MAC Scheduling with Low Overheads by Learning Neighborhood Contention Patterns

Yung Yi, *Member, IEEE,* Gustavo de Veciana, *Fellow, IEEE*, and Sanjay Shakkottai, *Member, IEEE*

***Abstract*—Aggregate traffic loads and topology in multi-hop wireless networks may vary slowly, permitting MAC protocols to 'learn' how to spatially coordinate and adapt contention patterns. Such an approach could reduce contention, leading to better throughput. To that end we propose a family of MAC scheduling algorithms and demonstrate general conditions, which, if satisfied, ensure lattice-rate-optimality (i.e., achieving any rate-point on a uniform discrete-lattice within the throughput-region). This general framework enables the design of MAC protocols which meet various objectives and conditions. In this paper, as instances of such a lattice-rate-optimal family, we propose distributed, synchronous contention-based scheduling algorithms that (i) are lattice-rate-optimal under both the SINR-based and graph-based interference models, (ii) do not require node location information, and (iii) only require three-stage RTS/CTS message exchanges for contention signaling. Thus, the protocols are amenable to simple implementation, and may be robust to network dynamics such as topology and load changes. Finally, we propose a heuristic, which also belongs to the proposed lattice-rate-optimal family of protocols and achieves faster convergence, leading to a better transient throughput.**

I. Introduction

Since the seminal work [2] on throughput maximization in wireless scheduling, there has been significant interest in distributed MAC scheduling algorithms with *provable* throughput-guarantees over wireless multi-hop networks. The problem is to find "throughput-optimal" algorithms — scheduling algorithms that stabilize the system whenever possible, subject to the constraints on the sets of simultaneously schedulable links. Since the centralized algorithm in [2], referred to as the Max-Weight algorithm, requires high complexity (typically, exponential complexity), distributed algorithms such as Maximal/Greedy scheme (e.g., [3]–[7]), Pick-and-Compare (e.g., [8], [9]), and queue-length based random access (e.g., [10], [11]) have been proposed, which stabilize networks for arrival rates within (the entire or some fixed fraction of) the throughput region with polynomial complexity. Such algorithms have been primarily analyzed in the context of graph-based interference models, and still rely on significant message passing to compute the schedule.

The research mentioned above has the following two aspects that need improvement and motivate our study. First, in wireless multi-hop networks with limited bandwidth, a simple signaling procedure is of great importance to ease implementation and reduce overheads. Second, in spite of extensive work assuming a graph-based interference model, it is more realistic to model interference relationships based on SINR (Signal-to-Interference-Noise-Ratio) that consider aggregate interference even from far-field transmissions, e.g., path-loss based interference with a minimum SIR requirement for successful packet decoding at receivers. However, there is little work on stabilizing algorithms (whenever possible) that both work under the physical interference model and require simple control signaling because of complex interference coupling over links and aggregate interference. Even if we admit that both interference models have their proponents, it would be useful to develop a unified scheduling framework, which allows the development of stabilizing distributed scheduling algorithms for both models and needs inexpensive signaling.

Most algorithms that we have discussed so far (e.g., Max-Weight and Pick-and-Compare, etc.) for ensuring throughput-optimality do so by means of scheduling a collection of links at each time-slot, and this collection of links is chosen based on the current queue-lengths. As the queue-lengths keep changing (even with stationary, ergodic traffic), these algorithms recompute the schedule periodically. To recompute the schedule (per-time-slot or periodically) requires the network to solve a global optimization problem which relies on heavy messaging.

We consider an alternate rate-based framework, which implicitly deals with traffic having a fixed long-term arrival rate. We assume that the each node knows its *local* offered load on its outgoing links. Given this "extra" information, our objective is to determine a schedule in a distributed manner that can support a "quantized" version of these rates if at all possible. Note that once a schedule has been determined, we can continue to reuse this schedule as long as the offered load does not change. Thus, our framework differs from queue-based scheduling, where a schedule has to be periodically recomputed even for statistically stationary traffic. We term such scheduling algorithms to be *lattice-rate-optimal,* meaning that the algorithm achieves any rate-point on a discrete-lattice (that can be chosen to be as fine as desired) within the throughput-region, given that nodes know their local offered-load on outgoing links (a more precise description is deferred to Section II-B).

Note that lattice-rate-optimality is a rate stability requirement and is thus weaker than queue stability. Lattice-rate-optimality only guarantees that the long-term departure rate is the same as the long-term arrival rate (as long as the quantized arrival rates lie on a discrete-lattice within the throughput-region), and does not guarantee ergodicity of queues (in the case of Markovian arrivals, positive recurrence of queues) in the system. Further, the notion of throughput-optimality in the

This research was supported by NSF Grants CNS-043507, CNS-0347400, CCF-0634898, and the DARPA ITMANET program. The work of G. de Veciana was supported in part by NSF Award CNS-0721532. A shorter version of this paper appeared in the Proceedings of IEEE Infocom 2007 [1].

context of queue stability [2] does not assume any knowledge of arrival rates and is thus different (and a stronger notion) from lattice-rate-optimality.

In this paper, we develop a lattice-rate-scheduling framework as described above, and propose its distributed instances for both the graph-based and physical interference models that: *(i)* are lattice-rate-optimal, *(ii)* do not require node location/explicit path-loss information, and *(iii)* have a very simple signaling mechanisms. Thus, they are amenable to simple implementation. The instance of the proposed family of lattice-rate-optimal algorithms, which we call RCAMA (Randomized Contention Aware Multiple Access), uses the simple multi-stage RTS/CTS exchanges. The signaling mechanism enables each node to 'learn' its neighborhood's contention patterns in an autonomous manner and possibly also adapt to changes in traffic load and network topology well. We are able to prove that with the additional local information at nodes (local offered loads) only a rate-stability requirement leads to a distributed, lattice-rate-optimal algorithm with only three stages of simple contention signaling per time-slot under the physical and graph interference models, respectively, irrespective of network size (thus, resulting in much less signaling than a queue-based scheduler such as Max-Weight).

Note that under the physical interference model, even *centralized scheduling* algorithms require knowledge of the exact topology (i.e., node locations, path-loss coefficients, and network connectivity) to achieve some form of (rate or throughput) optimality. This global information is needed to compute the amount of interference generated by simultaneously activated nodes, which in-turn is needed for computing a "good" schedule. However, somewhat surprisingly, our algorithm achieves lattice-rate-optimality *without* centralized geographical information. *To the best of our knowledge, this paper is the first to propose a distributed scheduling algorithm under the physical interference model that has any form of rate-optimality properties.*

In practice, depending on the types of services supported by the network, information on the offered load can either be explicitly given to the nodes or be measured by the nodes. If we have a guaranteed-service network based on a resource reservation signaling (e.g., RSVP [12]), the amount of load could be known a priori by nodes in the path of a reserved flow. However, in a typical best-effort service network, the amount of load is not explicitly provided to the nodes, but the nodes could measure/estimate offered load over a suitable time-period. The main motivation for RCAMA is that although individual (end-to-end) traffic loads may change quickly, the aggregates on some congested links may, in many relevant applications, change more slowly and locally. Similarly, node mobility (that leads to changes in topology and load) might be slow enough to permit a MAC scheduler to learn and exploit the offered traffic characteristics so as to quickly realize "good" schedules. Because the loads may exhibit some variation, or measurements may be noisy, a node may use an upper estimate for it.

Breaking the conventional belief that highly complex signaling (whether it is polynomial or exponential) is unavoidable for some form of rate-optimality does not come free – this requires some costs to be paid in our case, such as need of local knowledge on average load, convergence time, and granularity of supported load. We will also discuss how such performance metrics scale as we vary the system parameters to evaluate these costs. As mentioned earlier, our scheduling framework and the proposed algorithms can be applied to both physical and graph interference models with slight modifications. However, scheduling problems are generally more challenging in the physical interference model. Thus, we restrict our attention to the physical model in this paper, and we refer the readers to our technical report [13] for graph based interference models and other details.

The main contributions and organization of this paper are as follows:

1) In Section II, we describe the system model.
2) In Section III, we first propose a scheduling framework (DRS: Dynamic Randomized Scheduling), that provably achieves any rate-point on a uniform discrete-lattice within the throughput region (i.e., lattice-rate-optimal). To that end, we give two key conditions in the DRS family, which, if satisfied, ensure that an algorithm in the family is lattice-rate-optimal. We further study their rate of convergence to optimality.
3) In Sections IV and V, as an instance of the DRS family, we propose a synchronous contention-based algorithm, RCAMA (Randomized Contention-Aware Multiple Access), where *multi-stage contention signaling* in conjunction with *randomized time-slot selection* is used. We prove lattice-rate-optimality of RCAMA, by showing that RCAMA satisfies the two conditions in 2).
4) In Section VI, we propose an adaptive variation of RCAMA, ARCAMA (Adaptive RCAMA) for the purpose of better adaptation to load/topology changes and faster convergence to optimality. ARCAMA again satisfies the two conditions in 2) and adaptively biases slot selection probabilities based on the past contention histories. In Section VII, we show via simulation that only a short duration of memory is required to increase performance, resulting in good adaptation to load/topology changes.
5) In Section VIII, we review the related work, and conclude the paper.

II. System Model

A. Network and Traffic Model

We assume that time is slotted. A time-slot duration is suitably chosen to accommodate the transmission of one fixed-size packet. We model the wireless multi-hop network by a graph $G(L, V)$, where L and V denote a set of directed links and nodes, respectively. We abuse the notation L and V to also refer to the number of links and nodes, respectively. We assume that for any link between two nodes there is a counterpart in the opposite direction. We denote a directed link from node i to node j by $i \rightarrow j$. For concreteness, the wireless system under study has a *single* frequency/code and each node is time-synchronized and has a half-duplex radio.

We assume a fixed power model, where a transmitter uses the power P for data transmission, and SINR (Signal-to-

Interference-Noise Radio) is considered to determine success or failure of a transmission.

A message from i to j is *decodable,* if

$$\frac{G_{ij}P}{\eta_j + \sum_{k\in V_I(i)} G_{kj}P} \quad \geq \quad \gamma, \tag{1}$$

where γ is the minimum SINR threshold required for message decoding, $V_I(i)$ is the set of nodes transmitting simultaneously with i (thus interfering the transmission over the link $i\rightarrow j$) on a given time-slot, G_{ij} is the propagation loss from i to j, and η_j is the thermal noise power at j. The SINR threshold γ depends on the desired bit rate, bit error rate, and design parameters such as modulation, coding, and so on.

In practice, in addition to interference, wireless links are prone to errors due to many other factors (e.g., fading). This leads to high packet loss rate detrimental to upper-layer performance. Thus, in many MAC protocols, reliability is provided by acknowledging transmissions and possibly retransmitting. Thus, we say that a transmission over $i\rightarrow j$ is *successful*, if both the *data* message from i to j and the corresponding *ack* message from j to i are decodable at j and i, respectively, where the ack message from j will be sent only when the data message is decodable at i.

A *link schedule*, or simply a schedule $\boldsymbol{A} = (A_l \in \{0,1\} : l = 1,\ldots,L)$, is a 0-1 vector representing the set of scheduled links, where $A_l = 1$ if the link l is scheduled for attempted transmission, and 0 otherwise. A link schedule $\boldsymbol{A}$ is said to be *successful*, if all transmissions scheduled by $\boldsymbol{A}$ are successful. We denote the collection of all *successful* schedules by $\mathcal{A}$. A *scheduling algorithm* chooses a sequence of link schedules (which are not necessarily successful), $(\boldsymbol{A}[s] : s = 0,1,\ldots)$, where $\boldsymbol{A}[s]$ is the link schedule on time-slot s. We denote by $D_l[t]$ the number of arrivals over the link l with its mean $\mathbb{E}[D_l[t]] = \rho_l$. The vector $\boldsymbol{\rho} = (\rho_l : l \in L)$ is said to be the *load* in the system. As mentioned earlier, we assume that each link knows only its own load.

B. Performance Metric: Lattice-Rate-Optimality

Our objective is to develop scheduling algorithms that achieve lattice-rate-optimality, i.e., rate-stabilize the system whenever possible for any load which can be quantized to lie on a discrete-lattice within the throughput region where nodes have knowledge of their local, long-term arrival rates on outgoing links (more precise definition will follow shortly).

The throughput region Λ is defined by the set of all loads $\boldsymbol{\rho} = (\rho_l : l \in L)$ that are stabilized by some scheduling algorithm, where stability means that the long-term departure rate per queue is the same as its long-term arrival rate, i.e., rate stability. Then, it is well-known that the throughput region can be characterized by the following (see e.g., [2])[1]:

$$\Lambda \quad = \quad \left\{ \boldsymbol{B} \mid \boldsymbol{B} = \sum_{\boldsymbol{A}_i\in\mathcal{A}} \beta_i \boldsymbol{A}_i, \quad 0 \leq \beta_i \leq 1, \quad \sum_{i=1}^{|\mathcal{A}|} \beta_i \leq 1 \right\},$$

where $\boldsymbol{A}_i$ is the i-th schedule in $\mathcal{A}$.

[1]When queue stability is considered, then the throughput region is the interior of Λ.

We define a weaker notion of throughput region Λ_F that is the *lattice-sampling of* Λ with adjacent points having distance $1/F$ for some positive integer F:

$$\Lambda_F \;=\; \left\{ \boldsymbol{B} \,|\, \boldsymbol{B} = \sum_{\boldsymbol{A}_i\in\mathcal{A}} \beta_i \boldsymbol{A}_i,\, \beta_i = \frac{k_i}{F},\, \sum_{i=1}^{|\mathcal{A}|} k_i \leq F,\, k_i \in \{0,1,\ldots,F\} \right\}.$$

Note that $\Lambda = \mathsf{CL}(\cup_{F=1,\ldots,\infty}\Lambda_F)$, where $\mathsf{CL}(Z)$ is the closure of a set Z.

For a given F, the load $\boldsymbol{\rho}$ is said to be *F-lattice-feasible* if $\boldsymbol{\rho} \in \Lambda_F$. A scheduling algorithm is said to be *F-lattice-rate-optimal*[2] if it stabilizes the system for any F-lattice-feasible load, where each node has local knowledge of the long-term (aggregate) arrival rates on its outgoing links. For a F-lattice-feasible load $\boldsymbol{\rho} \in \Lambda_F$, let $\boldsymbol{\theta} = \boldsymbol{\rho}F$. Then, the vector $\boldsymbol{\theta} \in \mathcal{Z}_+^L$ can also be used to represent load measured over F slots. Henceforth a group of F time-slots is said to be a *frame* throughout this paper, and we use $\boldsymbol{\rho}$ and $\boldsymbol{\theta}$ interchangeably unless it generates confusion.

III. Dynamic Randomized Scheduling

A. Frame-Based Scheduling

We consider "frame-based" scheduling algorithms, where scheduling patterns are determined on a frame-by-frame basis (i.e., F time-slots)[3]. Thus, frame-based scheduling considers a *frame schedule* (FS) that is a consecutive sequence of F link schedules, represented by a $L \times F$ matrix with 0-1 elements, $C = [c_{ls}]_{l\in L, s\in F}$ with $\sum_{s=1}^{F} c_{ls} = \theta_l$ for a given load $\boldsymbol{\theta}$ (i.e., the link load determine the number of accessed slots). The l-th row vector $c_{l\cdot}$ of $C = [c_{ls}]$ is said to be a *slot schedule* over l in a frame. Recall that a column vector $c_{\cdot s}$ is a link schedule on slot s. For a given load $\boldsymbol{\theta}$, a frame schedule is said to be *feasible*, if all of F link schedules (column vectors) are successful for all $l \in L$. It is clear that $\boldsymbol{\theta}$ is F-lattice-feasible, if and only if there exists a feasible frame schedule $C = [c_{ls}]$. For notational simplicity, we remove dependency of C on F and $\boldsymbol{\theta}$, unless explicitly needed. Also, throughout this paper we implicitly assume that the lattice-parameter is fixed by F.

In our framework, F is a system-wide parameter that is assumed to be known to every node in the network a-priori. Recall that we assume that a node has knowledge only of the local load (i.e., mean arrival rate) on each of its outgoing links. Under such an assumption, we aim at designing frame-based algorithms (hopefully having small control overhead) that find a feasible frame schedule within a finite time, and sustains the schedule thereafter, for any given feasible load.

It can be easily shown that such a frame-based scheduling is F-lattice-rate-optimal, since after finding a feasible frame schedule within a finite time, each link can serve the packets successfully at least with the same rate as the input arrival rate, i.e., achieves lattice-rate-stability. By serving the network with the (distributedly determined) feasible frame schedule for a sufficiently large amount of time, the system can stabilize the offered load.

[2]For simplicity, we just use "rate-optimal" and "feasible load" to refer to "F-lattice-rate-optimal" and "F-lattice-feasible", unless explicitly needed.

[3]Thus, we henceforth use a term 'time-slot s' to refer to the s-th time-slot inside a frame. We typically use 's' and 't' to index a slot and a frame.

Different from a large class of queue-based scheduling algorithms mentioned in Section I, queue length information does not have to be exchanged in our frame based scheduling. Achieving lattice-rate-optimality is possible, because we have local knowledge of (mean) arrival rates, whereas queue-based scheduling is oblivious of the arrival statistics. However, queue-based scheduling typically requires messaging with high complexity, which is the cost of unawareness of arrival rate. In addition to the cost of using the arrival rates, our scheme guarantees only lattice-rate-optimality instead of the "true" throughput region Λ.

B. Dynamic Randomized Scheduling

Now, out of many possible frame-based algorithms, we focus on a class of randomized algorithms that choose FSs (frame schedules) in a randomized manner possibly based on the past histories, e.g., transmission success/failures in the past frames. Randomization has potential advantages in a sense that a simple *random access*-type of algorithms can be modified and utilized in our framework.

Algorithms may be able to define and use additional system control states. As an example that is particularly useful in our paper, we define transmission priority matrix as follows: for a given frame schedule C, we define a *transmission priority,* $R = [r_{ls}]_{l\in L, s\in F}$, where

$$r_{ls} = \begin{cases} 1 & \text{if } c_{ls} = 1 \text{ and high priority,} \\ 0 & \text{if } c_{ls} = 1 \text{ and low priority,} \\ \text{NULL} & \text{if } c_{ls} = 0. \end{cases}$$

As seen in Section IV, this priority state can be used to give preferential chances to some links in performing contention signaling.

To measure the "distance" between two FSs $C = [c_{ls}]$ and $C' = [c'_{ls}]$ (under the same topology and load), we define a distance function d to be the total number of different elements, i.e.,

$$d(C, C') \triangleq \sum_{l=1}^{L} \theta_l - \sum_{l=1}^{L}\sum_{s=1}^{F} c_{ls}c'_{ls}. \tag{2}$$

Recall that a frame schedule is a 0-1 matrix. Note that $d(C, C') = 0$ implies $C = C'$.

Now, we formally define a family of randomized algorithms satisfying two conditions, which provably guarantees lattice-rate-optimality.

Definition 3.1: A dynamic randomized scheduling (DRS) algorithm randomly chooses a sequence of frame schedules and priority matrices $(C[t], R[t] : t = 0, 1, \ldots)$, and satisfies the following two conditions:

C1 *Finite Sustenance Condition.* If $C[i]$ is feasible, $C[t] = C[i]$, $\forall t > i$, with probability 1.

C2 *Finite Improvement Condition.* If $C[i]$ is not feasible, for any feasible FS $C^\star$, there exists a t (which may depend on $C^\star$) with $i < t < \infty$, such that $d(C[i], C^\star) > d(C[t], C^\star)$ with positive probability.

The finite sustenance condition means that if a FS converges to a feasible one, it has to be sustained thereafter. The finite improvement condition is such that before converging to a feasible FS, a sequence of FSs over frames tend to become "closer" to a feasible FS with positive probability. As mentioned earlier, a state $(C[t], R[t])$ at frame t may depend on the states of the previous, say m, frames. In this case we say that a DRS algorithm has *history* m. Note that in a DRS algorithm without priority, $R[t]$ is not in use.

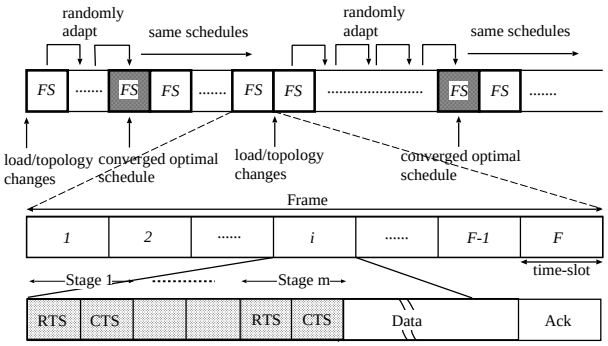


Fig. 1. Frame and slot structure of RCAMA

Theorem 3.1: For any fixed feasible load and topology, any DRS algorithm satisfies the following:

(i) It converges to a feasible FS within a finite time.

(ii) Let $\tau(C)$ be the convergence time to a feasible FS for a given initial frame schedule C. Then, $\forall t \in \mathcal{Z}_+$, there exist constants $0 < K < \infty$ and $0 < p < 1$, such that $\mathbb{P}[\tau(C) \geq tK] \leq p^t$.

For notational simplicity, we remove dependence of τ, K, and p on the considered scheduling. Recall that Theorem 3.1(i) implies lattice-rate-optimality. The proof is presented in Appendix.

The finite sustenance and improvement conditions described above enable us to verify rate-optimality of an instance of the DRS family. In addition, it allows customization or enhancement of an algorithm with its rate-optimality maintained, as long as the modified version satisfies those conditions. In Sections IVand V, we develop a "base-line" DRS algorithm—RCAMA—with history 1. We later discuss how such base-line algorithms with history 1 can be extended to adaptive versions with multiple frame histories for better adaptation to load/topology changes and faster convergence to the optimal schedule in Section VI. Our interests also lie in how the constants K and p scale in RCAMA, which we will also discuss later.

IV. RCAMA: Overview and Per-frame Operation

A. Overview

The general frame and time-slot structure of RCAMA are shown in Figure 1. A time-slot is divided into two parts: time for contention signaling and time for data and ack transmission (TR)[4]. We will describe RCAMA by dividing its behavior into two different time-scales: *(i) per-frame operation*, where each node randomly determines the slot-schedules for the TRs

[4]For notational simplicity, we use the term 'TR' to refer to the word 'transmission' throughout this paper.

over its adjacent outgoing links, and *(ii) per-slot operation*, where a node initiates a RTS/CTS-like contention signaling to resolve contentions and implicitly *learn* contention patterns in the neighborhood. In this section, we describe only per-frame operation, and per-slot operations will be discussed in Sections V. We first provide an overview of RCAMA.

The RCAMA is designed to ensure the following two properties:

1) **Persistence:** A successful TR at a given time-slot at the current frame persists on the same slot at the next frame.
2) **Preemption:** An unsuccessful TR can preempt a time-slot (with positive probability) used by a persistent successful TR.

As discussed earlier, it suffices to show that the system *converges* to a feasible FS to achieve rate-optimality. By the persistence property, once the system reaches a "good" (i.e., feasible) FS, it stays in that FS. Preemption property ensures that there is no deterministic "winner-loser" relationships among TRs, and enables the system to avoid deadlocks, i.e., being stuck in a "bad" FS. These two properties ensure that the system has positive probability of visiting arbitrary FSs, and finally reach a feasible FS, which is sustained thereafter. We satisfy these two properties by assigning just *two-level priorities* to scheduled TRs. More specifically, by assigning high priority to unsuccessful TRs and low priority to persistent successful TRs, respectively, we allow a newly scheduled unsuccessful TR on a time-slot to beat existing successful ones (which happens before convergence to a feasible FS). Intuitively, the persistent and preemption properties connect to finite sustenance and improvement conditions, respectively. We will see these connections in the formal proof of rate-optimality of RCAMA, described in Theorem 5.1.

In addition to provable rate-optimality, by using a low-complexity contention signaling, the algorithm can quickly adapt to load and topology changes by "learning" local contention patterns. In other words, RCAMA does not need any explicit mechanism to inform the nodes of such network changes, and it automatically avoids the situation where multiple time-slots are commonly accessed by interfering links. Further, application of non-uniform time-slot access probability for unsuccessful TRs enable the system to learn local contention levels, and to distribute scheduled TRs at different time-slots in a more efficient manner (see Section VI).

B. Per-Frame Operation: Randomized Slot-Selection

When each frame starts, each node (say, $v \in V$) determines the slot-schedules and contention priorities for the TRs over its adjacent outgoing links. To do this, the following simple rule is used:

Rule 4.1 (Slot and Priority Selection Rule):

(i) A successful TR on time-slot s at frame $t-1$ persists on the same time-slot s at frame t, with priority set to be *low.*

(ii) If a TR was unsuccessful at frame $t-1$, a time-slot is randomly selected from the time-slots not already taken in (i), and its priority is set to be *high.*

Rule 4.1(i) corresponds to the persistence property. Preemption property is satisfied by Rule 4.1(ii) in conjunction with the proposed multi-stage signaling (corresponding to per-slot operation) in Section V.

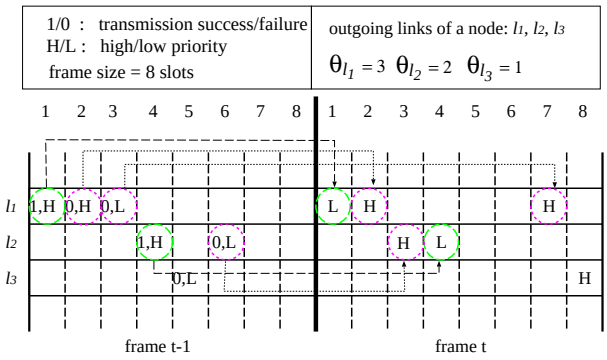


Fig. 2. Example of randomized slot-selection

An example of Rule 4.1 is shown in Figure 2. Since at frame $t-1$, the TR over l_1 on time-slot '1' and over l_2 on time-slot '4' were successful, these TRs are scheduled once again with *low* contention priority at the same time-slot positions at frame t. For the unsuccessful TRs over l_1 on time-slots '2' and '3', we randomly choose two time-slots of the remaining time-slots, which were not taken by previously successful TRs (i.e., the node does not consider time-slots '1' and '4' in this random selection). In the example, time-slot '2' and '7' are selected, and they are scheduled with *high* contention priority.

We first observe that Rule 4.1 immediately satisfies the following Property 4.1:

Property 4.1: For any time-slot s, and link l, there exists a positive probability that $c_{ls}[t-1] = c_{ls}[t]$, irrespective of $c_{l's'}[t-1], l' \neq l, s' \neq s$.

The above property holds, because any TR scheduled at some slot s could be re-scheduled at the same slot s with positive probability, whether it was successful or not at the previous frame. Especially, for a successful TR, probability that the same time-slot is chosen is '1' from Rule 4.1(i), whereas for an unsuccessful TR, the probability is at least $1/F$ from Rule 4.1(ii). We will use this property in the proof of Theorem 5.1 on rate-optimality of RCAMA in Section V-A.

V. Per-Slot Operation

Following the slot-schedules as explained in Section IV-B, this section explains how, on each slot, nodes use RTS/CTS-based contention signaling to resolve contentions (which is not perfect sometimes), followed by data/ack TRs.

A. Per-Slot Operation: Three-Stage Signaling

The signaling consists of three stages, at each of which a different set of links perform contention signaling. The objective of signaling is to determine the links over which data transmissions are *attempted.* However, note that since success of data transmissions are decided based on the received SINR, we *cannot guarantee success of all attempted data*

transmissions. Our aim is to design multi-stage signaling, such that the preemption property is met, and then together with the persistence property given by randomized slot selection in Section IV-B, the system achieves rate-optimality. The algorithm for contention signaling is conceptually and pictorially described in Figure 3.

Stage 1 Signaling for the links in $\mathcal{H}$
Stage 2 Signaling for the links in $\mathcal{H}_V^1$ and $\mathcal{M}$
Stage 3 Signaling for the links in $\mathcal{H}_V^1$ (with power adjustment for the links in $\mathcal{H}_I^2$) and $\mathcal{M}_V^2$
Data/Ack Data and Ack Transmissions for $\mathcal{H}_V^1$ and $\mathcal{M}_V^3$

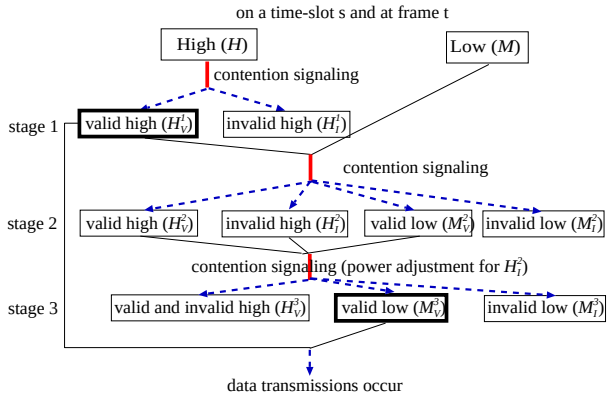


Fig. 3. Per-slot operation: three-stage signaling

Prior to being elaborate on discussion of the per-slot operation, we first define *validity* of a signaling: If a signaling over some link, say $i \rightarrow j$, is said to be *valid*, if j decodes the RTS from i, and i decodes the CTS (in response to RTS) from j. Clearly, validity of a signaling over a link depends on signalings over other links performed simultaneously at the corresponding slot. Note that the validity of a signaling over link l does not necessarily imply the success of TR over the link, because success or failure is determined by the result of data transmission, not that of signaling.

For ease of exposition, we first introduce some notations here. We denote by $\mathcal{H}$ (resp. $\mathcal{M}$) a set of links scheduled (on a given time-slot) with high (resp. low) priority, where $\mathcal{H}, \mathcal{M} \subset L$. At each stage, contention signaling is conducted for high and/or low priority TRs. We use the notations $\mathcal{H}_V^i$ and $\mathcal{H}_I^i$ to refer to valid and invalid high priority TRs at stage i, respectively. Similarly, $\mathcal{M}_V^i$ and $\mathcal{M}_I^i$ are used for low priority TRs. Now, in what follows, we explain the behaviors of all stages in more detail.

Stage 1: Signaling is performed over only high priority TRs, i.e., $\mathcal{H}$, based on which $\mathcal{H}_V^1$ and $\mathcal{H}_I^1$ are determined. Note that $\mathcal{H}_V^1 \cup \mathcal{H}_I^1 = \mathcal{H}$. In our three-stage signaling, *whatever happens at the subsequent Stages 2 and 3, data TRs are attempted for* $\mathcal{H}_V^1$. However, their success is not guaranteed, because some of low priority TRs in $\mathcal{M}$ (chosen following the results of signaling procedures at the subsequent Stages 2 and 3) will also perform data transmission, as we will explain shortly.

We will later show that *for rate-optimality, it suffices to guarantee the success of all TRs in* $\mathcal{H}_V^1$ (see Theorem 5.1). It is highly related to the preemption property in DRS scheduling algorithms. Thus, the objective of subsequent stages 2 and 3 is to ensure the success of TRs in $\mathcal{H}_V^1$.

Stage 2: Signaling is performed for the TRs in $\mathcal{H}_V^1$ and $\mathcal{M}$, after which $\mathcal{H}_V^2$, $\mathcal{H}_I^2$ $\mathcal{M}_V^2$, and $\mathcal{M}_I^2$ are determined. Note that $\mathcal{H}_V^2 \cup \mathcal{H}_I^2 = \mathcal{H}_V^1$, and $\mathcal{M}_V^2 \cup \mathcal{M}_I^2 = \mathcal{M}$. The role of this stage is to identity TRs in $\mathcal{H}_V^1$ that are "severely" interfered by *low priority* TRs, i.e., identify $\mathcal{H}_I^2$. Then, remarking our signaling objective mentioned at Stage 1 of ensuring success of TRs in $\mathcal{H}_V^1$, Stage 3 will be devoted to "kill" low priority TRs in $\mathcal{M}$ that significantly interfere with TRs in $\mathcal{H}_I^2$. Note that we do not need to care about TRs in $\mathcal{M}_I^2$, because they are already invalidated, and "killed" at this Stage 2.

Stage 3: Now, it is a turn to test whether there exist TRs only in $\mathcal{M}_V^2$ that significantly interfere with TRs in $\mathcal{H}_V^1$. Thus, signaling is performed for the TRs in $\mathcal{H}_V^1$ and only $\mathcal{M}_V^2$. The objective of Stage 3 is to invalidate low priority TRs, which can cause the TRs in $\mathcal{H}_I^2$ to fail (note that TRs in $\mathcal{H}_V^2$ will be successful even with interference by low priority TRs).

However, as we exemplify in Figure 4 shortly, there exists some scenario that when the given constant power P is used (see Section II), low priority TRs cannot be invalidated whatsoever. To handle such cases, we allow only one exception in Stage 3, and employ *signaling power adjustment* in RTS/CTS signaling for TRs of $\mathcal{H}_I^2$, i.e., the transmitters and the receivers in $\mathcal{H}_I^2$ adjust (and actually increase in most cases) their signaling powers appropriately, such that some interfering low priority TRs in $\mathcal{M}_V^2$ are invalidated. In Section V-A, we will explain how we adjust powers for the signaling at this stage, such that rate-optimality is guaranteed.

Data/ack TRs: Data TRs occur for TRs in $\mathcal{H}_V^1$ and TRs in $\mathcal{M}_V^3$. ACK messages are sent back to the transmitters by the receivers which can decode transmitted data.

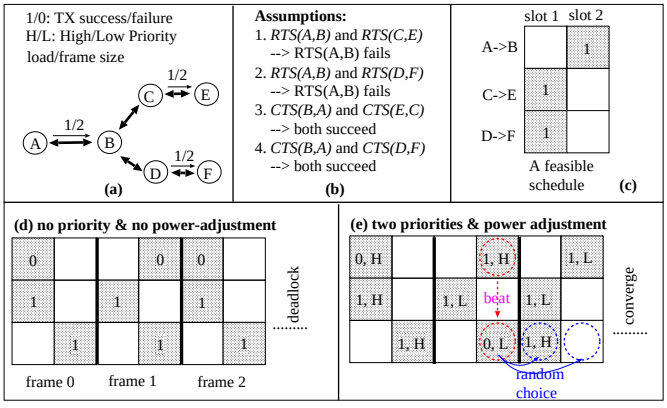


Fig. 4. Example of RCAMA

We exemplify the three-stage contention signaling in conjunction with randomized slot selection in RCAMA in Figure 4, where we can intuitively understand need of priority and convergence to a feasible FS. In absence of contention priority and signaling power adjustment, the TR over A→B keeps

failing with either choice of time-slot '1' or '2', since RTS from A is not decodable at B due to interference from either C or D, over frames. However, in RCAMA, from Rule 4.1, the unsuccessful TR over A→B at frame '0' is assigned high priority at frame '1,' and due to stages 2 and 3, B adjusts the power for its CTS (destined to A and broadcast to D), such that CTS from F is not decodable at D (see the frame 1 in (e)). The same procedure can be applied when TRs over A→B and C→E are assigned high and low on a same time-slot, respectively. By this procedure, the system ultimately converges to a feasible FS.

Now, Theorem 5.1 states that we can indeed achieve rate-optimality if on all slots TRs in $\mathcal{H}_V^1$ are guaranteed to be successful, which we call *High Priority Condition (HPC)*.

Theorem 5.1: For any given fixed topology and feasible load, RCAMA satisfying HPC is a DRS scheduling. Thus, by Theorem 3.1, it is rate-optimal. Further, the bounds on the constants K and p in Theorem 3.1 are given by:

$$K \leq 2n_\theta, \quad \text{and} \quad p \leq 1 - (1/F)^{2n_\theta^2},$$

where n_θ is the total load in the network, i.e., $n_\theta = \sum_{l \in L} \theta_l$.

The proof is long and is postponed to Appendix. It basically proceeds by proving that finite sustenance and improvement conditions are satisfied. Intuitively, by guaranteeing TRs in $\mathcal{H}_V^1$, preemption property is ensured, which in turn implies finite improvement condition. As we can see in the above, as F increases, i.e., granularity of load representation increases (meaning that our lattice-throughput-region becomes finer), the convergence time becomes slower. Also, as the network size grows with very high load, n_θ increases, again resulting in slower convergence. Note that the bounds representing the worst-case convergence time to a feasible FS given above is *pessimistic*. Thus, the actual convergence may occur faster.

As a next step, it remains to design signaling power adjustment at Stage 3, such that the HPC is satisfied.

B. RCAMA-MAX

Theorem 5.1 enables us to develop the following simple, distributed rate-optimal algorithm:

RCAMA-MAX: All the adjusted powers in the signaling of Stage 3, conducted by TRs in $\mathcal{H}_I^2$, is set to be $P_{\max}$, where $P_{\max}$ is the amount of signaling power, such that it can invalidate any other signaling.

The assumption that $P_{\max}$ exists is reasonable for wireless multi-hop networks deployed of a finite size. Then, the following proposition immediately follows:

Proposition 5.1 (RCAMA-MAX): For any fixed topology and feasible load, RCAMA-MAX satisfies HPC, and thus is rate-optimal from Theorem 5.1.

The proof is obvious, since at Stage 3, with $P_{\max}$, all the low priority TRs become invalidated. TRs in $\mathcal{H}_V^1$ may be invalidated at Stage 3, but note that data transmission attempts for TRs in $\mathcal{H}_V^1$ are independent of the signaling results at Stage 3.

Remark 5.1: Note that under the physical interference model, even a centralized algorithm needs information on node locations and network connectivity to achieve rate-optimality. Surprisingly, however, Proposition 5.1 implies that there exists a distributed rate-optimal scheduling algorithm that does not need such centralized topology information.

In spite of the provable rate-optimality and the fully distributed nature of RCAMA-MAX, it may not be a practical algorithm, since for a large-scale multi-hop network, $P_{\max}$ should be very large. This is not a desirable feature leading to low efficiency of energy utilization and poor transient throughput. In other words, with RCAMA-MAX, every low priority TRs will fail at all slots, and only high priority TRs surviving Stage 1 will succeed, which happens before convergence to a feasible FS. The main observation behind this limitation of RCAMA-MAX is that we need to consider the "worst-case," i.e., the case when a large number of far field low priority TRs interfere with a high priority TR (which was valid at Stage 1). However, it is known that interference is dominated by a small number of nearby transmissions mainly due to non-linear signal power loss. Using this observation, in the next section, we propose an enhanced algorithm, RCAMA-VIR, which uses far lower powers than $P_{\max}$, but still guarantees rate-optimality under reasonable assumptions.

C. RCAMA-VIR

The main idea in RCAMA-VIR is to use a sufficiently high power (but not as large as $P_{\max}$ in Stage 3 signaling), such that low priority interferers of $\mathcal{H}_I^2$ can be suppressed. This is done by estimating (and developing bounds) on the interference power.

In this section, we assume the following: *(i)* A receiver can only measure the total received signal power (the desired signal power plus interference) and know a boolean result about the target SINR (i.e., the target SINR is larger than the threshold γ or not)[5]; *(ii)* the propagation loss is modeled by $G_{ij} = 1/d(i,j)^{\alpha(i,j)}$, where $d(i,j)$ is the distance between nodes i and j, and $\alpha(i,j)$ is an "effective" path loss exponent (which may depend on the node-pair), for which each node knows (lower and upper) bounds (i.e., $\underline{\alpha} \leq \alpha(i,j) \leq \bar{\alpha}$); *(iii)* the system is interference-limited[6].

The transmitter $s(l)$ and the receiver $d(l)$ of link $l \in \mathcal{H}_I^2$ perform the following procedures:

RCAMA-VIR:

1) $d(l)$ ($s(l)$) estimates the aggregate interference generated by low priority TRs during RTS (CTS) slot, and assumes that such interference is caused by the transmitter (receiver) of a single *virtual* low priority TR. (see Section V-D for discussion on estimation of the aggregate interference).
2) $d(l)$ ($s(l)$) computes an upper-bound on the distance to the transmitter (the receiver) of the virtual TR. This

[5]Note that we do not assume that the receiver is able to know the exact SINR value as well as individual or even aggregate pure interference generated by other transmissions.

[6]In this system, the link operates at a sufficiently high γ (SINR threshold), so that the effect of thermal noise is negligible as compared to the interference. However, this can be readily extended to the more general assumption that $0 \leq \eta_j \leq \epsilon \times$ (interference), where ϵ is the ratio of thermal noise to the total interference.

upper-bound is computed based on the bounds on the path loss exponent (i.e., $\underline{\alpha} \le \alpha \le \bar{\alpha}$), and the interference estimation in (i).

3) By assuming that there is no power path-loss between the virtual transmitter and receiver, $d(l)$ ($s(l)$) computes the adjusted CTS (RTS) power, required to invalidate the virtual TR.

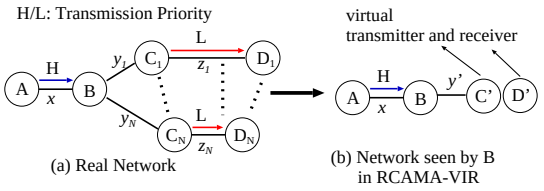


Fig. 5. Example of RCAMA-VIR

An example of RCAMA-VIR is shown in Figure 5. We have one high priority and N low priority TRs scheduled on a same time-slot. The high priority TR is clearly valid at Stage 1. At Stage 2, suppose that at Stage 2 an RTS over A→B is not decodable due to the aggregate interference of RTSs from $\mathtt{C}_i$ to $\mathtt{D}_i$, $i = 1, \ldots, N$. Now, B assumes that its RTS decoding failure is due to a single virtual low priority TR. By estimating such aggregate interference, B computes the distance from itself to C′ (the virtual transmitter). In the CTS-slot of Stage 3, B sets the sufficiently large CTS power to invalidate a CTS from D′ (the virtual receiver of C′), based on the "worst-case" assumption that there does not exist a signal power path-loss between C′ and D′.

Note that RCAMA-VIR may not be rate-optimal, when many far field low priority transmissions are interfering a high priority TR. However, we will show that RCAMA-VIR achieves rate-optimality under reasonable assumptions (see Theorem 5.2).

D. Estimation of Interference

Note that the major difference between Stages 1 and 2 is the existence of low priority TRs. Thus, it is intuitive to use measurement of the total received signal powers at Stages 1 and 2 and using their differences to estimate the interference by low priority TRs.

Consider a TR $l \in \mathcal{H}_I^2$. We denote by $\hat{R}^1_{d(l)}$ (resp. $\hat{C}^1_{s(l)}$), the total received signal power on RTS (resp. CTS) slots at Stage 1 by $d(l)$ (resp. $s(l)$). Similarly, we use the notations $\hat{R}^2_{d(l)}$ and $\hat{C}^2_{s(l)}$, at Stage 2. We also let $I^r_{d(l)}$ and $I^c_{s(l)}$ be the *exact aggregate low priority interference* to $d(l)$ and $s(l)$. To estimate the interference by low priority transmitters and receivers, we use the values defined in the following: $\hat{I}^r_{d(l)} \triangleq \hat{R}^2_{d(l)} - \hat{R}^1_{d(l)}$ and similarly, $\hat{I}^c_{s(l)} \triangleq \hat{C}^2_{s(l)} - \hat{C}^1_{s(l)}$.

Using the above method for estimation, we have

$$\hat{I}^r_{d(l)} \le I^r_{d(l)}, \qquad \hat{I}^c_{s(l)} \le I^c_{s(l)}, \tag{3}$$

since we have

$$\begin{aligned} \hat{I}^r_{d(l)} &= \hat{R}^2_l - \hat{R}^1_l \\ &= \hat{R}^2_l(\mathcal{H}^1_V) + \hat{R}^2_l(\mathcal{M}) - (\hat{R}^2_l(\mathcal{H}^1_V) + \hat{R}^2_l(\mathcal{H}^1_I)) \\ &= \hat{R}^2_l(\mathcal{M}) - \hat{R}^2_l(\mathcal{H}^1_I) \le I^r_{d(l)} \ (\because \hat{R}^2_l(\mathcal{M}) = I^r_{d(l)}), \end{aligned} \tag{4}$$

where $\hat{R}^i_l(\mathcal{A})$ corresponds to the total received power by a receiver of link l from RTS TRs in some set of links $\mathcal{A}$ during stage i. Similarly, we also have that $\hat{I}^c_{s(l)} \le I^c_{s(l)}$.

In other words, our estimation is a lower-bound on the exact interference by low priority TRs. This lower-bound in the interference estimation and the bounds on the path loss exponent lead to an upper-bound on the distance to the transmitter/receiver of the virtual TR, which is used in the proof of rate-optimality of RCAMA-VIR.

Theorem 5.2 (RCAMA-VIR): Suppose that there exists a maximum distance of interference between nodes and a maximum number of interferers, denoted by d_{int} and N_{int}, respectively. If $\sqrt[2\underline{\alpha}]{N_{\text{int}}}(d_{\text{int}})^{\bar{\alpha}/(2\underline{\alpha})} \le d_{\min}$, where $d_{\min}$ is the minimum distance between two nodes, then RCAMA-VIR satisfies HPC. Thus, it is rate-optimal from Theorem 5.1.

Theorem 5.2 implies that if the inter-node distance is sufficiently large, i.e., node density in a plane is not too high and nodes are distributed in a sufficiently uniform manner, rate-optimality is provably guaranteed in RCAMA-VIR. The proof is presented in Appendix.

Numerical Example 5.1: As a numerical example, consider the case when $d_{\text{int}} = 2 \times d_{\min}$ (a typical setting in the IEEE 802.11 DCF by assuming that transmission rage is set to be $d_{\min}$) for different values of bounds on path-loss exponents and N_{int}, given by:

$$\begin{aligned} d_{\min} &\ge 2.5 \text{ m} \quad \text{if} \quad \bar{\alpha} = \underline{\alpha} = 3, \ N_{\text{int}} = 2, \\ d_{\min} &\ge 4 \text{ m} \quad \text{if} \quad \bar{\alpha} = \underline{\alpha} = 4, \ N_{\text{int}} = 16, \\ d_{\min} &\ge 8 \text{ m} \quad \text{if} \quad \bar{\alpha} = 4, \ \underline{\alpha} = 3, \ N_{\text{int}} = 4. \end{aligned}$$

As discussed earlier, due to non-linear path-loss exponents, the number of interferers affecting other simultaneously scheduled TRs seems to be quite limited, i.e., N_{int} is small, where we have more relaxed condition on $d_{\min}$, which still gives a provable guarantee on performance.

VI. ARCAMA (Adaptive RCAMA)

Note that RCAMA chooses new time-slots for unsuccessful TRs with equal probability in the subsequent frames. In fact, one can potentially increase the rate of convergence or adapt to load changes more effectively by intelligently guessing which time-slot is likely to be successful and by biasing the time-slot access probability. As an example, a time-slot with consecutive success is highly likely to be "safe", so that it would be beneficial to sustain the corresponding time-slot with higher probability at the next frame than other time-slots. In this section, we propose a general family of variations of RCAMA, ARCAMA (Adaptive RCAMA) family (a subset of the DRS family), which adaptively assigns different time-slot access probabilities, depending on the past contention history. This provides ARCAMA with a more efficient learning of local contention patterns, leading to more robustness to network changes. As shown in Proposition 6.1 below, such variations of RCAMA inherit all rate-optimal properties.

To that end, each link is assigned its own *slot weight vector,* and the individual nodes maintain slot weight vectors for its adjacent outgoing links. This slot weight vector is updated

TABLE I
WEIGHT INCREASE/DECREASE: SS (SLOT STATUS), t IS THE FRAME INDEX.

SS at $t-3$	SS at $t-2$	SS at $t-1$	INC/DEC
SUCC	SUCC	SUCC	$-D_1$
FAIL/IDLE	SUCC	SUCC	$-D_2$
FAIL	FAIL	FAIL	$+I_1$
SUCC/IDLE	FAIL	FAIL	$+I_2$

every frame, mainly based on the transmission results (success or failure) at the past frames. To increase/decrease the slot weight vector, we define the *time-slot status*, which corresponds to the result of past TRs on the corresponding time-slots. Then, the slot access probability is set to be *inversely proportional* to the current weight. This biased probability is used for selecting time-slots for unsuccessful TRs. Also, by setting a minimum and maximum for each weight, we can avoid pathological cases (e.g., the time-slot access probability could be arbitrarily small or close to '1'), i.e., there exist $\bar{w}$ and $\underline{w}$, such that $1 \leq \underline{w} < \bar{w} < \infty$ and $\forall s \in \{1, 2, \ldots, F\}, \forall l \in L$, and $\forall t > 0$, $\underline{w} \leq w_s^l[t] \leq \bar{w}$, where we denote the slot weight vector of link l at frame t by $\boldsymbol{w}^l[t] = (w_s^l[t] : \ s = 1. \cdots, F,)$.

Proposition 6.1: For any fixed topology and feasible load and any positive integer $m < \infty$, in ARCAMA with history m, Theorems 5.1 and 5.2 still hold.

We skip the proof for brevity, since it is analogous to those for RCAMA.

VII. SIMULATIONS

A. Simulation Setup

Simulated network. We simulate wireless multi-hop networks with nodes which are randomly distributed in a 1000 $\times$ 1000 meter-square area. Thermal noise power at each receiver (i.e., η_j), the minimum required SINR level (i.e., γ), and the transmit power level (i.e., P) are set to be -90 dBm, 18 dB, and 15 dBm, respectively. The frame size F is set to be 10 time-slots. Other simulation setup will be explained as needed. Each simulation results are generated through 50 experiments with different random seeds.

Algorithms for comparison. We evaluate the performance of the three variants of RCAMA and ARCAMA: (A)RCAMA-VIR, (A)RCAMA-MAX, and (A)RCAMA-NOR, where (A)RCAMA-NOR represents the (A)RCAMA without signaling power adjustment at Stage 3. We compare the algorithms proposed in this paper to a simple CSMA (Carrier-Sense Multiple Access) that resolves collisions through sensing of transmissions over interfering links. The frame-based version of CSMA we used is operated as follows:

(a) All links are randomly ordered, denoted by the sequence of links: $(l_1, l_2, \ldots, l_L)$.
(b) Each link l_i randomly selects ρ_{l_i} slots out of all slots that are not sensed to be interfering from the previously scheduled links. We use the sensing threshold to be twice of transmission range.

After a frame schedule is determined by the CSMA, for per-slot operation, it just uses a single-stage RTS/CTS signaling to gain access to the channel.

Weight maintenance in ARCAMA. We use a simple weight maintenance algorithm based on three frame contention history in ARCAMA, where we increase (decrease) a weight more aggressively for back-to-back failures (successes) on a slot over the past three frames. We expect to see even better performance increase when more sophisticated maintenance algorithms are used. The intuition for these choices is that more back-to-back successes at a time slot indicate that the offered loads around the corresponding node at that time-slot are relatively low (i.e., less "congested"), and transmissions in that time-slot are likely to be successful in the future. Similar intuition is applied for back-to-back failures. We have three kinds of time-slot status: SUCC (FAIL), where a transmission occurs and are successful (unsuccessful), and IDLE otherwise. Table I shows the (additive) increase/decrease parameters to adapt slot weights based on the past three transmission result histories, respectively. The parameters are chosen such that $D_1 > D_2 > 0$, and $I_2 > I_1 > 0$. We have used $D_1 = I_1 = 3$, $D_2 = I_2 = 1$, in all simulation results, where the maximum and minimum weights (i.e., $\bar{w}$ and $\underline{w}$) are set to 30 and 1.

In the next two subsections, we show the simulations results and their interpretations. In Section VII-B, we first investigate the effect of different signaling power adjustment schemes on the throughput performance and energy consumption when there is no load or topology changes for some time. Next, in Section VII-C, we investigate the effect of changes in load on the performance of RCAMA-NOR and ARCAMA-NOR algorithm, again for the network topology in Figure 6(a).

B. Different signaling power adjustment schemes

Figure 6(a) shows the network topology with 40 nodes, and link connectivity generated at random using the parameters in the previous subsection. Figure 6(b) shows the performance of RCAMA and ARCAMA for different loads. We first randomly select a *maximally feasible load* as follows: (i) randomly select a link and increment its load by one and (ii) repeat such an increment step until we cannot add more loads. The x-axis of Figure 6(b) represents the *normalized* load with respect to the number of loads of the chosen maximally feasible load, ranging from 50% to 100%. In y-axis of Figure 6(b) , we plot the aggregate normalized throughput by the aggregate sum (over links) of the maximally feasible load given initially over 3000 frames.

Note that even RCAMA-MAX and RCAMA-VIR that are provably optimal do not achieve rate-optimality (i.e., arrival rate equals departure rate) in Figure 6(b), since out of 50 simulations, there were some instances where 3000 frames were not sufficient to ensure the convergence of the schedule (recall that rate-stability occurs only after convergence of the schedule). Similarly, Figure 6(c) shows the aggregate average power used in contention signaling per one successful transmission for different values of normalized load. Figure 6(d) shows a trace of the used powers for different RCAMA versions when we fix the normalized load to be 0.7.

From these simulation results, we observe the following: (i) ARCAMA has better transient throughput than RCAMA, (ii) With both ARCAMA and RCAMA, the algorithm without power adjustment has greater transient throughput than

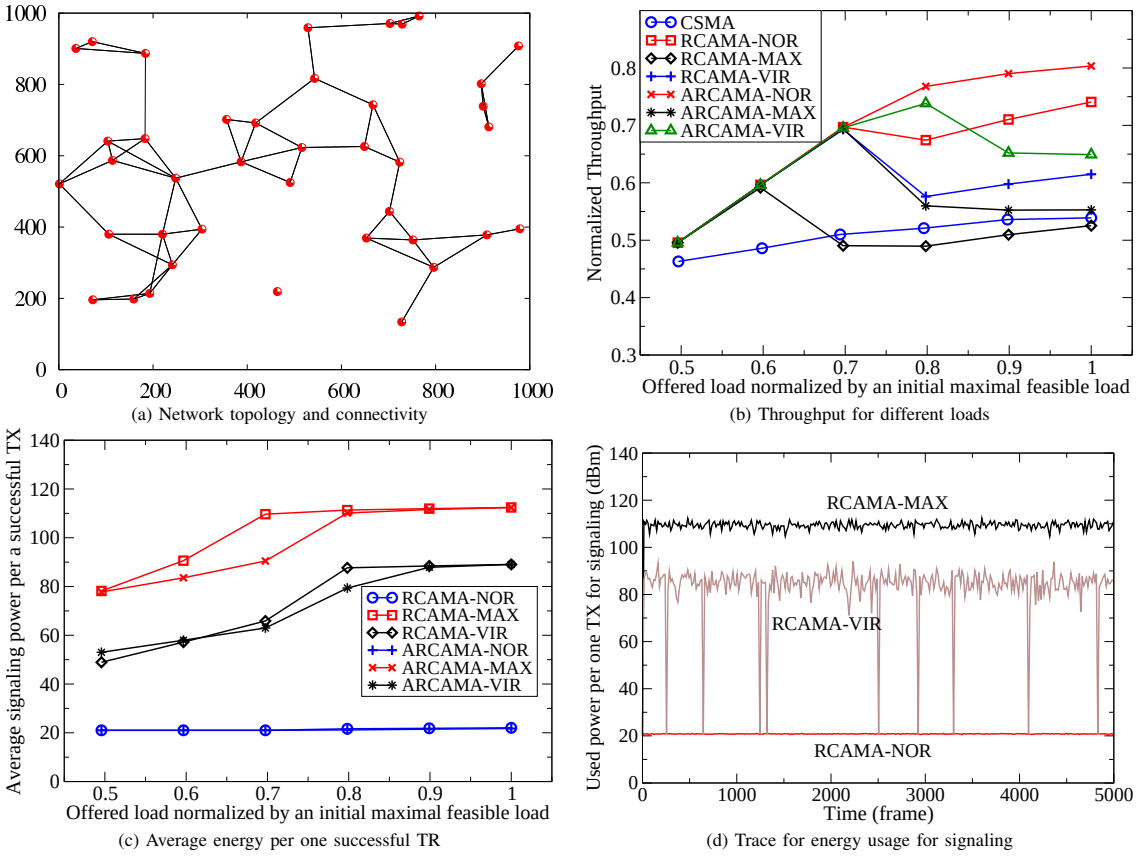


Fig. 6. Throughput performance and energy usage

other rate-optimal versions with power adjustment (i.e., (A)RCAMA-VIR and (A)RCAMA-MAX), as well as better energy saving. Note that, in practice, we may need lower powers than those used by RCAMA-VIR, and the condition on $d_{\min}$ in Theorem 5.2 can be relaxed. This is because RCAMA-VIR is conservatively designed again by considering the point-of-view from *one* single high priority TR and other low priority TRs for the provable rate-optimality. In other words, we have not considered the fact that other high priority TRs, which were valid at Stage 1, also generate interference to interfering low priority TRs, and interference among low-priority TRs still exists. In fact, as seen above from the simulation results, RCAMA with no signaling power adjustment has a better (transient) performance than RCAMA-MAX and RCAMA-VIR even if it is not provably rate-optimal. Essentially, overall high performance of RCAMA (which is significantly larger than CSMA, in particular) is due to accessing the channel with *two-level priority,* which significantly reduces contentions.

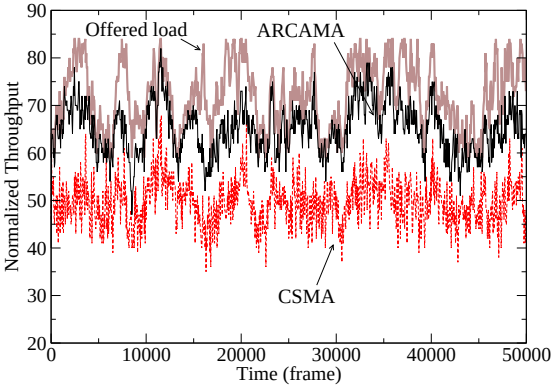
C. Adaptation to load changes

We generate time-varying loads by a random walk model, where we first start with the load that is the link-by-link 60% of a randomly chosen maximally feasible load. Then, at the beginning of each frame we randomly choose L_{ch} links and increase their link loads by one slot with probability P_I, decrease their link loads with probability P_D, or stay at the current load (i.e., no change) with probability $1 - P_I - P_D$. For simplicity, in the simulation, we set $\hat{P} \triangleq P_I = P_D$. Thus, higher values of $\hat{P}$ corresponds to a faster load change with time. Then, the mean load change time (MLCT) over L_{ch} links is $1/(2\hat{P})$ frames.

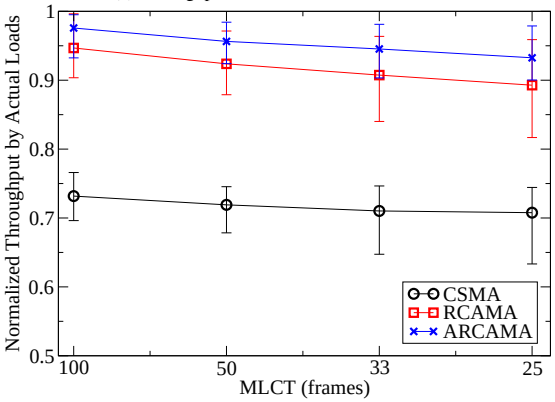
Figure 7(a) shows an example trace of normalized throughput by the aggregate initial maximal feasible load over links for MLCT= 25 frames and $L_{ch} = 5$, where we observe that ARCAMA algorithm tracks the actual load very well, resulting in nice adaptation to time-varying load changes. Figure 7(b) shows that the throughput (over 50000 frames) normalized by the actual (time-varying) offered load for different values of MLCTs ($L_{ch} = 1$) varying from 25 to 100 frames, where the error bars represent the maximum and minimum values of 10 simulations with different random seed values (i.e., different load changing patterns). For a network with a link capacity of 10 Mbps, and a frame-size of 10 (which corresponds to a 10 msec frame duration), this corresponds to a load change ranging from once every 250 msec to once every 1 seconds. We observe that with ARCAMA algorithm, the normalized throughput is above 90%, whereas the CSMA achieves about 7%.

VIII. Related Work and Concluding Remarks

In this section, we review the related work to our research in addition to queue-based scheduling already mentioned in Section I. In regard to scheduling research under the physical interference model (which, however, have different objectives



(a) Throughput traces: MLCT= 25 frames



(b) Normalized throughput for different MLCTs

Fig. 7. Adaptation to load changes

from that of this paper) include [14]–[16]. The work of [14]–[16] develops a mathematical programming formulation for minimizing the frame size over a TDMA wireless multi-hop networks, and proposes a distributed heuristic [14], [15] that considers only closest interferers and a centralized heuristic which is used as a benchmark [16]. The authors in [17] define the "scheduling complexity," i.e., minimum amount of time required until every link is scheduled at least once, which is studied in an asymptotic manner. In [18], [19], the authors have focused only on computing maximum throughput under the physical interference model by jointly considering routing, MAC scheduling, and power control in an optimization framework, but no practical, rate-optimal, distributed algorithm is presented.

Power control only for signaling, which is similar to signaling power adjustment in this paper, has been proposed with the main objective of throughput improvement in literature (e.g., see [20] and references therein). The approaches in [20], however, do not consider the physical interference model and they do not provide a study of provable performance guarantees (i.e., no rate-optimal properties). The idea of using multiple priorities was also used in Z-MAC [21]. However, Z-MAC considers only the graph-based interference model, and its major objective of multiple priorities is to solve the hidden terminal problem with no provable throughput-guarantee, whereas we use two-level priority to get both provable convergence and throughput-guarantee.

To conclude, we have studied the problem of dynamic MAC scheduling for a time-slotted wireless networks. We have proposed a generalized frame-based scheduling framework—DRS—achieving lattice-rate-optimality. As a simple instance of the DRS family, we have proposed RCAMA that operates based on randomized slot selection with synchronous multi-stage signaling having two-level priority. The RCAMA works under both physical and graph based interference models, which shows the advantage of developing an algorithm starting from a general framework. This general framework also allows us to come up with a simple variation that allows faster convergence—ARCAMA.

An issue that we do not directly address in this paper is the behavior of the algorithms when the load is not feasible. To handle such cases, we will need to combine admission control strategies (long time-scale control) along with the MAC algorithms (short time-scale resource allocation) to ensure that a feasible solution exists.

REFERENCES

[1] Y. Yi, G. de Veciana, and S. Shakkottai, "On optimal MAC scheduling with physical interference," in *Proceedings of IEEE Infocom*, 2007.

[2] L. Tassiulas and A. Ephremides, "Stability properties of constrained queueing systems and scheduling for maximum throughput in multihop radio networks," *IEEE Transactions on Automatic Control*, vol. 37, no. 12, pp. 1936–1949, December 1992.

[3] P. Chaporkar, K. Kar, and S. Sarkar, "Throughput guarantees through maximal scheduling in wireless networks," in *Proceedings of Allerton*, 2005.

[4] X. Lin and N. B. Shroff, "The impact of imperfect scheduling on cross-layer rate control in wireless networks," in *Proceedings of IEEE Infocom*, 2005.

[5] X. Wu and R. Srikant, "Bounds on the capacity region of multi-hop wireless networks under distributed greedy scheduling," in *Proceedings of IEEE Infocom*, 2006.

[6] G. Sharma, N. B. Shroff, and R. R. Mazumdar, "Joint congestion control and distributed scheduling for throughput guarantees in wireless networks," in *Proceedings of IEEE Infocom*, 2007.

[7] G. Sharma, R. R. Mazumdar, and N. B. Shroff, "On the complexity of scheduling in wireless networks," in *Proceedings of ACM Mobicom*, 2006.

[8] E. Modiano, D. Shah, and G. Zussman, "Maximizing throughput in wireless networks via gossiping," in *Proceedings of ACM Sigmetrics*, 2006.

[9] A. Eryilmaz, A. Ozdaglar, and E. Modiano, "Polynomial complexity algorithms for full utilization of multi-hop wireless networks," in *Proceedings of IEEE Infocom*, 2007.

[10] X. Lin and S. Rasool, "Constant-time distributed scheduling policies for ad hoc wireless networks," in *Proceedings of IEEE CDC*, 2006.

[11] C. Joo and N. B. Shroff, "Performance of random access scheduling schemes in multi-hop wireless networks," in *Proceedings of IEEE Infocom*, 2007.

[12] P. While, "RSVP and integrated services in the internet: A tutorial," *IEEE Communications Magazine*, May 1997.

[13] Y. Yi, G. de Veciana, and S. Shakkottai, "MAC scheduling with low overheads by learning neighborhood's contention patterns," Princeton University, Tech. Rep., 2007.

[14] A. Behzad and I. Rubin, "Optimum integrated link scheduling and power control for ad hoc wireless networks," *IEEE Transactions on Vehicular Technology*, vol. 3, pp. 275–283, 2006.

[15] R. Negi and A. Rajeswaran, "Physical layer effect on mac performance in ad-hoc wireless networks," in *Proceedings of Communications, Internet, and Information Technology*, 2003.

[16] P. Björklund, P. Värbrand, and D. Yuan, "Resource optimization of spatial TDMA in ad hoc radio networks: A column generation approach." in *Proceedings of IEEE Infocom*, 2003.

[17] T. Moscibroda and R. Wattenhofer, "The complexity of connectivity in wireless networks," in *Proceedings of IEEE Infocom*, 2006.

[18] R. Cruz and A. Santhanam, "Optimal routing, link scheduling and power control in multi-hop wireless networks," in *Proceedings of IEEE Infocom*, 2003.

[19] K. Jain, J. Padhye, V. N. Padmanabhan, and L. Qiu, "Impact of interference on multi-hop wireless network performance," in *Proceedings of ACM Mobicom*, 2003.
[20] M. Krunz, A. Muqattash, and S. Lee, "Transmission power control in wireless ad hoc networks: Challenges, solutions, and open issues," *IEEE Network Magazine*, vol. 18, no. 5, pp. 8–14, 2004.
[21] I. Rhee, A. Warrier, M. Aia, and J. Min, "Z-mac: a hybrid mac for wireless sensor networks," in *Proceedings of ACM SenSys*, 2005.

APPENDIX

Proof of Theorem 3.1(i): We prove the theorem for a DRS algorithm with history 1, which can be readily extended to the case with history $m > 1$.

It is easy to see that a sequence of $(C[t], R[t])$ over frames forms a Markov chain with its state $X[t] = (C[t], R[t])$. Denote by $P^n(a, b)$ the n-step transition probability from the states a to b. Then, to prove convergence to a feasible FS within a finite time, from the standard Markov chain theory with absorbing states (i.e., states with feasible schedules) and the finite sustenance condition, it suffices to show that there exists $0 \leq n < \infty$, such that $P^n(X[0], X[n]) > 0$, where $C[0]$ is infeasible and and $C[n]$ is feasible. Note that if $C[0]$ is feasible, then the result immediately follows. We prove the above by constructing a finite sequence of times: $0 = t_1 < t_2 < \ldots < t_j < \infty$ for some j, such that (i) $P(X[t_{i-1}], X[t_i]) > 0,\ i = 1, \ldots, j$, (ii) $C[t_j]$ in $X[t_j]$ is feasible, and (iii) $d(C[t_i], C[t_j]) > d(C[t_{i+1}], C[t_j]),\ i = 1, \ldots j-1$ (i.e., distance to a feasible $C[t_j]$ is strictly decreasing). The (i), (ii), and (iii) are indeed true from the finite improvement condition and the fact that $d(A, B)$ is upper-bounded for any two FSs A and B. ■

Proof of Theorem 3.1(ii): Let $n_\theta = \sum_{l=1}^{L} \theta_l$ be the total number of loads in the network. Fix a feasible FS $C^\star$, and consider any infeasible initial FS $C \triangleq C[0]$. Note that when C is feasible, then $\tau(C) = 0$, and thus the result immediately follows. From the finite improvement condition and the fact that the total number of FSs are fixed for a fixed F and L, we can easily see that time to decrease $d(C, C^\star)$ by 1 with positive probability is *uniformly upper-bounded* by some frame time $T < \infty$ over all infeasible initial C, and additionally such positive probabilities are *uniformly lower-bounded* by some probability q. Then, we have for some $T < \infty$ and $0 < q < 1$,

$$\mathbb{P}\Big[d(C[T], C^\star) = d(C, C^\star) - 1\Big] \quad \geq \quad q. \tag{5}$$

Note that T and q may depend on the considered scheduling algorithm as well as the given $C^\star$.

The above also implies that with at least probability of $q^{d(C,C^\star)}$, the system converges to $C^\star$ within $T \times d(C, C^\star)$ frames, because within at most T, the distance can decrease at least by one. Note that convergence to a different FS could occur much earlier, since there could be multiple feasible FSs. Further, note that for any initial FS C, its distance to $C^\star$ is upper-bounded by n_θ, i.e., $d(C, C^\star) \leq n_\theta$. Thus,

$$\begin{aligned} \mathbb{P}[\tau(C) \leq Tn_\theta] \quad &\geq \quad \mathbb{P}[\tau(C) \leq Td(C, C^\star)] \geq q^{d(C,C^\star)} \\ &\geq \quad q^{n_\theta}, \quad \forall \text{ infeasible } C. \end{aligned} \tag{6}$$

Now, consider the evolution of the FSs at frames $\{0, Tn_\theta, 2Tn_\theta, \cdots\}$, let i-th interval $I_i = [iTn_\theta, (i+1)Tn_\theta)$, $i = 0, \ldots, t-1$. Then, from (6),

$$\mathbb{P}[\tau(C) \geq tTn_\theta] =$$
$$\mathbb{P}\Big[\text{no convergence over } I_i, i = 0, \ldots, t-1\Big] \leq (1 - q^{n_\theta})^t.$$

By letting $p = 1 - q^{n_\theta}$, and $K = Tn_\theta$, we are done. ■

Proof of Theorem 5.1: The proof consists of two steps: proving the finite sustenance condition, and the finite improvement condition.

1. Finite sustenance condition. Note that from Rule 4.1, the time-slots for the scheduled TRs that were successful are sustained in the same position at the next frame. Clearly, if all the scheduled TRs were successful (i.e., $C[t]$ reaches a feasible FS), then the frame schedule at the current frame would be same as that at the previous frame. Thus, the finite sustenance condition is satisfied.

2. Finite improvement condition. Let the current frame be t, and choose an arbitrary feasible FS $C^\star = [c^\star_{ls}]$. Since $C[t]$ is not feasible (otherwise the result immediately follows), there exists some link l, such that the TR over l on some time-slot s is not successful at this frame t. Let us denote the set of such "unsatisfied links" by $L_u[t]$. Also, denote by $L_g[t]$ the set of links with "good" position w.r.t. $C^\star$, i.e., the set of links whose slot schedules (i.e., row vector of a FS) are equal to those in $C^\star$. We sometimes omit the frame index $[t]$, if it is clear from the text, for ease of presentation. We will show that with positive probability, an $l \in L_u$ can be moved to a slot, such that the TR over l becomes successful (in a finite number of slots).

Note that a link l may be in *both* L_u *and* L_g, since even $l \in L_g$ may be unsuccessful because of other links that are scheduled at different slots from those specified by $C^\star$. Thus, we first choose an $l \in L_u \setminus L_g$ if $L_u \setminus L_g \neq \emptyset$ (**Case 1**), and then choose $l \in L_g, L_u \subset L_g$, otherwise (**Case 2**).

Case 1: $l \in L_u \setminus L_g$. In this case, we again consider two sub-cases based on whether $c^\star_{ls}$ = 0 or 1 (i.e., whether the unsuccessful TR over l on slot s is scheduled by $C^\star$ or not).

- **(i)** $c^\star_{ls} = 0$. Since $l \notin L_g$, there is a slot $s' \neq s$, such that $c^\star_{ls'} = 1$ and $c_{ls'}[t] = 0$. Note that the slot schedules of l in $C[t]$ and $C^\star$ must have the same number of 1's. Now from Property 4.1, there is a positive probability that at frame $t+1$ we have $C[t+1]$, such that the scheduled transmission over l on slot s at frame t is *moved* to slot s', and all other scheduled transmissions at frame $t+1$ are scheduled at the same slots as at frame t. If it happens, we have $d(C[t+1], C^\star) = d(C[t], C^\star) - 1$. Note that decrease in distance does not necessarily mean increase in the number of successful TRs.
- **(ii)** $c^\star_{ls} = 1$. We first let L'_s denote the set of scheduled links on s by $C[t]$, but not by $C^\star$, i.e., $L'_s = \{i \in L \mid c_{is} = 1,\ c^\star_{is} = 0\}$. Then, again there are two sub-cases: **(a)** there exists a unsuccessful link $l' \in L'_s,\ l' \neq l$, or **(b)** all the scheduled links in L'_s are successful on s.

  (a): Similar to **(i)**, we can move the unsuccessful TR over l' on s to a time-slot $s' \neq s$, on which a TR is scheduled by $C^\star$, since $L'_s \cap L_g = \emptyset$. Then, we have $d(C[t+1], C^\star) = d(C[t], C^\star) - 1$.

(b): In this sub-case, link l is in the "right" slot s w.r.t. $C^\star$, and all other links scheduled in the wrong slot s w.r.t. $C^\star$ (i.e., L'_s) are nevertheless successful. To have a decrease of distance to $C^\star$, somehow we have to kick out some link in L'_s to other places, and then move any of some other links (scheduled on this slot s by $C^\star$) to this slot s. Now, we need the following claim whose proof will be presented later:

Claim 8.1: Suppose that $C[t+1] = C[t]$ (which is possible from Property 4.1). Then, there exists a link $l' \in L'_s[t+1]$, such that the scheduled TR over l' on s becomes *unsuccessful* at frame $t+1$.

If Claim 8.1 is true, then, at the frame $t+1$, l' corresponds to **Case 1(i)**. Thus, after two frames from t, we can have $d(C[t+2], C^\star) = d(C[t], C^\star) - 1$ with positive probability.

Case 2: $l \in L_g, L_u \subset L_g$. Note that the fact we are in this case implies that all the links in $L_u \setminus L_g$ are *satisfied*, since we always choose first an unsatisfied link in $L_u \setminus L_g$ by construction.

Then, using the same definition of L'_s as that in **Case 1(ii)**, this case corresponds to **Case 1(ii)(b)**, i.e., when all the links in L'_s are successful on slot s. Thus, again based on Claim 8.1, with a positive probability we have $d(C[t+2], C^\star) = d(C[t], C^\star) - 1$. This completes the proof of the first part of Theorem 5.1. Now, it remains to prove Claim 8.1.

Proof of Claim 8.1: By hypothesis (i.e., **Case 1(ii)(b)**), all the scheduled links on s are successful except for l. In other words, all the links in $L_g \cup L'_s[t]$ are successful except for l. Note that $c^\star_{ls} = 1$ and $l \in L_u$. This implies that the aggregate interference by TRs over the links in L_g that are scheduled on s is not enough to make the TR over l on s unsuccessful, and the TRs in $L'_s[t] \setminus \{l\}$ necessarily contribute to the TR failure over l.

Also, since $C[t+1] = C[t]$ by assumption, all the scheduled links on s except for l should have low priority at frame $t+1$. Then, the HPC implies that the high priority valid TR at Stage 1 (in fact, the TR over l should be valid at Stage 1) is guaranteed to be successful, there must an unsuccessful TR over a link in $L'_s[t]$ at frame $t+1$.

When it comes to the constants K and p from Theorem 3.1 for our RCAMA algorithm, we have:

$$K = Tn_\theta \le 2n_\theta,$$

where the second inequality comes from the fact that in all cases in the first part of this proof, $T \le 2$. Also, in (5), assuming that every scheduled link tries to move (randomly) over T slots, whose probability is at least $1/F$, we have:

$$q = (1/F)^{Tn_\theta} \ge (1/F)^{2n_\theta},$$

which implies that

$$p = 1 - q^{n_\theta} \le 1 - (1/F)^{2n_\theta^2}. \quad \blacksquare$$

Proof of Theorem 5.2: To prove that RCAMA-VIR satisfies HPC subject to the condition on $d_{\min}$, it suffices to show that for arbitrary high priority TR in $\mathcal{H}^2_I$, its success is guaranteed. In this proof, we consider the case of RTS-decoding failure of a high priority TR in $\mathcal{H}^2_I$. Similar proof can be applied to the case of CTS-decoding failure.

We consider a high priority TR over over the link from A to B, and a set of most dominant N_{int} low priority TRs over $\mathtt{C}_i \to \mathtt{D}_i$ with its distance z_i, $i = 1, \ldots, N_{\text{int}}$. Note that we have more high and low priority TRs in the network. Suppose that RTS from A to B fails due to the aggregate interference by RTSs from $\mathtt{C}_i$, $i = 1, \ldots, N_{\text{int}}$. (see Figure 5 for a similar scenario with only differences that more high and low priority TRs are in the network and N is replaced by N_{int}). We denote by y_i the distance between the nodes B and $\mathtt{C}_i$, $i = 1, \ldots, N_{\text{int}}$.

First, from (4) we have that

$$\hat{I}^r_{d(l)} \quad \le \quad I^r_{d(l)}. \tag{7}$$

Note that the exact aggregate interference by RTS messages from $\mathtt{C}_1, \ldots, \mathtt{C}_{N_{\text{int}}}$ is given by:

$$P \times \sum_{i=1}^{N_{\text{int}}} \frac{1}{(y_i)^{\alpha_i}}, \tag{8}$$

where α_i is the path loss exponent from $\mathtt{C}_i$ to B. Then, from (7) and (8), we have

$$\hat{I}^r_{d(l)} \le I^r_{d(l)} \le P \times \sum_{i=1}^{N_{\text{int}}} \frac{1}{(y_i)^{\underline{\alpha}}}. \tag{9}$$

Denote by $\mathtt{C}'$ and $\mathtt{D}'$ the virtual transmitter and the receiver, respectively. Now, let $y' = d(\mathtt{B}, \mathtt{C}')$, which is computed by B as follows:

$$(y')^{\alpha_v} = \frac{P}{\hat{I}^r_{d(l)}} \ge 1 / \sum_{i=1}^{N_{\text{int}}} \frac{1}{(y_i)^{\underline{\alpha}}}, \tag{10}$$

where α_v is the path loss exponent from B to $\mathtt{C}'$.

As described in the algorithm description, B assumes that there is no signal power loss between $\mathtt{C}'$ and $\mathtt{D}'$. Based on such assumption and y', B will adjust its CTS message power (denoted by P^v_c) enough to invalidate the CTS from $\mathtt{D}'$ to $\mathtt{C}'$. From (10), this is given by:

$$\frac{P}{P^v_c/(y')^{\alpha_v}} \le \gamma \Rightarrow P^v_c \ge \frac{P(y')^{\alpha_v}}{\gamma} \ge \frac{P}{\gamma \sum_{i=1}^{N_{\text{int}}} 1/(y_i)^{\underline{\alpha}}} \tag{11}$$

Let the path loss exponent from $\mathtt{D}_i$ to $\mathtt{C}_i$ be β_i. Then, the SINR value at $\mathtt{C}'$ for its CTS message from $\mathtt{D}'$ will be:

$$\frac{P/(z_i)^{\beta_i}}{Z_H + P^v_c/(y_i)^{\alpha_i}},$$

where Z_H is the total received power at $\mathtt{C}'$ by other high and low priority TRs except for TRs over $\mathtt{A} \to \mathtt{B}$ and $\mathtt{C}_i \to \mathtt{D}_i$, $i = 1, \ldots, N_{\text{int}}$.

Then, it suffices to show that with P^v_c, all of N_{int} low priority TRs over $\mathtt{C}_i \to \mathtt{D}_i, i = 1, \ldots, N_{\text{int}}$ are invalidated, i,e.,

$$\frac{P/(z_i)^{\beta_i}}{Z_H + P^v_c/(y_i)^{\alpha_i}} \le \frac{P/(z_i)^{\beta_i}}{P^v_c/(y_i)^{\alpha_i}} = \frac{P(y_i)^{\alpha_i}}{P^v_c(z_i)^{\beta_i}} \le \gamma. \tag{12}$$

Now, we have

$$\frac{P/(z_i)^{\beta_i}}{Z_H + P^v_c/(y_i)^{\alpha_i}} \quad \le \quad \frac{P(y_i)^{\alpha_i}}{P^v_c(z_i)^{\beta_i}} \le \frac{P(d_{\text{int}})^{\alpha_i}}{P^v_c(d_{\min})^{\beta_i}} \le \frac{P(d_{\text{int}})^{\bar{\alpha}}}{P^v_c(d_{\min})^{\underline{\alpha}}}$$

$$\leq \frac{\gamma (d_{\text{int}})^{\bar{\alpha}} \sum_{i=1}^{N_{\text{int}}} 1/(y_i)^{\underline{\alpha}}}{(d_{\min})^{\underline{\alpha}}} \quad \text{(from (11))}$$

$$\leq \gamma \frac{N_{\text{int}} (d_{\text{int}})^{\bar{\alpha}}}{(d_{\min})^{2\underline{\alpha}}}. \quad (13)$$

Thus, if $\sqrt[2\underline{\alpha}]{N_{\text{int}}} (d_{\text{int}})^{\bar{\alpha}/(2\underline{\alpha})} \leq d_{\min}$, we have

$$\frac{N_{\text{int}} (d_{\text{int}})^{\bar{\alpha}}}{(d_{\min})^{2\underline{\alpha}}} \leq 1.$$

Thus, (12) is proved, which completes the proof. ■

Yung Yi Yung Yi (S'03-M'06) received his B.S. and the M.S. in the School of Computer Science and Engineering from Seoul National University, South Korea in 1997 and 1999, respectively, and his Ph.D. in the Department of Electrical and Computer Engineering at the University of Texas at Austin, USA in 2006. From 2006 to 2008, he was a post-doctoral research associate in the Department of Electrical Engineering at Princeton University. Now, he is an assistant professor at the Department of Electrical Engineering at KAIST, South Korea. His current research interests include the design and analysis of computer networking and wireless communication systems, especially congestion control, scheduling, and interference management, with applications in wireless ad hoc networks, broadband access networks, economic aspects of communication networks, and green networking systems. He has been served as a TPC member at various conferences such as Mobihoc 2008-2009, Wicon 2009-2010, WiOpt 2009-2010, Infocom 2010-2011, and ICC/Globecom 2010, and also as the local arrangement chair of WiOpt 2009 and CFI (Conference on Future Internet) 2010.

Gustavo de Veciana Gustavo de Veciana (S'88-M'94-SM'01-F'09) received his B.S., M.S, and Ph.D. in electrical engineering from the University of California at Berkeley in 1987, 1990, and 1993 respectively. He is currently a Professor at the Department of Electrical and Computer Engineering at the University of Texas at Austin and the Director of the Wireless Networking and Communications Group (WNCG). His research focuses on the design, analysis and control of telecommunication networks. Current interests include: measurement, modeling and performance evaluation; wireless and sensor networks; architectures and algorithms to design reliable computing and network systems. Dr. de Veciana has been an editor for the IEEE/ACM Transactions on Networking. He is the recipient of General Motors Foundation Centennial Fellowship in Electrical Engineering and a 1996 National Science Foundation CAREER Award, co-recipient of the IEEE William McCalla Best ICCAD Paper Award for 2000, and co-recipient of the Best Paper in ACM Transactions on Design Automation of Electronic Systems, Jan 2002-2004.

Sanjay Shakkottai Sanjay Shakkottai (M'02) received his Ph.D. from the University of Illinois at Urbana-Champaign in 2002. He is currently with The University of Texas at Austin, where he is an Associate Professor in the Department of Electrical and Computer Engineering. He was the finance chair of the 2002 IEEE Computer Communications Workshop in Santa Fe, NM. He received the NSF CAREER award in 2004. His research interests include wireless and sensor networks, stochastic processes and queueing theory. His email address is shakkott@ece.utexas.edu.