K}}\right)^N. \quad \square\end{aligned}$$

And $\mathbf{R}^\star(t)$ is also a feasible schedule so we have $\mathbb{P}[F = \mathbf{R}^\star(t)] \geq \delta = \left(\frac{1}{2e^{4\Delta^K}}\right)^N$.

Remark 1. Even though the Randomized Feasible Allocation Algorithm can guarantee a positive probability of picking a feasible schedule with $O(1)$ complexity, it does not return a maximal schedule and its picking probability decreases exponentially with the number of nodes in wireless networks.

So, the final statement of Remark 1 leads to the following corollary:

Corollary 1. *With optimal performance of the randomized scheduling framework, the Randomized Feasible Allocation Algorithm can lead to the exponential growth of delay in network size.*

Proof. The result follows from (6) and (5). □

4.3. Randomized Maximal Matching Algorithm

In this subsection, motivated from Remark 1, we continue proposing another feasible allocation algorithm named Randomized Maximal Matching Algorithm, which has more attractive properties than the first one in that it can return a maximal matching for the K-hop interference model with high probability.

Basically, the main idea of this algorithm is still based on the basic 6-round operation of our previous algorithm. By running this operation in more iterations nested in a number of steps decided by a parameter C, we can improve the probability of a matched node and the number of matchings can meet the constraint stipulated by maximal matching policy. We start by analyzing the performance of Randomized Maximal Matching Algorithm.

Algorithm 2 Randomized Maximal Matching Algorithm

```
1:  At each time-slot, each node $n \in \mathcal{N}$ does
2:  if $n \in N_K(n') \cup N_K(m')$ senses ACK$(n', m')$, $\forall n', m' \in \mathcal{N}$ then
3:      disabled$(n) = 1$
4:  if disabled$(n) \neq 1$ then
5:      for $s = 1$ to $(\log N)$ do
6:          $\mathbf{R}_s(t) := \mathbf{R}_{s-1}(t)$
7:          for $i = 1$ to $(C\, e^{4\Delta^K} \log N)$ do
8:              $n$ chooses an arbitrary $m \in N(n)$ and sends
9:              RTS with probability $p$
10:             if $n$ senses other RTS's or senses COL then
11:                 $n$ sends COL
12:             if $m$ senses $n'$s RTS and senses no COL then
13:                 $m$ sends CTS
14:             if $m$ senses other's RTS's then
15:                 $m$ sends COL
16:             if $n$ senses $m'$s CTS and senses no COL then
17:                 $n$ sends ACK$(n, m)$
18:             if $m$ senses ACK$(n, m)$ successfully then
19:                 $\mathbf{R}_s(t) := \mathbf{R}_s(t) \cup (n, m)$
20:
```

Proposition 2. *If $C \geq 6$, then the Randomized Maximal Matching Algorithm has a $O\left(e^{4\Delta^K} \log^2 N\right)$ time-complexity and can return a maximal schedule with the probability of at least δ satisfying* **C1** *of the randomized scheduling framework as below:*

$$\mathbb{P}[\mathbf{N}(t) = \mathbf{R}^\star(t)] \geq \delta = 1 - \frac{1}{N^{C'}} \quad (C' > 1). \tag{8}$$

Proof. Using the proof result of the first proposed algorithm, from (7), we have the probability that node n is not matched during an iteration i is $\mathbb{P}_i[n \text{ is not matched}] \leq \left(1 - \frac{e^{-4\Delta^K}}{2}\right)$. Thus, for one step, we have:

$$\mathbb{P}_s[n \text{ is not matched}] \leq \left(1 - \frac{e^{-4\Delta^K}}{2}\right)^{C\, e^{4\Delta^K} \log N} \tag{9}$$

$$\leq e^{-\frac{e^{-4\Delta^K}}{2} \times C\, e^{4\Delta^K} \log N} \leq e^{-\frac{C \log N}{2}} = \frac{1}{N^{\frac{C}{2}}}. \tag{10}$$

Following a similar argument as in the proof of Proposition 1, we have exactly $|M|$ nodes of any maximal matching $M \in \mathcal{R}$ that are matched at the end of step s with a probability:

$$\mathbb{P}_s[M] \geq \left(1 - \frac{1}{N^{\frac{C}{2}}}\right)^{|M|} \geq 1 - \frac{|M|}{N^{\frac{C}{2}}} \geq 1 - \frac{1}{N^{\frac{C}{2}-1}}.$$

Since $\log N$ steps are fulfilled for the requirement of a maximal matching running time, this algorithm can return a maximal schedule M with probability at least:

$$\begin{aligned}\mathbb{P}[M] &\geq \left(1 - \frac{1}{N^{\frac{C}{2}-1}}\right)^{\log N} \geq 1 - \frac{\log N}{N^{\frac{C}{2}-1}}\\ &\geq 1 - \frac{1}{N^{\frac{C}{2}-2}}.\end{aligned}$$

Letting $C' = C/2 - 2$, and since $\mathbf{R}^\star(t)$ is also a maximal schedule so finally we have $\mathbb{P}[M = \mathbf{R}^\star(t)] \geq \delta = 1 - \frac{1}{N^{\frac{C}{2}-2}}$.

We have shown that in order to achieve a maximal matching with high probability, i.e. $\mathbb{P}[M] \geq 1 - \frac{1}{N^{C'}}$ $(C' > 1)$, the whole algorithm must run in $O(\log N)$ steps and each step includes $O(\mathrm{e}^{4\Delta^K} \log N)$ rounds. So the Randomized Maximal Matching Algorithm has a $O(\mathrm{e}^{4\Delta^K} \log^2 N)$ time-complexity. We complete our proof here. □

Remark 2. Upon sensing the ACK packet, all links in the K-hop neighborhood of the activated matching will disable themselves from the schedule construction process. Hence, the number of feasible links will reduce rapidly over consecutive rounds. So the actual time-complexity of this algorithm is less than $O(\mathrm{e}^{4\Delta^K} \log^2 N)$.

For picking probability, this algorithm outperforms the first proposed one since it can return a maximal schedule w.h.p. Therefore, we have the consequent result.

Corollary 2. *With optimal performance of the randomized scheduling framework, the Randomized Maximal Matching Algorithm can lead to the polynomial growth of delay in network size.*

Proof. The result follows from (8) and (5). □

4.4. Compare algorithm

We realize that the algorithm for the Compare-operation of [12], when combined with our algorithms, can be integrated into the randomized scheduling framework to achieve optimal performance. In this subsection, we first briefly restate the main properties of this Compare algorithm, then we prove how it can work with the K-hop interference model, even though the original design is only destined for a 2-hop interference model.

The main idea of the Compare algorithm in [12] is using a *conflict graph* to compare the total weight of the previously used schedule $\mathbf{R}(t-1)$ and the newly generated schedule $\mathbf{N}(t)$. And then, the decisions can be made independently between disconnected components which are created by the natural partitioning property of the conflict graph. With this independent decision feature, the new chosen schedule $\mathbf{R}(t)$ may be the combination of $\mathbf{R}(t-1)$ and $\mathbf{N}(t)$ because each individual component just chooses the higher weight schedule, which may be different between components.

We analyze the performance of the Compare algorithm that works for the K-hop interference model.

Proposition 3 ([12]). *Given a conflict graph $\mathbf{G}^c(\mathcal{N}^c, \mathcal{L}^c)$ with Δ^c as the maximum degree of $\mathbf{G}^c$, the Compare algorithm correctly chooses the better schedule (i.e. $\gamma = \phi = 0$) in $O(\Delta^c L^c)$ time-complexity.*

Now we find the time-complexity of this algorithm transformed from conflict graph to network graph. We know that the depth of the spanning tree is at most $O(L^c)$, and the distance between a parent and its leaf in a conflict graph is at most K number of rounds to exchange local messages in network graph. We argue similarly for the collision resolution operation, which can take at most $O(\Delta^c)$ time, each of them is at most K rounds in network graph.

Therefore, by using these properties when transforming: $L^c < KN^2$, $\Delta^c < KN$, we have the following result.

Proposition 4. *The time-complexity of the Compare algorithm is $O(N^3)$ under the K-hop interference model.*

We state the final result.

Theorem 3. *The whole complexity of the randomized scheduling framework for the K-hop inference model is $O(N^3)$ with any finite value of K.*

Proof. The result directly follows from Propositions 1, 2 and 4. □

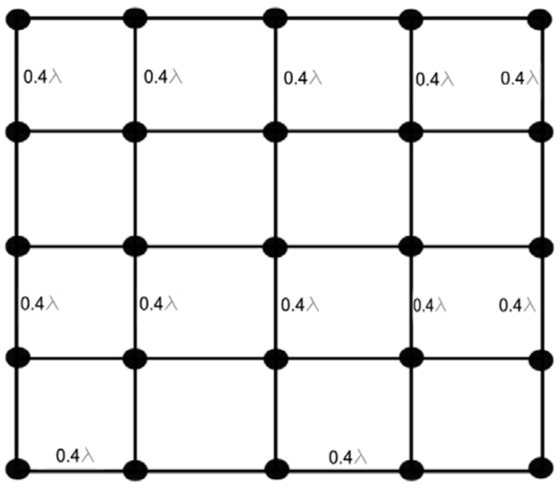
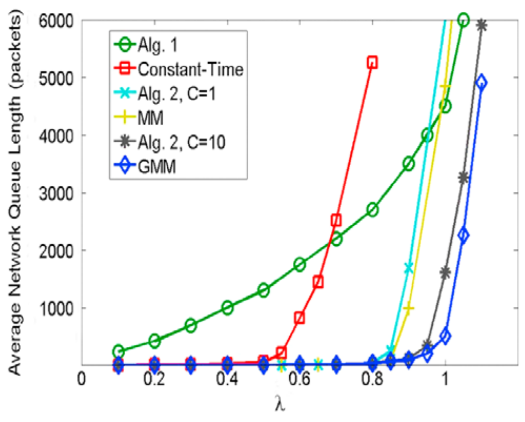
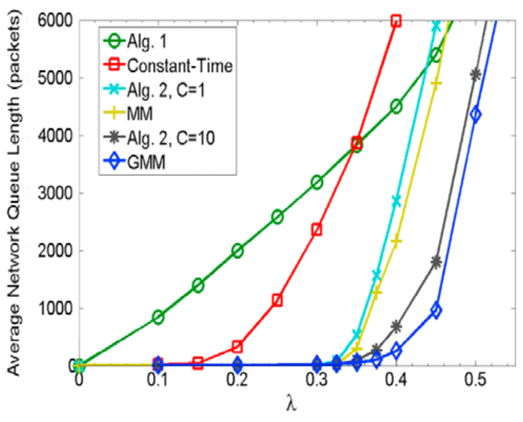


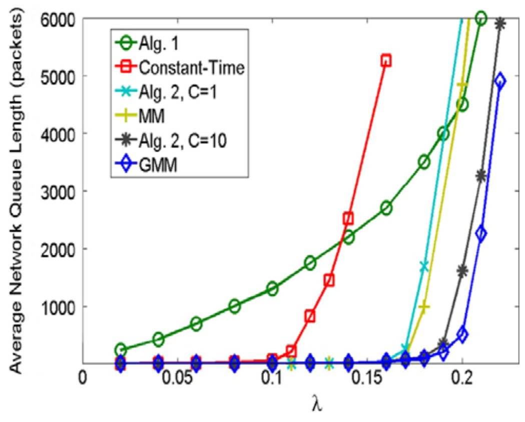
Fig. 2. 5 × 5 grid network.



(a) $K = 1$.



(b) $K = 2$.

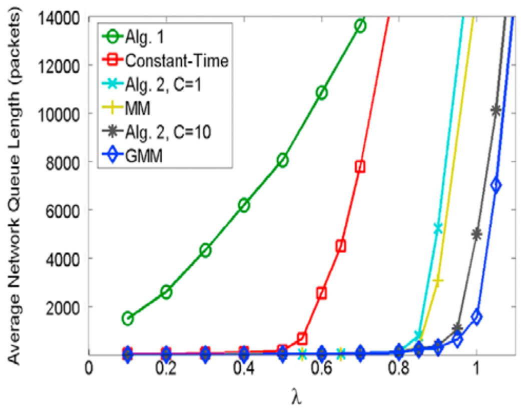


(c) $K = 3$.

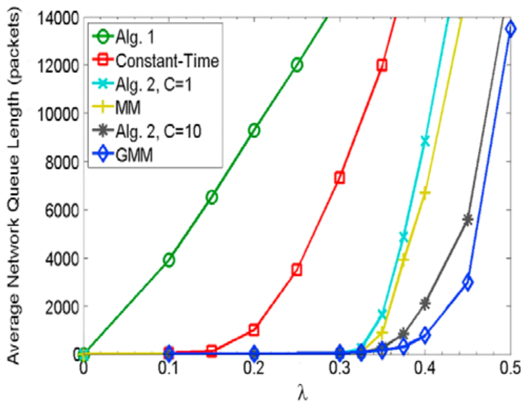
Fig. 3. Capacity region of scheduling algorithms in 5 × 5 grid network.

5. Simulation results

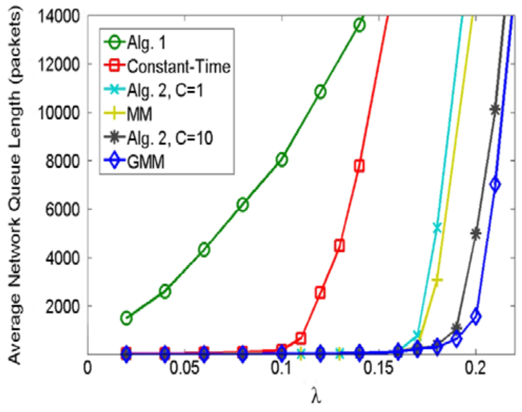
In this section, first we evaluate the throughput performance between the Constant-Time (CT) [20], Maximal Matching (MM), Greedy Maximal Matching (GMM) [15] algorithms and our proposed algorithms with the Compare operation. We use the performance of centralized GMM, which was shown to achieve a nearly optimal performance in most practical scenarios [20], as a reference value. All of the simulations are carried out by our own C++ simulator.



(a) $K = 1$.



(b) $K = 2$.



(c) $K = 3$.

Fig. 4. Capacity region of scheduling algorithms in 10×10 grid network.

The network topology we use for simulations is a 5×5 grid network (see Fig. 2). We implemented all algorithms under the single-hop traffic model, a special case of the traffic model where all sessions consist of only one link. Every link has a capacity of 1 packet/slot. There are 12 heavy-load links that are one-hop away from each other and they have Poisson traffic arrivals with mean 0.4λ. The remaining ones are light-load links with mean 0.2λ. We set the traffic load on links with mean 0.4λ and 0.2λ to ensure that the traffic arrivals generated by each node are no more than λ in this grid topology. The parameter λ characterizes the stability behavior of the network system and apparently the capacity region of the network must lie inside the region of $\lambda < 1$. We note that when K is larger, the capacity region of wireless networks shrink into smaller regions. Fig. 3 shows the total queue length of the system averaged out of the last 1000 time-slots of 10,000 time-slots simulated for different scheduling algorithms. By varying λ to adjust the traffic loads, we observe that the total queue length grows significantly when λ reaches a particular threshold for each algorithm. And each threshold can be considered as the boundary of the capacity region of the corresponding scheduling algorithm. Therefore, through these boundary values, we can estimate the capacity regions of wireless networks and use them to validate the analytical results.

In the case of the 1-hop interference model, Fig. 3(a) shows that with $C = 10$, Algorithm 2 has almost the same performance as GMM, which is close to the optimal and dominates other algorithms. This means that in this case Algorithm 2 can work at loads very close to the threshold $\lambda = 1$ while in case of $\lambda \geq 1$, the queues become unstable. However, when $C = 1$, Algorithm 2 and MM have almost the same performance which is more than 80% of the capacity region. Therefore, the value of C should be chosen carefully to achieve the optimal performance, which supports our analysis result. With more than 80% and 60% capacity empirically obtained, the performance of MM and CT coincide with those in the literature [20,9]. About Algorithm 1, we observe that even though it can attain 100% capacity, it suffers a large queue length, which agrees with the Corollary 1. When the load is light (i.e. $\lambda \leq 0.6$), it shows that Algorithm 1 has the worst performance; however, when the load increases, there is a cross-over performance between Algorithm 1 and Constant-Time, MM and Algorithm 2 with $C = 1$. In the case of the 2-hop and 3-hop interference models (see Fig. 3(b) and (c)), the behaviors of all algorithms are similar to the first case except the capacity region approximately corresponds to $\lambda < 0.45$ and $\lambda < 0.2$.

In order to access the delay performance, we conducted the same simulation scenarios as above in a 10×10 grid network, which shows an increasing number of nodes in the network. As can be seen in Fig. 4, at the threshold, while the queue length of the network just increases approximately three times that of the previous 5×5 grid network by utilizing Algorithm 2, Algorithm 1 explodes remarkably even with light loads. These observations again support our analytical results.

6. Conclusion

We consider throughput-optimal scheduling schemes of wireless networks in a distributed fashion. The Pick-and-Compare approach has been included into a randomized scheduling framework, where we not only provide the throughput-optimal analysis, but also give an upper bound for the delay of the whole system. Specifically, we propose two randomized distributed scheduling algorithms that can be integrated into this framework under the K-hop interference model. We also show that the computational complexity of the framework is unchanged whatever the value of K is and the network delay is exponentially increased or polynomially increased with network size by applying the first or the second algorithm, respectively.

Acknowledgements

This research was supported by the MKE, Korea, under the ITRC support program supervised by the NIPA (NIPA-2010-(C1090-1021-0003)).

Appendix A. Proof of Theorem 1

Proof. Following the technique in [13], we extend this proof to the case of multi-hop flows in wireless networks. First, we introduce some notations relating to a fixed session s. All vector quantities associated with session s are denoted by a subscript s: $\boldsymbol{Q}_s(t) = \left(Q_n^s(t)\right)_{n\in\mathcal{N}}$, $\boldsymbol{R}_s(t) = \left(R_l^s(t)\right)_{l\in\mathcal{L}}$, $\boldsymbol{A}_s(t) = \left(A_n^s(t)\right)_{n\in\mathcal{N}}$, $\boldsymbol{\lambda}_s = \left(\lambda_n^s\right)_{n\in\mathcal{N}}$. We also define a $N \times L$ matrix $\boldsymbol{H}_s$ with $[H_s]_{n,l} = -1$ if n is the receiving node of link l and n is also not the destination of session s, $[H_s]_{n,l} = 1$ if n is the transmitting node of link l, and 0 otherwise. We also use the notation $[.]^T$ to denote the transpose vector. With these notations and from (1), we have the queue revolution of a session s as the following:

$$\mathbf{Q_s}(t+1) = \mathbf{Q_s}(t) - \mathbf{H_s R_s}(t) + \mathbf{A_s}(t).$$

Now we consider a quadratic Lyapunov function $L(t) = \frac{1}{2}\sum_{s,n} Q_n^s(t)^2$ and the corresponding one-step mean drift $\Delta L(t) = \mathbb{E}[L(t+1) - L(t)|\mathbf{Z(t)}]$ where $\mathbf{Z}(t) = (\mathbf{Q}(t), \mathbf{R}(t))$, which forms a Markov chain. We have:

$$\begin{aligned}\Delta L(t) &= \frac{1}{2}\sum_s \mathbb{E}[\|\mathbf{Q_s}(t+1)\|_2^2 - \|\mathbf{Q_s}(t)\|_2^2 \mid \mathbf{Z}(t)]\\ &= \sum_s \frac{1}{2}\mathbb{E}[\|\mathbf{A_s}(t) - \mathbf{H_s R_s}(t)\|_2^2 \mid \mathbf{Z}(t)] + \mathbb{E}[\mathbf{Q_s}(t)^T(\mathbf{A_s}(t) - \mathbf{H_s R_s}(t)) \mid \mathbf{Z}(t)]. \end{aligned} \tag{11}$$

Since the maximum service rate is upper-bounded by one, so the maximum amount of packets per slot that one node may receive (transmit) from (to) its neighboring nodes are upper-bounded by the maximum degree Δ of the network. So the first term of (11) is bounded by a constant $\frac{1}{2}N\Phi^2$. And the second term is equivalent to:

$$\sum_s \mathbf{Q_s}(t)^T(\boldsymbol{\lambda}_\mathbf{s} - \mathbf{H_s R_s^\star}(t)) + \sum_s \mathbf{Q_s}(t)^T(\mathbf{H_{s R_s^\star}}(t) - \mathbf{H_s R_s}(t)). \tag{12}$$

Note that we can rewrite the weight of the schedule $\mathbf{R}(t)$ as:

$$W(t) = \sum_{s\in\mathcal{S}} \mathbf{Q_s}(t)^T \mathbf{H_s R_s},$$

so the maximum weight over all possible schedules is $W^\star(t) = \sum_{s\in\mathcal{S}} \mathbf{Q_s}(t)^T\mathbf{H_s R_s^\star}$ and from (3), the first term of (12) is:

$$\begin{aligned} &= \mathbf{Q_s}(t)^T((1-\theta)\mathbf{H_s \bar{R}_s}(t) - \mathbf{H_s R_s^\star}(t))\\ &\le \mathbf{Q_s}(t)^T((1-\theta)\mathbf{H_s R_s^\star}(t) - \mathbf{H_s R_s^\star}(t))\\ &= -\theta W^\star(t).\end{aligned}$$

So we have:

$$(12) \le -\theta W^\star(t) + (W^\star(t) - W(t)).$$

Substituting the upper bound into (11) yields:

$$\Delta L(t) \le \mathbb{E}[-\theta W^\star(t) + (W^\star(t) - W(t)) \mid \mathbf{Z}(t)] + \frac{1}{2}N\Phi^2. \tag{13}$$

Next, we use the above bound to study the drift in $L(t)$ at every Tth time-slot for some large enough finite T as follows:

$$\mathbb{E}[L(t+T) - L(t)|\mathbf{Z}(t)] \le -\theta\sum_{k=0}^{T-1}\mathbb{E}[W^\star(t+k) \mid \mathbf{Z}(t)] + \frac{1}{2}N\Phi^2 T + \sum_{k=0}^{T-1}\mathbb{E}[W^\star(t+k) - W(t+k) \mid \mathbf{Z}(t)]. \tag{14}$$

Since at most $N\Phi$ arrivals and departures can happen in a time-slot, we have $|W^\star(t+k) - W^\star(t)| \le N\Phi k$ for every t, k. Using this inequality and summing for $t = 0$ to T yields:

$$\sum_{k=0}^{T-1} W^\star(t+k) \ge \sum_{k=0}^{T-1}(W^\star(t) - N\Phi k) \ge TW^\star(t) - N\Phi T^2. \tag{15}$$

Next, let $\Omega(t) \triangleq W^\star(t) - W(t)$, we need to find the upper bound of the second term in the right side of (14). Motivated by **C1** and **C2** of the framework, let us define:

$$\kappa_1 = \inf_{k\ge 0}\{\mathbf{N}(t+k) = \mathbf{R}^\star(t+k)\},$$
$$\kappa_2 = \inf_{k\ge\kappa_1}\{\textbf{C2} \text{ fails at time } t+k\}.$$

By property **C2**, we can easily see that for any $k \in [\kappa_1, \min(\kappa_2, T)]$:

$$\Omega(t+k) \le \gamma W^\star(t+k) \le \gamma W^\star(t) + \gamma N\Phi k. \tag{16}$$

Furthermore, for any $k \ge 0$ which is not in that interval, we have:

$$\Omega(t+k) \le W^\star(t) + N\Phi k. \tag{17}$$

Using (16) and (17), we can obtain:

$$\sum_{k=0}^{T-1} \Omega(t+k) \le W^\star(t)(\min(\kappa_1, T) + \hat{T}) + \gamma W^\star(t)T + N\Phi T^2, \tag{18}$$

where $\hat{T} = T - \min(\kappa_2, T)$, which represents the remaining time after κ_2 until the end of time T, if any. By **C1**, we have $\mathbb{E}[\min(\kappa_1, T)] \le \mathbb{E}[\kappa_1] \le 1/\delta$. And by **C2**, it is easy to see that $\mathbb{P}[\kappa_2 \ge T] \ge 1 - \phi T$, hence $\mathbb{E}[\min(T, \kappa_2)] \ge T(1 - \phi T)$. Thus, $\mathbb{E}[\hat{T}] \le \phi T^2$. And we can rewrite (18) as:

$$\sum_{k=0}^{T-1} \Omega(t+k) \le TW^\star(t)(\gamma + (\delta T)^{-1} + \phi T) + N\Phi T^2. \tag{19}$$

By substituting (15) and (19) in (14), we have:

$$\begin{aligned}\mathbb{E}[L(t+T) - L(t)|\mathbf{Z}(t)] &\le -\theta W^\star(t) + \theta N\Phi T^2 + \frac{1}{2}N\Phi^2 T + TW^\star(t)(\gamma + (\delta T)^{-1} + \phi T) + N\Phi T^2 \\ &= -TW^\star(t)(\theta - \gamma - (\delta T)^{-1} - \phi T) + B \\ &\le -\epsilon\left(\theta - \gamma - 2\sqrt{\frac{\phi}{\delta}}\right)\sum_{s,n} Q_n^s(t) + B,\end{aligned} \tag{20}$$

where the final inequality follows from the fact that $W^\star(t) \ge \sum_{s,n} Q_n^s(t)/SN$ and we put $T = 1/\sqrt{\phi\delta}$, so $\epsilon = T/SN > 0$ and $B = (1+\theta)N\Phi T^2 + \frac{1}{2}N\Phi^2 T > 0$.

By choosing $\theta > \gamma + 2\sqrt{\frac{\phi}{\delta}}$, (20) shows that the Foster–Lyapunov criterion is satisfied [21], leading to $\limsup_{T\to\infty} \frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}\left[\sum_{s\in\mathcal{S}, n\in\mathcal{N}} Q_n^s(t)\right] \le B/\epsilon < \infty$. We complete the proof here. □

Appendix B. Proof of Theorem 2

Proof. First, we need to find the upper bound of $W^\star(t) - W(t)$. Denote $\{t_k\}_{k=0,1,\dots}$ as the sequence of random time slots at which $\mathbf{N}(t_k) = \mathbf{R}^\star(t_k)$, so:

$$W^\star(t_k) = W(t_k). \tag{21}$$

Now, with an arbitrary $t \in (t_k, t_{k+1})$ and let $\nabla t_k = t_{k+1} - t_k$, we already know:

$$W^\star(t) \le W^\star(t_k) + \nabla t_k N\Phi,$$
$$W(t) \ge W(t_k) - \nabla t_k N\Phi. \tag{22}$$

From (21) and (22), we have:

$$W^\star(t) - W(t) \le 2\nabla t_k N\Phi. \tag{23}$$

Substituting (23) in (13) and taking θ as large as $\theta_{\max}$ yields:

$$\begin{aligned}\Delta L(t) &\le \mathbb{E}[-\theta W^\star(t) + 2\nabla t_k N\Phi \mid \mathbf{Z}(t)] + \frac{1}{2}N\Phi^2 \\ &\le -\frac{\theta_{\max}}{SN}\sum_{s,n} Q_n^s(t) + \frac{2N\Phi}{\delta} + \frac{N\Phi^2}{2},\end{aligned} \tag{24}$$

where from **C1**, we know that $\mathbb{E}[\nabla t_k] \le 1/\delta$. Summing the above inequality for $t = 1$ to τ, we have:

$$\mathbb{E}[L(t+\tau) - L(t)|\mathbf{Z}(t)] \le -\frac{\theta_{\max}}{SN}\sum_{t=0}^{\tau-1}\sum_{s,n} Q_n^s(t) + \tau\left(\frac{2N\Phi}{\delta} + \frac{N\Phi^2}{2}\right).$$

We derive the following, which completes our proof after letting $\tau \to \infty$:

$$\frac{1}{\tau}\sum_{t=0}^{\tau-1}\sum_{s,n} Q_n^s(t) \le \frac{SN^2\Phi}{\theta_{\max}}\left(\frac{2}{\delta} + \frac{\Phi}{2}\right) + \frac{SN\mathbb{E}[L(\mathbf{Q}(0))]}{\tau\theta_{\max}}. \quad \square$$

References

[1] L. Tassiulas, A. Ephremides, Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks, IEEE Transactions on Automatic Control 37 (12) (1992) 1936–1948.
[2] M.J. Neely, E. Modiano, C. Li, Fairness and optimal stochastic control for heterogeneous networks, in: IEEE INFOCOM, 2005.
[3] M.J. Neely, E. Modiano, C.E. Rohrs, Dynamic power allocation and routing for time varying wireless networks, in: IEEE INFOCOM, 2003.
[4] G. Sharma, C. Joo, N.B. Shroff, Distributed scheduling schemes for throughput guarantees in wireless networks, in: 44rd Annual Allerton Conf. on Communications, Control, and Computing, 2006.
[5] G. Sharma, R.R. Mazumdar, N.B. Shroff, On the complexity of scheduling in multihop wireless systems, in: MOBICOM 2006, 2006.
[6] X. Wu, R. Srikant, Bounds on the capacity region of multihop wireless networks under distributed greedy scheduling, in: IEEE INFOCOM, 2006.
[7] P. Chaporkar, K. Kar, S. Sarkar, Throughput guarantees through maximal scheduling in wireless networks, in: 43rd Annual Allerton Conf. on Communications, Control, and Computing, 2005.
[8] C. Joo, N.B. Shroff, Performance of random access scheduling schemes in multi-hop wireless networks, in: IEEE INFOCOM, 2007.
[9] S. Sanghavi, L. Bui, R. Srikant, Distributed link scheduling with constant overhead, in: ACM Sigmetrics, 2007.
[10] Y. Yi, M. Chiang, Wireless scheduling with $O(1)$ complexity for M-hop interference model, in: ICC, 2008.
[11] K. Jung, D. Shah, Low delay scheduling in wireless network, in: ISIT, 2007.
[12] A. Eryilmaz, A. Ozdaglar, E. Modiano, Polynomial complexity algorithms for full utilization of multi-hop wireless networks, in: IEEE INFOCOM, 2007.
[13] E. Modiano, D. Shah, G. Zussman, Maximizing throughput in wireless networks via gossiping, in: ACM Sigmetrics, 2006.
[14] L. Tassiulas, Linear complexity algorithms for maximum throughput in radio networks and input queued switches, in: IEEE INFOCOM, 1998.
[15] X. Lin, N.B. Shroff, The impact of imperfect scheduling on cross-layer rate control in multihop wireless networks, in: IEEE INFOCOM, 2005.
[16] L. Bui, A. Eryilmaz, R. Srikant, X. Wu, Joint asynchronous congestion control and distributed scheduling for multi-hop wireless networks, in: IEEE INFOCOM, 2006.
[17] D. Shah, M. Kopikare, Delay bounds for approximate maximum weight matching algorithms for input queued switches, in: IEEE INFOCOM, 2002.
[18] E. Leonardi, M. Mellia, F. Neri, M.A. Marsan, Bounds on delays and queue lengths in input-queued cell switches, Journal of the ACM 50 (4) (2003) 520–550.
[19] G. Sharma, R.R. Mazumdar, N.B. Shroff, Joint congestion control and distributed scheduling for throughput guarantees in wireless networks, in: IEEE INFOCOM, 2006.
[20] X. Lin, S. Rasool, Constant-time distributed scheduling policies for ad hoc wireless networks, in: IEEE CDC, 2006.
[21] S.P. Meyn, R.L. Tweedie, Markov Chains and Stochastic Stability, Springer-Verlag, 1993.