Distributed Greedy Approximation to Maximum Weighted Independent Set for Scheduling with Fading Channels

Changhee Joo, *Senior Member, IEEE*, Xiaojun Lin, *Senior Member, IEEE*, Jiho Ryu, and Ness B. Shroff *Fellow, IEEE*

***Abstract*—It has been known that scheduling algorithms designed to achieve throughput optimality and good delay performance often require solving the Maximum Weighted Independent Set (MWIS) problem. However, under most realistic network settings, the MWIS problem is known to be NP-hard. In non-fading environments, low-complexity scheduling algorithms have been provided that converge either to the MWIS solution in time or to a solution that achieves at least a provable fraction of the achievable throughput. However, in more practical systems the channel conditions can vary at faster time-scales than convergence occurs in these lower-complexity algorithms. Hence, these algorithms cannot take advantage of opportunistic gains, and may no longer result in achieving good performance. In this paper, we propose a low-complexity scheduling scheme that performs provably well under fading channels and is amenable to implement in a distributed manner. To the best of our knowledge, this is the first scheduling scheme under fading environments that requires only local information, has a low complexity that grows logarithmically with the network size (provided that the conflict graph has bounded maximum vertex degree), and achieves provable performance guarantees (arbitrarily close to that of the well-known centralized Greedy Maximal Scheduler). We verify that the throughput and the delay of our proposed scheme are close to those of the optimal MaxWeight that solves MWIS at each time. Further, we implement our algorithm in a testbed by modifying the existing IEEE 802.11 DCF. The experiment results show that our implementation successfully accounts for wireless fading, attains the short-term opportunistic gains in practice, and hence substantially outperforms IEEE 802.11 DCF.**

***Index Terms*—Distributed algorithm, wireless scheduling with fading channels, maximum weighted independent set.**

I. Introduction

Scheduling is one of the most fundamental functionalities of wireless networks. It determines which links should transmit at what time and at what data rate. It is well-known that solving the scheduling problem is inherently difficult because the interference relationship is often non-convex and even combinatorial in nature [2]. Further, for large networks it is imperative that the scheduling algorithm is of low complexity and can be implemented in a fully distributed manner. Such requirements make it highly challenging to design easy-to-implement scheduling algorithms.

In the literature, it is well-known that the so-called MaxWeight algorithm is throughput optimal [3]. For graph based interference models, where whether two links interfere or not can be specified by a binary parameter, the MaxWeight algorithm corresponds to the solution to a Maximum Weighted Independent Set (MWIS) problem in the conflict (or interference) graph as follows. In the conflict graph, each link is mapped onto a vertex and two vertices (links) that interfere with each other are connected by an edge. A set of non-connected vertices, which is called an independent set, can transmit data simultaneously. Further, each vertex of the conflict graph is given a weight, which is typically the product of the link rate and its queue length, and which varies across time due to changing queue lengths and time-varying channels. The MaxWeight algorithm then computes an independent set that has the largest total weight (i.e., solution to the MWIS problem). Although the MaxWeight algorithm is throughput optimal, the MWIS problem is NP-Hard in general [2]. Hence, the MaxWeight algorithm incurs high complexity, and further, it is a centralized algorithm that requires global information. Thus, the MaxWeight algorithm is not amenable to practical implementation.

In the literature, there have been many efforts to develop low-complexity and distributed scheduling algorithms with provably good throughput performance [4]–[15]. These algorithms differ in terms of their throughput guarantee, complexity, and delay performance. They can be classified into two categories, depending on whether or not they account for channel fading.

There have been many more scheduling solutions for wireless systems *without fading*. Low-complexity scheduling algorithms have been developed with complexity that grows significantly slower than the network size, and can yet guarantee a non-negligible fraction of the optimal system capacity. As a point of comparison, the *Greedy Maximal Scheduling* (GMS) algorithm (also known as *Longest Queue First* (LQF) algorithm) can provably attain a fraction of the optimal capacity, with complexity that grows linearly with the total number of links L [16]. Other algorithms can reduce the complexity even further. For example, the *Maximal Scheduling* algorithm can attain at least $\frac{1}{\Delta}$ of the optimal capacity, with $O(\log N)$

A preliminary version of this work was presented at ACM MobiHoc'13 [1].

C. Joo and J. Ryu are with ECE, UNIST, email: {cjoo, jihoryu}@unist.ac.kr

X. Lin is with ECE, Purdue University, e-mail: linx@ecn.purdue.edu

N. B. Shroff is with ECE and CSE, The Ohio State University, e-mail: shroff@ece.osu.edu

Jiho Ryu is the corresponding author. This work has been partially supported by NSF grants CNS-1012700, CNS-1065136, CNS-1421576, a grant from the Army Research Office W911NF-14-1-0368, and in part by the Basic Science Research Program through the NRF of Korea (NRF-2014R1A1A1A05002557) funded by the MSIP, and the Center for Integrated Smart Sensors funded by MISP as GFP (CISS-2011-0031863).

complexity [17], where Δ denotes the maximum conflict degree (see (2) for the definition) and N denotes the number of nodes. The *Constant-time* scheduling algorithms, instead, can achieve a comparable capacity with $O(1)$ complexity, i.e., the complexity does not grow with the network size [11]. Further, both the *Pick-and-Compare* algorithm [4], [5] and *Carrier Sensing Multiple Access (CSMA)* algorithm [6], [7] have been shown to achieve the optimal throughput. They incur $O(L)$ and $O(1)$ complexity, respectively. We note that these two algorithms have been observed to lead to poor delay performance [8], [9], and hence the utility of the throughput gain may be debatable, especially for delay-sensitive applications. Nonetheless, these results indicate that good throughput performance may be attained for non-fading environments using algorithms with very low complexity.

In practice, however, most wireless systems experience some level of channel fading. When link rates vary across time due to fading, the system throughput can be further improved by scheduling links when their rate is high. This is known as the opportunistic gain [18]. Exploiting opportunistic gain has been extremely popular in cellular systems. For ad hoc wireless networks, the MaxWeight algorithm can exploit this opportunistic gain and in fact achieve the optimal throughput even with fading. However, as we will elaborate in Section II-A, many of the low-complexity scheduling algorithms described in the previous paragraph cannot exploit the opportunistic gain, and their performance in fading environments will be much worse. Recently, there have been a few other low-complexity schemes that are provably efficient with fading channels. However, they are also either limited to single-hop networks [15] or their performance guarantees are much lower [14].

An exception is perhaps the GMS scheduling algorithm, which computes an approximation to the MWIS problem by choosing the highest weight vertex first, and can guarantee $\frac{1}{\Delta}$ fraction of the optimal capacity *in both fading and non-fading environment* [19]. Other greedy approximations have also been proposed in [20], [21]. However, they require centralized operations and linear complexity $O(L)$. Although distributed greedy approximation algorithms have been developed [22], [23], they still incur a worst case time-complexity of $O(L)$. This high complexity has become a major obstacle preventing these algorithms from being used in practical system because the channel conditions can vary at faster time-scales than $O(L)$. Given that fading is a prevalent phenomenon in most modern wireless systems, an interesting open question is how one can develop *distributed* scheduling algorithms with even *lower complexity* and yet *guarantee good performance*.

In this paper, we answer this open question by proposing a novel low-complexity and distributed greedy approximation algorithm, called DistGreedy, for both fading and non-fading environments. In contrast to the known greedy approximations [20]–[23], our proposed DistGreedy algorithm incurs a low logarithmic complexity that grows slowly with the network size. Further, it requires only local information (such as queue length and link rates of neighboring links) and can be implemented in a distributed fashion. We analytically show that our low-complexity distributed algorithm achieves the fraction arbitrarily close to $\frac{1}{\Delta}$ of the optimal throughput, and demonstrate through simulations that DistGreedy often achieves scheduling performance far better than the provable bounds. Indeed, it empirically achieves throughput and delay performance that is close to that of the MaxWeight scheduler. We also conduct experiments with implementation in a real testbed. We implement a new MAC protocol that captures the essence of DistGreedy by modifying the IEEE 802.11 DCF. Performance comparison with the IEEE 802.11 DCF under channel fading shows that the DistGreedy algorithm can better exploit the opportunistic gains and thus substantially outperform IEEE 802.11 DCF in fading environments.

The rest of the paper is organized as follows. The system model is described in Section II. The DistGreedy algorithm is proposed and analyzed in Section III. We numerically evaluate its performance in Section IV, and provide experiment results based on a testbed implementation in Section V. Then, we conclude.

II. System Model

We consider a wireless network with N nodes and L directed links. We assume that time is slotted and that a single frequency channel is shared by all the links. Multiple link transmissions at the same time slot may fail due to wireless interference. We assume that there is no link error, i.e., a link transmission is successful if there is no simultaneous interfering transmission.

The link rate of a successful transmission depends on its channel state. We assume that the channel state is fixed during a time slot, and changes across time slots. We denote the (global) channel state by h. When the channel is in state h, link l can transfer r_l^h unit of data if its transmission is successful. Let $\mathcal{H}$ denote the set of all the channel states. We assume that the channel state attains values from a finite set with probabilities given by the distribution π^h, with $\sum_{h\in\mathcal{H}} \pi^h = 1$.

In order to account for wireless fading, we employ a channel-dependent interference model as follows. Let $C_{kl}^h \in \{0,1\}$ denote the interference relationship between link k and link l when the channel state is h. We set $C_{kl}^h = 0$ if link l does not interfere with link k (and therefore they can transmit simultaneously), and $C_{kl}^h = 1$, otherwise. We assume that the interference relationship is symmetric, i.e., $C_{kl}^h = C_{lk}^h$. We note that the dependency on h represents a major departure from existing works for non-fading environments. Specifically, the interference relationship as well as the link rate may change across time in our model. This model is not only a simplified version that captures the fundamental characteristics of the more accurate SINR interference model [24], but also a general model that includes many interference models used in the literature to model FH-CDMA, Bluetooth, and IEEE 802.11 DCF network systems [17], [25], as special cases.

Given a network system, our interference model admits a unique *conflict graph* at each channel state h, which clearly presents the underlying interference constraints. For each link $l \in L$, we draw a vertex in the conflict graph, which is also denoted by the same alphabet l. For every two vertices k, l with $C_{kl}^h = 1$, we connect them with an edge in the conflict graph.

Let $G^h = (V, E^h)$ denote the conflict graph with the set V of vertices and the set E^h of edges under channel state h. The conflict graph explicitly shows the interference relationship of any two vertices (i.e., links in the original network). In the sequel, we deal with the conflict graph throughout the paper.

We now formally formulate the Maximum Weighted Independent Set (MWIS) problem. Suppose that the channel state is h at time slot t. We consider the conflict graph G^h constructed from the interference constraints under channel state h. We begin with some definitions. Vertex x is a *neighbor* of vertex v, if they are connected by an edge in the conflict graph. Let $I^h(v)$ denote the set of neighbors of vertex v including v, and let $I^h(A)$ denote the set of neighbors of vertices in A, i.e., $I^h(A) := \cup_{v\in A} I^h(v)$. Let $w_v(t,h)$ denote a weight associated with vertex v. In particular, we define the weight of vertex v as the product of queue length $Q_v(t)$ and transmission rate r_v^h. Let $w^*(t,h)$ denote the largest weight, i.e., $w^*(t,h) := \max_{v\in V} w_v(t,h)$. Further, let $\bar{w}_v(X;t,h)$ denote the largest weight in the neighborhood of vertex v within X, i.e., $\bar{w}_v(X;t,h) := \max_{x\in X\cap I^h(v)} w_x(t,h)$.

We say that a set S of vertices is an *independent set* (or a *feasible schedule*) if no two vertices in the set are neighbors. Further, an independent set is maximal if no extra vertex can be added. Let $\mathbb{S}^h$ denote the collection of all the feasible independent sets that are available in G^h. The MWIS problem can be formulated as finding S^* such that

$$S^* \in \operatorname*{argmax}_{S\in\mathbb{S}^h} \sum_{v\in S} w_v(t,h). \tag{1}$$

It has been known that at each time t, given a channel state h, the solution to the MWIS problem with weight $w_v(t,h) = Q_v(t)\cdot r_v^h$ results in a throughput-optimal MaxWeight scheduling scheme [26]. However, due to the high computational complexity and the requirement of global information, such a MaxWeight algorithm is difficult to implement in practice. On the other hand, it has been shown in [19], [27] that an imperfect scheduling solution that solves (1) within a factor of γ at every time t, which is called as a γ-approximation algorithm, achieves at least γ fraction of the optimal throughput. To this end, our goal is to develop practical low-complexity scheduling algorithm that can approximately solve (1) with a provable fraction in a distributed fashion.

Remarks: In the above MaxWeight scheduling scheme, we implicitly assume single-hop traffic, i.e., packets are transmitted over a single link and leave the system immediately after the transmission. For multi-hop traffic, the same MaxWeight algorithm can be used by replacing the queue length with a queue differential. (See [3] for the details.) Similarly, our DistGreedy algorithm described in the next section can be extended to multi-hop scenarios in a straightforward manner.

Finally, we define the vertex degree $\delta(h) := \max_{v\in V} |I^h(v)|$, where $|\cdot|$ denotes the cardinality of the set, and the maximum conflict (or interference) degree $\Delta(h)$ as

$$\Delta(h) := \max_{v\in V, S\in\mathbb{S}^h} |I^h(v)\cap S|. \tag{2}$$

In the network, the maximum conflict degree represents the maximum number of simultaneous transmissions in the neighborhood of any link, which can be upper bounded by a constant in many practical interference models [24], [28], and denoted by $\Delta^* = \max_{h\in\mathcal{H}} \Delta(h)$.

A. Related low-complexity scheduling algorithms

Before we describe our solution and evaluate its performance, we illustrate using an example that many of the previous low-complexity scheduling algorithms in the literature suffer from poor performance in systems with fading. The Maximal Scheduling algorithm and the Constant-time scheduling algorithms have been shown to achieve an approximation ratios around $\frac{1}{\Delta^*}$ for system without fading. (Note that this result is comparable to that of GMS.) However, as the following example shows, their performance could be much worse when there is fading.

Let us consider a simple example network with L links, where all the links interfere with each other and only one link can be active at a time. Suppose that there are L channel states $\{h_1, h_2, \ldots, h_L\}$. In each state h_i, link i has rate 1 and the others have rate $\epsilon \ll 1$. Suppose that the channel state changes independently across time and each state occurs with equal probability. Thus, the transmission rate of each link is either 1 with probability $\frac{1}{L}$ or ϵ with probability $1-\frac{1}{L}$. The average link rate is $\frac{1}{L} + \epsilon(1-\frac{1}{L})$. Suppose that the arrival rate at all links are the same and all links are backlogged. In this scenario, the Randomized Maximal Scheduling algorithm may choose any backlogged link uniformly. As a result, the expected transmission rate of the chosen link will also be $\frac{1}{L} + \epsilon(1-\frac{1}{L})$. Hence, the network will become unstable when the arrival rate to each link is larger than $\frac{1}{L}(\frac{1}{L} + \epsilon(1-\frac{1}{L}))$. On the other hand, at each channel state an optimal scheduler can always choose the link with the rate 1. Thus the network can be stable as long as the arrival rate to each link is less than $\frac{1}{L}$. From the above discussion, we can see that the approximation ratio of the Maximal Scheduling could diminish toward 0 (or an arbitrarily small value of ϵ) as L increases. In contrast, if there were no fading and the link rates were fixed (e.g., if all the link rates were 1), then the approximation ratio would have been 1 regardless of the number (L) of links [29]. This example shows that an algorithm that does not use the channel-rate information will likely yield low approximation ratios in systems with fading. Hence, there is a critical difference between systems with fading and systems without fading. In the case of the Constant-time scheduling algorithms in [11], [12], although it could use the channel rates as an input to the algorithm, a careful examination of the proof indicates that the constant-rate assumption is critical, without which the approximation-ratio proof will not work either.

In [30], two variants of the CSMA algorithm have been studied for system with fading. For the first variant, throughput-optimality may be achieved when the speed with which the CSMA algorithm updates states can be very large. In this case, the CSMA algorithm will be able to track the maximum-weight schedule as quickly as the channel state changes. However, such results only hold under the perfect carrier-sensing assumption. In practice, when there is an unavoidable minimal length for the sensing interval, there will be an upper bound on how fast the CSMA algorithm can update its states.

In this case, as shown for the second variant in [30], the approximation ratio of the corresponding CSMA algorithm may be as low as $\frac{1}{\chi}$, where χ denotes the chromatic number of the conflict graph.

We note that the chromatic number can be much larger than Δ^*. For example, in the above simple L-link network, the chromatic number is $\chi = L$, while the value of Δ^* is 1 because there can be at most 1 active link at any time. In a more general setting, e.g., a network under the unit-disc interference model [28], one can show[1] that $\Delta^* \leq 8$, but the chromatic number (and the vertex degree in the conflict graph) can be arbitrarily large as in the L-link network example.

In summary, it remains an important open question to design low-complexity scheduling algorithms that can achieve an approximation ratio closer to that of GMS for systems with fading, but with much lower overhead and without the type of perfect sensing assumption [30].

III. Distributed Greedy Approximation

In this section, we describe our distributed approximate solution to (1) and analyze its performance. We emphasize that the algorithm operates in a distributed manner and each vertex (link) requires only local information from its neighbors in the conflict graph. Throughout this section, we consider the conflict graph G^h at time t under channel state h, and omit the subscripts t and h if there is no confusion.

A. Algorithm description

We assume that each time slot has two parts: contention and transmission. The contention part has several intervals, and each interval is further divided into mini-slots. We determine a feasible schedule during the contention part, and with the computed schedule, transmits actual data during the transmission part.

Our high-level idea is as follows. During the contention part, each node v calculates the largest weight $\bar{w}_v$ in its neighborhood, and a maximal independent set is effected for the links with weight greater than $\frac{\bar{w}_v}{\alpha}$ for some constant α. The chosen (active) links and their neighboring (blocked) links are removed from the conflict graph. The same process repeats over the remaining conflict graph until the graph left becomes empty.

We next present the detailed per-slot operation. At a time slot, let B denote the feasible schedule (independent set of vertices) chosen by our algorithm. We start with an empty set and add vertices to B by executing an iterative algorithm as shown in Algorithm 1.

At each interval i, some vertices are 'determined' as to whether they belong to set B or not. Specifically, vertices in B_i are 'determined' to be in B at interval i, and vertices in $(I(B_i)\backslash B_i)$ are 'determined' not to be in B at interval i. Let

[1]Under the unit-disc interference model, the transmitter and the receiver of any active link must be within a unit distance, and there should be no other active nodes within a unit distance from either the transmitter or the receiver. Thus, the interference is determined by node locations, independent of the channel states. Under this interference model, if we turn off an active link, no more than 8 links in its neighborhood can be activated at the same time [28], and hence, we have $\Delta^* \leq 8$.

Algorithm 1 DistGreedy algorithm.

$V_0 \leftarrow V$, $B_0 \leftarrow \emptyset$
1: **for** $i = 1$ to $\log_\alpha \beta|V|$ **do**
2: $V_i \leftarrow V_{i-1}\backslash I(B_{i-1})$
3: $A_i \leftarrow \emptyset$
4: **for** each $v \in V_i$ **do**
5: calculate $\bar{w}_v(V_i) := \max_{x \in V_i \cap I(v)} w_x$
6: **if** $w_v \geq \frac{\bar{w}_v(V_i)}{\alpha}$ **then**
7: $A_i \leftarrow A_i \cup \{v\}$
8: **end if**
9: **end for**
10: $B_i \leftarrow$ dist_maximal(A_i)
11: **end for**

V_i denote the set of vertices that have not been determined yet at the beginning of interval i, i.e.,

$$V_i := V \backslash \left(\cup_{j=1}^{i-1} I(B_j) \right),$$

which can be rewritten in a recursive form as

$$V_i = V_{i-1} \backslash I(B_{i-1}).$$

We say that a vertex in V_i is *eligible* at interval i. Let $A_i \subset V_i$ denote the set of vertices that will be 'determined' during interval i, from which we will compute B_i. We will soon see how to find A_i and B_i. From the definitions, it is clear that $V_0 = V$ and $B_0 = \emptyset$. Finally, we have a couple of configuration parameters α, β that will be explained later.

Suppose that each vertex v knows its neighbors' weights. In wireless networks, this can be obtained by piggybacking/overhearing the information exchange or by explicitly exchanging control messages.

1) At each interval i, the set of eligible vertices V_i is updated by excluding $I(B_{i-1})$ from V_{i-1} (line 2 in Algorithm 1), where B_{i-1} denotes the set of vertices that are chosen during interval $i-1$ and $I(B_{i-1})$ denotes the set of neighbors for B_{i-1} (including B_{i-1} itself). For this purpose, each vertex that belongs to B_{i-1} should notify its neighbors by broadcasting a control message during interval $i-1$. (See Step 4 below.)
2) Each vertex v in V_i calculates its local maximum weight $\bar{w}_v(V_i) := \max_{x \in V_i \cap I(v)} w_x$ from the weight information of its neighbors (in V_i). Then each vertex v sets itself as one of A_i if $w_v \geq \bar{w}_v(V_i)/\alpha$, where $\alpha > 1$. (Lines $5-8$.)
3) On the set A_i, we compute a maximal independent set in a distributed manner (line 10), which requires $O(\delta \log^2 |V|)$ complexity [31], [32] or $O(\delta)$ complexity with precomputation [10], where δ is the vertex degree. Let B_i denote the obtained set.
4) In the process of computing the maximal independent set, each neighbor of vertex $v \in B_i$ should be informed that v belongs to B_i. Hence, the vertices in $I(B_i)$ will not participate in the next interval.
5) The above procedure repeats for J times, where $J := \lceil \log_\alpha \beta|V| \rceil$. The set $B(= \cup_i B_i)$ of vertices will be

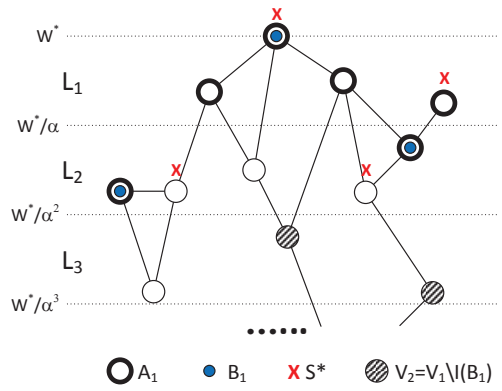


Fig. 1. Conflict graph with vertices and edges in the layered format, where vertices are partitioned into layers according to their weights.

returned as the final result. This is the set of links that will transmit data packets during the time slot.

We have two configuration parameters α and β that will be further discussed in the next section.

Note that our distributed greedy (DistGreedy) algorithm requires $O(\delta)$ complexity at each interval and will run for $\log_\alpha \beta|V|$ intervals (using the algorithm in [10]). Hence, for fixed α and β, the worst-case complexity will be $O(\delta \log_\alpha \beta|V|)$. In some applications, e.g., in regular topologies where δ is a fixed constant, it grows logarithmically with the network size (see further remarks under Proposition 6). Hence, the complexity of DistGreedy is much lower than the $O(|V|)$ complexity of the known distributed implementation of GMS [22], [23].

Remarks: Note that the previous greedy approximations to the MWIS problem shown in [20] have a similar iterative procedure as DistGreedy. However, their algorithm works vertex-by-vertex sequentially, which results in linear complexity in the worst case (e.g., consider a ring topology such that, starting from a link, the link weights decrease in a clockwise direction). Further, they have a different rule for selecting a vertex at each interval, e.g., they select vertex v with the largest $\frac{w_v}{|V_i \cap I(v)|}$ or with $w_v \geq \sum_{x \in V_i \cap I(v)} \frac{w_x}{|V_i \cap I(x)|}$. This selection rule is the key to achieve the provable approximation ratio of $\frac{1}{\delta}$. Unlike this previous work [20], DistGreedy reduces the complexity significantly by considering multiple vertices in parallel and do not follow the strict sequential ordering. Further, the procedure stops after a certain number of intervals. At each interval, DistGreedy selects the vertices v with $w_v \geq \max_{x \in V_i \cap I(v)} w_x/\alpha$. The end result is a much better approximation ratio ($\approx \frac{1}{\Delta^*}$) and a much better complexity. However, the parallel processing also makes it more difficult to analyze the performance of DistGreedy. Nonetheless, in the next section, we show that due to the selection rule of DistGreedy, it can achieve the approximation ratio arbitrarily close to $\frac{1}{\Delta^*}$.

B. Performance Analysis

We evaluate the performance of our distributed greedy (DistGreedy) algorithm, and show that it can achieve arbitrarily close-to $\frac{1}{\Delta^*}$ fraction of the optimal performance. Motivated by [21], we divide the vertices into layers $L_1, L_2, \ldots$ based on the ratio of their weight to the maximum weight w^*, as [2]

$$L_i = \left\{ v \in V \;\middle|\; \frac{w^*}{\alpha^i} < w_v \leq \frac{w^*}{\alpha^{i-1}} \right\}. \tag{3}$$

Fig. 1 illustrates an example conflict graph in the layered format.

We start our analysis with the following lemmas.

Lemma 1: For $i \leq \log_\alpha \beta|V|$, if vertex $v \in L_i$, then $v \in I(\cup_{j=1}^i B_j)$, and thus

$$L_i \subset I(\cup_{j=1}^i B_j). \tag{4}$$

Proof: If each vertex $v \in L_i$ selects itself for distributed maximal independent set no later than the i-th interval, i.e., if $v \in L_i$ implies $v \in \cup_{j=1}^i A_j$, then we can obtain the lemma, since

$$v \in L_i \Rightarrow v \in \cup_{j=1}^i A_j \Rightarrow v \in \cup_{j=1}^i I(B_j) \Rightarrow v \in I(\cup_{j=1}^i B_j), \tag{5}$$

where the second step comes from the fact $A_j \subset I(B_j)$, since B_j is a maximal independent set on A_j.

Now what remains to be shown is that $v \in L_i$ implies $v \in \cup_{j=1}^i A_j$. We show this by induction. It is clear that when $i = 1$, all vertices $v \in L_1$ belong to A_1, because $w_v > \frac{w^*}{\alpha}$. Suppose that the statement is true for all $i \leq c$. Note that all vertices in $\cup_{j=1}^c A_j$ are not eligible at interval $c+1$ since each vertex in A_j belongs to $I(B_j)$ for $j = 1, 2, \ldots, c$ under our algorithm. Hence, at interval $c+1$, no vertex in $\cup_{j=1}^c A_j$ is eligible, which immediately implies that no vertex in $\cup_{j=1}^c L_j$ is eligible since $L_i \subset \cup_{j=1}^i A_j$ for all $i \leq c$. Now, if there is a vertex $v \in L_{c+1}$ eligible at interval $c+1$, i.e., $v \in V_{c+1}$, then vertex v should be included in A_{c+1}, since all vertices e with $w_e > \frac{w^*}{\alpha^c}$ (i.e., $e \in \cup_{j=1}^c L_j$) are not eligible. Hence, the induction hypothesis must hold for $i = c+1$. This completes the proof. ■

Lemma 1 states that under DistGreedy, any vertex in layer L_i will be 'determined' after interval i ends. In the following lemma, we show that each vertex in layer i must have a neighboring vertex that has a similar or higher weight and that is chosen by DistGreedy after interval i ends. Combining these two lemmas, we can show that after interval i, every vertex in L_i or above is a neighbor of a vertex that is already chosen by DistGreedy.

Lemma 2: For each vertex $x \in L_i$ (with $i \leq \log_\alpha \beta|V|$), there exists $v \in \cup_{j=1}^i B_j$ such that $x \in I(v)$ and $\alpha \cdot w_v \geq w_x$.

Proof: From Lemma 1, we have $x \in I(\cup_{j=1}^i B_j)$, and thus there exists $v \in \cup_{j=1}^i B_j$ such that $x \in I(v)$. Let $k \leq i$ be the smallest index such that $x \in I(B_k)$. Then, $x \notin I(\cup_{j=1}^{k-1} B_j)$ and there exists $v \in B_k$ with $x \in I(v)$. Since $v \in B_k \subset A_k$, it should satisfy $\bar{w}_v(V_k)/w_v \leq \alpha$ from line 6 of Algorithm 1. Also since $x \in I(v)$ and x is eligible at interval k (because

[2]The algorithm in [21] computes a maximal independent set for each layer, and thus requires for each node to know which layer it belongs to, or equivalently to know w^*. However, knowing the maximum weight w^* may take $O(|V|)$ time to propagate in the worst case. In contrast, DistGreedy works with local weight information and the layering structure is only for the purpose of analysis.

$x \notin I(\cup_{j=1}^{k-1} B_j))$, we have $\bar{w}_v(V_k) \geq w_x$. Hence, we obtain that $\alpha \cdot w_v \geq w_x$. ∎

We can also obtain the following corollary by applying Lemma 2 for $i = 1$ and $x = \text{argmax}_{v \in V} w_v$.

Corollary 3:

$$\alpha \sum_{v \in B_1} w_v \geq w^*. \tag{6}$$

Recall that S^* denotes the maximum weighted independent set over V. We define $D_i(v)$ as the set of vertices in $S^* \cap L_i$ that are connected to v by an edge in the conflict graph, i.e.,

$$D_i(v) := \{x \mid x \in S^* \cap L_i, \text{ and } x \in I(v)\}. \tag{7}$$

Then $|D_i(v)|$ denotes the number of vertices selected by the MWIS solution in layer L_i that conflicts with v. The following lemma shows that the weight sum for S^* within layer L_i can be bounded by the weight sum for the independent set chosen by DistGreedy up to interval i, multiplied by $\alpha \cdot |D_i(v)|$.

Lemma 4: At each interval i, we have

$$\sum_{v \in \cup_{j=1}^{i} B_j} \alpha \cdot |D_i(v)| \cdot w_v \geq \sum_{x \in L_i \cap S^*} w_x. \tag{8}$$

Proof: From Lemma 2, we have that for each vertex $x \in S^* \cap L_i$, there exists $v \in \cup_{j=1}^{i} B_j$ such that $x \in I(v)$ and $\alpha \cdot w_v \geq w_x$. However, multiple x may map to the same v. Nonetheless, for each of such v, at most $|D_i(v)|$ vertices in $S^* \cap L_i$ can potentially be neighbor of v in the conflict graph. Therefore, we can obtain the result. ∎

By summing both sides of (8) for all i, we can bound the maximum weight sum by the weight sum of the vertices chosen by DistGreedy within a constant factor $\alpha\Delta$. (See the proof of Lemma 5 below.) However, if we were to terminate after all vertices are considered, it would have resulted in $O(|V|)$ complexity (e.g., consider a fully connected graph with vertices whose weights are $1, \frac{1}{\alpha+\epsilon}, \frac{1}{(\alpha+\epsilon)^2}, ...$). In the next lemma, we show that even if DistGreedy stops after $\log_\alpha \beta|V|$ intervals, the performance loss would still be negligible.

Lemma 5: Given α and β, Algorithm 1 is a $\frac{1}{\alpha(\Delta(h)+1/\beta)}$-approximation algorithm.

Proof: Let $B := \cup_{j=1}^{J} B_j$, where $J := \lceil \log_\alpha \beta|V| \rceil$. By summing (8) from $i = 1$ to J, we can obtain that

$$\begin{aligned} \sum_{i=1}^{J} \sum_{x \in L_i \cap S^*} w_x &\leq \sum_{i=1}^{J} \sum_{v \in \cup_{j=1}^{i} B_j} \alpha \cdot |D_i(v)| \cdot w_v \\ &\leq \sum_{v \in B} \sum_{i=1}^{J} \alpha \cdot |D_i(v)| \cdot w_v \\ &\leq \sum_{v \in B} \alpha \cdot \Delta \cdot w_v. \end{aligned} \tag{9}$$

Also, for $i > J$, we can obtain that

$$\sum_{i=J+1}^{\infty} \sum_{x \in L_i \cap S^*} w_x \leq \sum_{i=J+1}^{\infty} \sum_{x \in L_i} w_x \leq |V| \cdot \frac{w^*}{\alpha^J} \leq \frac{w^*}{\beta}, \tag{10}$$

where w^* denotes the largest weight among all the vertices. The last inequality holds since $J = \lceil \log_\alpha \beta|V| \rceil$.

Combining (9) and (10), we can obtain that

$$\sum_{x \in S^*} w_x \leq \alpha\Delta \sum_{v \in B} w_v + \frac{w^*}{\beta} \leq \alpha(\Delta + \frac{1}{\beta}) \sum_{v \in B} w_v, \tag{11}$$

where the last inequality comes from (6) and $B_1 \subset B$. Thus our result follows. ∎

It has been shown in non-fading environments that a scheduling solution that is a γ-approximation to the MWIS problem at each time slot can achieve at least γ fraction of the optimal throughput [19], [27]. The result can be applied to fading environment: a scheduling solution that is a $\gamma(h)$-approximation to the MWIS problem under channel state h at each time slot can achieve at least $\min_{h \in \mathcal{H}} \gamma(h)$ fraction of the optimal throughput. Combining it with Lemma 5, we can obtain the following Proposition.

Proposition 6: For any $0 < \epsilon < 1$, by choosing $\alpha = 1 + \frac{\epsilon}{3}$ and $\beta = \frac{3}{\epsilon}$, the corresponding DistGreedy algorithm achieves a competitive ratio of $\frac{1}{\Delta^*(1+\epsilon)}$, where $\Delta^* = \max_{h \in \mathcal{H}} \Delta(h)$.

Proof: With the above choice of α and β, we have from Lemma 5 and $\Delta(h) \geq 1$ that

$$\alpha \left(\Delta(h) + \frac{1}{\beta} \right) \leq \Delta(h) \left(1 + \frac{\epsilon}{3} \right)^2 \leq \Delta(h)(1+\epsilon).$$

In other words, DistGreedy can achieve $\frac{1}{\Delta(h)(1+\epsilon)}$ fraction of the maximum weight sum at each time slot.

It has been shown through the standard Lyapunov stability analysis that a scheduling scheme that solves the maximum weighted independent set problem at each time, where the link weight is the multiplication of its queue length and instantaneous link rate, can achieve the capacity region. Further, if the weight of the schedule computed by an algorithm at each time is at least γ times the maximum weight (i.e., γ-approximation algorithm), the algorithm achieves an approximation ratio of γ of the capacity region (Section V.B of [19]). The result of the proposition follows immediately. ∎

Remark: Recall that GMS achieves the approximation ratio of $\frac{1}{\Delta^*}$. The above result indicates that, by choosing a sufficiently small ϵ, DistGreedy can achieve an approximation ratio arbitrarily close to $\frac{1}{\Delta^*}$. For any fixed $\epsilon > 0$, the value of α and β are independent of the network size. Thus, the number of iterations of the corresponding DistGreedy algorithm will be logarithmic with respect to $|V|$.

In developing a distributed low-complexity GMS algorithm, one of the main difficulties lies in the requirement of the strict global ordering of selected vertices. For example, if the conflict graph is a linear graph, where the weights of vertices monotonically decrease from the left to the right, then the selection of the right-most vertex v with the smallest weight can be made only after the selection of its left-side neighbor u, which again can be made only after the selection of the left-side neighbor of vertex u due to the linear topology. This implies that the selection of the right-most vertex v needs to be made after $O(|V|)$ time.

Our result implies that the strict global ordering in the GMS algorithm is not required for high scheduling performance. A loose ordering would be sufficient, which can result in significant complexity reduction with negligible performance degradation. We highlight that the state-of-the-art

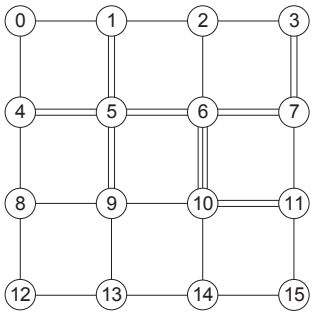


Fig. 2. Grid network topology.

"distributed" $\frac{1}{\Delta(h)}$-approximation algorithm requires $O(|V|)$ complexity [22], [23], while our local greedy algorithm significantly lowers the complexity to $O(\delta(h)\log|V|)$.

IV. Numerical Results

We evaluate DistGreedy, Greedy, and MaxWeight through simulations, where MaxWeight is the optimal scheduler that solves the MWIS problem at each time slot. We simulate two networks: one with a grid topology and the other with a randomly generated topology.

We first consider a grid topology as shown in Fig. 2. Let $\bar{r}_l$ denote the average transmission rate of link l, i.e., $\bar{r}_l := \sum_{h\in\mathcal{H}} r_l^h \pi^h$. Each link l has an average transmission rate of one, two, or three packets per time slot, which are signified in the figures by the number of lines between two nodes, e.g., one line implies one packet per time on average. At each time slot, actual link rate changes and chosen uniformly at random from the range $[0, 2\bar{r}_l]$. Since DistGreedy approximates the optimal solution to the MWIS problem at each time slot, we focus on the behavior of DistGreedy under static interference models, where the interference relationship does not change across time. In particular, we use one-hop (or primary) interference model, under which two links that share a node cannot transmit at the same time. We impose single-hop traffic of load ρ on every link: at each time slot, each link has a packet arrival with probability ρ. The arrivals are i.i.d. across time slots and links. We set DistGreedy to have $\lceil \log_\alpha \beta|V| \rceil$ intervals at each time slot. We use a link-coloring technique to find a maximal independent set, under which $(\delta+1)$ mini-slots are sufficient [10]. Since $\delta = 6$ in our grid topology, we use 10 mini-slots at each interval. The number of mini-slots are not taken into account in the performance measurements. Each result is an average of 10 simulation runs for 10^6 time slots.

Fig. 3 illustrates the performance of DistGreedy in terms of total queue lengths with different α settings and $\beta = 1$. Sharp increases of queue lengths imply the boundary of the capacity region. Note that a larger α means thicker layers and thus a smaller number of intervals. The results show that the performance is not sensitive to the value of α in a range $[1.1, 3]$ and a small number of intervals (e.g., $\alpha = 2.0$) would be sufficient for high throughput performance.

While running DistGreedy, we also trace the maximum weight sum, the weight sum of the schedule selected by DistGreedy, and the weight sum of the schedule that would

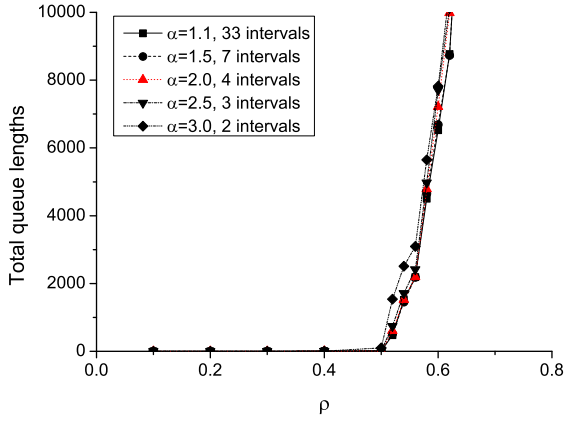
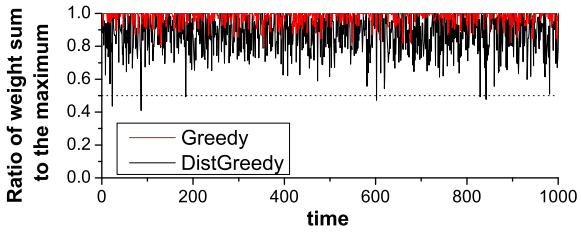
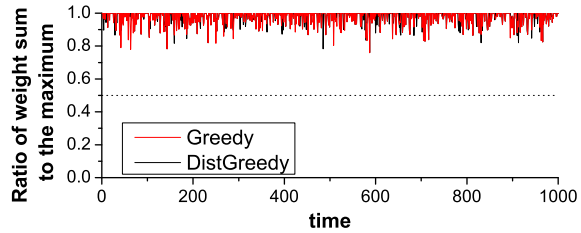


Fig. 3. Performance of DistGreedy with different α.



(a) $\alpha = 2.0$



(b) $\alpha = 1.1$

Fig. 4. Ratio of the achieved weight sum to the maximum weighted sum.

be chosen by Greedy, at each time slot. Fig. 4 depicts the ratio of each weight sum (from DistGreedy and Greedy) to the maximum weight sum. It shows that both GMS and DistGreedy typically achieve much higher ratios than the analytical bound, which is $\frac{1}{2}$ in the one-hop interference model and shown in the figure using a dotted line. Also, as α gets closer to 1, DistGreedy algorithm approaches Greedy algorithm because layers become narrower and the number of intervals increases.

In Fig. 5, we compare the performance of MaxWeight, Greedy, DistGreedy (with $\alpha = 2$, $\beta = 1$), and Q-CSMA, where Q-CSMA is a CSMA algorithm known to be throughput-optimal in non-fading environments. For Q-CSMA, we use the version shown in [9], i.e., each link v sets its access probability for the decision vector to $\frac{1}{|I(v)|}$ and sets its weight for link activity to $\log(Q_l(t)\cdot r_l^h(t))$. The results in Fig. 5 illustrate that Q-CSMA, which has the lowest complexity $O(1)$ among the scheduling schemes, has much poorer throughput and delay performance than the others. In particular, its delay grows quickly at an offered

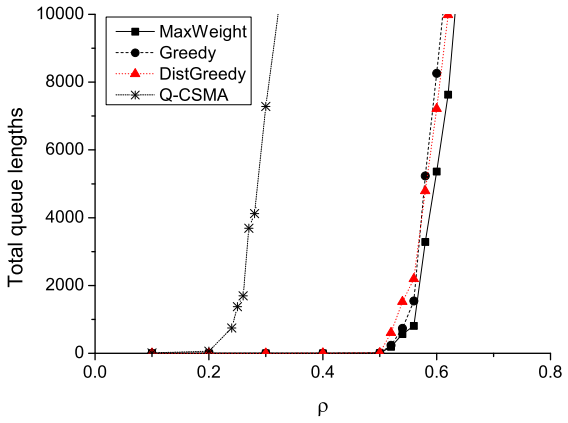
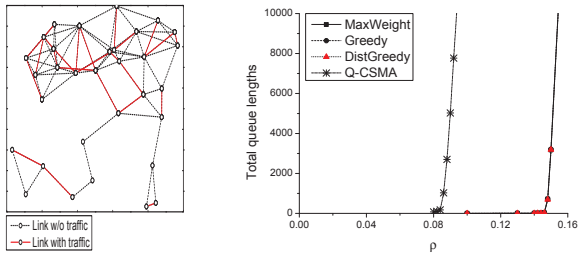


Fig. 5. Performance comparison of scheduling schemes.



(a) Network topology (b) Total queue lengths

Fig. 6. Performance comparison in a random topology.

load much lower than that in other algorithms, which suggests that it is not throughput optimal in fading environments. In contrast, DistGreedy has similar queue lengths to MaxWeight and Greedy. In other words, it empirically achieves similar throughput and delay performance to the optimal. Further, the simulation results suggest that the actual performance of DistGreedy could significantly outperform the analytical lower bounds.

Next we consider a network that is randomly generated. We place a total of 32 nodes at random within a 1×1 area. We connect two nodes with a link if they are within a distance of 0.25. Each link has a time-varying link rate, which is randomly chosen in the range of $[0, 10]$ packets per slot, and i.i.d. across links and time. We generated single-hop traffic over 24 links, which are chosen at random as shown in Fig. 6(a). Each source injects a random number of packets in the range of $[1, 5]$ at each time slot, with probability ρ. Fig. 6(b) illustrates the performance of MaxWeight, Greedy, DistGreedy, and Q-CSMA under different traffic loads. Again, it shows that DistGreedy achieves similar throughput and delay characteristics as MaxWeight and Greedy even though it has a significantly lower complexity. In comparison, Q-CSMA suffers poor throughput and delay performance.

V. Experiment Results

In this section, we develop DistGreedy implementations by modifying the standard IEEE 802.11 DCF, and provide experimental results comparing their performance with the IEEE 802.11 DCF. We show that the opportunistic gains of wireless fading can indeed be achieved in practice.

We implement versions of DistGreedy in hardware driver by modifying the medium access control of the standard IEEE 802.11 DCF. Specifically, we use the `ath5k` device driver [33] over Voyage Linux [34] installed into the Alix 2D2 system board [35]. We modified it such that each node maintains information of queue lengths and transmission rates (link capacity) of neighboring nodes. We use a part of the IP header to report the queue length information of the transmitter. The transmission-rate information can also be obtained from the device if the frame is successfully received. Each node that receives or overhears the frame updates the queue and rate information of the transmitter from the header. Although the physical layer of our devices is based on the IEEE 802.11b standard, the same MAC can be used with IEEE 802.11a and 802.11g standards. Some experimental results with the 802.11a physical layer can be found in [1].

Unlike IEEE 802.11 DCF, the proposed DistGreedy algorithm (see Algorithm 1) requires time synchronization between nodes. We avoid the synchronization overhead in its implementation by approximating Algorithm 1 while capturing its essential feature. In addition, we modify the original DistGreedy scheme to operate with per-node queues like IEEE 802.11 DCF (instead of per-link queues). Due to these differences, we use $w'_n(t)$ and $\bar{w}'_n(t)$ to denote node n's weight and its local maximum weight at time t, respectively.

Our DistGreedy implementation works just like IEEE 802.11 DCF, except for the operations of selecting the number of contention slots. When transmitter n has a frame to send (asynchronously with other nodes), it chooses the number of contention slots at random within $[0, \mathit{CW\_min}]$ and starts the contention procedure. If the contention fails due to a collision, the upper bound of the window is doubled (e.g., binary exponential backoff) up to at most *CW_max* and the contention restarts. The upper bound is reset to *CW_min* upon a successful transmission. However, the main difference from IEEE 802.11 DCF is that we set *CW_min* and *CW_max* dynamically. Under our DistGreedy implementation, each node maintains the local maximum weight $\bar{w}'_n(t)$ from the information that it has received from its neighbors, and estimates $\theta := \lfloor \frac{\bar{w}'_n(t)}{\alpha \cdot w'_n(t)} \rfloor$. When node n transmits a data frame, it sets *CW_min* and *CW_max* according to the value of θ. For example, in one version of our implementation (see DistGreedy-A7 in Table I), the transmitting node sets $(\mathit{CW\_min}, \mathit{CW\_max}) = (7, 31)$ if $\theta = 0$, $(31, 1023)$ if $\theta = 1$, $(63, 1023)$ if $\theta = 2$, and $(127, 1023)$ otherwise. In this way, our DistGreedy implementation effectively divides the contention period into four intervals (see Section III-A for the definition of the interval).

Note that changing *CW_min* and *CW_max* can have a direct impact on the contention overhead and the throughput performance. For example, if there are only a few nodes in the network, one can decrease *CW_min* and *CW_max*, which effectively reduces the contention period and results in higher throughput even when there is no channel fading. Since the focus of this paper is on how to extract the fundamental performance gains when there is channel fading, we would

like to exclude the above effect from our experiments. Hence, we first conduct a preliminary experiment for the case when the nodes are stationary and when the channel states are static. Here, our goal is to find a parameter setting for DistGreedy such that its performance is close to IEEE 802.11 DCF when the channel states are static, and thus later on we can specifically measure the gain of DistGreedy due to its channel-dependent scheduling. To this end, we evaluate the following three versions of DistGreedy implementation shown in Table I.

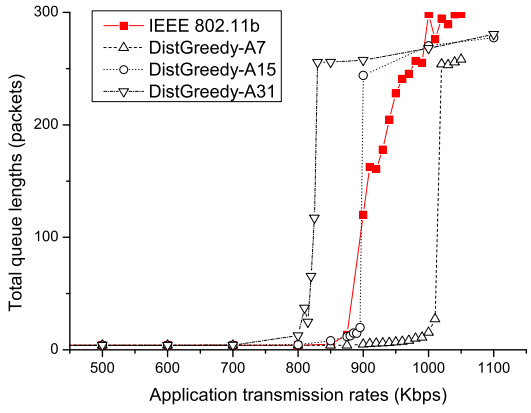
TABLE I
DIFFERENT CONTENTION PARAMETERS OF (*CW_min*, *CW_max*) FOR ASYNCHRONOUS DISTGREEDY APPROXIMATIONS.

	$\theta = 0$	$\theta = 1$	$\theta = 2$	$\theta \geq 3$
DistGreedy-A7	$(7, 31)$	$(31, 1023)$	$(63, 1023)$	$(127, 1023)$
DistGreedy-A15	$(15, 31)$	$(31, 1023)$	$(63, 1023)$	$(127, 1023)$
DistGreedy-A31	$(31, 31)$	$(63, 1023)$	$(127, 1023)$	$(255, 1023)$

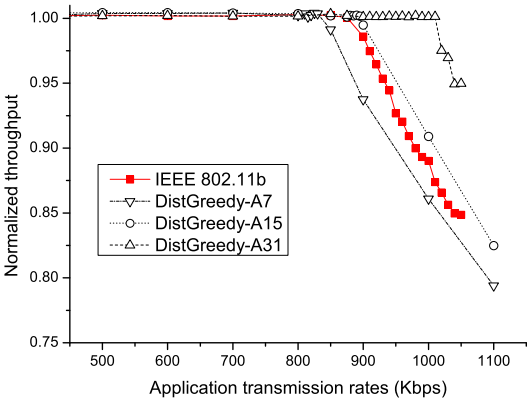
We conduct a simple experiment with these three asynchronous approximations of DistGreedy and the IEEE 802.11 DCF in a network of 5 static nodes placed in a small area of 1x1 m^2. Each of 4 nodes transmits packets to a single receiver. We set $\alpha = 10/9$ for DistGreedy schemes. Further, we set the maximum buffer size to 100 frames, and the IP packet size to 512 bytes. We set all the nodes to transmit at the same rate, and increase their transmit rate at the application level using `iperf` [36].

The results are shown in Fig. 7. As we increase the application transmission rate at each node, the queue lengths grow and the capacity corresponds to the rate at which the queue lengths become substantially large. (See Fig. 7(a).) Alternatively, we can observe performance degradation from the normalized throughput, which is the ratio of the achieved rate to the transmitted rate at the application. (See Fig. 7(b).) As expected, DistGreedy schemes with a smaller contention window performs better in our small-sized network. By comparing their performance with that of the standard IEEE 802.11 DCF, we observe that DistGreedy-A15 achieves similar performance as IEEE 802.11 DCF, which implies that they have similar contention overhead. In the sequel, we continue our experiments using DistGreedy-A15 and focus on performance gains from channel-dependent scheduling. Note that finding an optimal number of intervals and the contention window sizes is still an interesting open question but is beyond the scope of the paper.

We now focus on the performance of DistGreedy with channel fading. We generate a network with three stationary stations and a single mobile receiver, and let each station transmit data to the receiver. Fig. 8 shows the topology in our experiments deployed on the 5th floor of the ECE department building in UNIST, South Korea. The receiver moves back and forth between positions A and C via position B (starts from A). Each one-way travel takes about 20 seconds, and each experiment lasts for 120 seconds. Throughout the experiments, each station can overhear the other's transmissions. Since we only have one receiver, for convenience, we will refer to the channel from STA i to the receiver as 'the channel of STA i'. Overall, the channel of STA 1 is the best and the channel of STA 3 is the worst. While the channel of STA 1 is good when the receiver is at any position, the channel of STA 2 becomes bad when the receiver is at position C and the channel of STA 3 becomes bad when the receiver is at position B (especially when the receiver enters B from C). Note that the transmission rate of a link has a discrete value in $\{1, 2, 5.5, 11\}$ Mbps under the physical layer of the IEEE



(a) Total queue length



(b) Normalized throughput

Fig. 7. Performance of asynchronous DistGreedy scheme with different settings. DistGreedy-A15 achieves similar performance with IEEE 802.11 DCF.

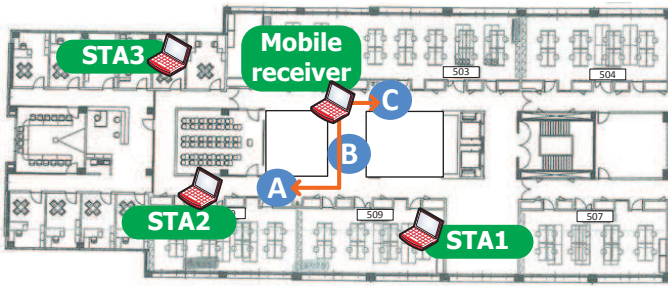
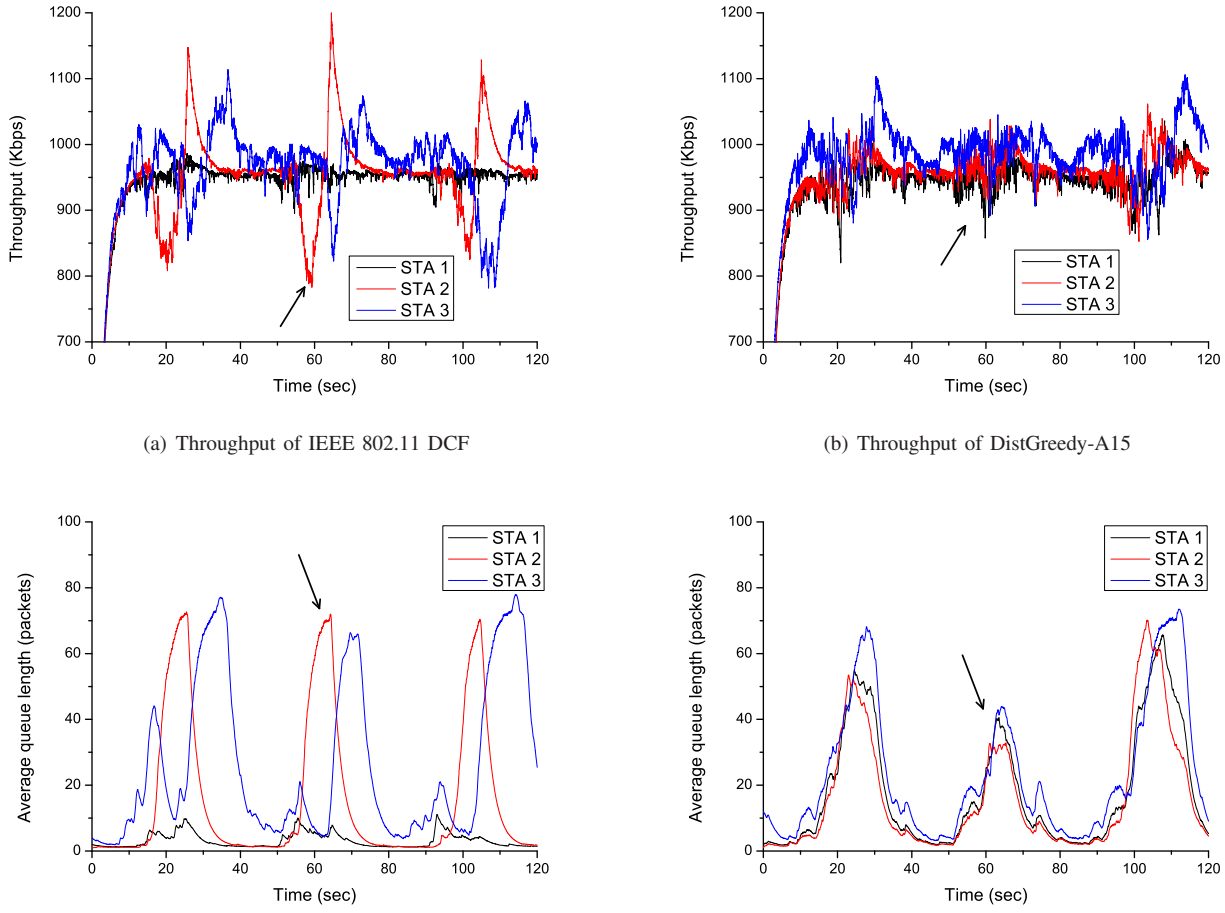


Fig. 8. Experiment setup. Starting from A, the mobile receiver moves back and forth between A and C, via B, for 120 seconds. Each one-way movement takes 20 seconds. The channel of STA 1 is consistently good, while the channel of STA 2 becomes bad at C and the channel of STA 3 becomes bad at B.



(a) Throughput of IEEE 802.11 DCF

(b) Throughput of DistGreedy-A15

(c) Queue length of IEEE 802.11 DCF

(d) Qeuue length of DistGreedy-A15

Fig. 9. Performance of IEEE 802.11 DCF and DistGreedy-A15 when the channel states change. Under IEEE 802.11 DCF, the stations of good channel (e.g., STA 3 at 60 sec) overwhelm the station of bad channel (e.g., STA 1 at 60 sec). In contrast, under DistGreedy-A15, the station of good channel cooperate with the station of bad channel.

802.11b standard. Instantaneous link rate changes frequently across time, and is chosen by the hardware depending on the channel state at a given time. We have observed that the transmission rate changes frequently, commonly between 5.5 and 11 Mbps, and often in an unpredictable manner. Since the transmission rate is not under our control, it is a challenge to maintain exactly the same channel environment when we compare different MAC protocols. In our experiments, we observe that the regular movement of the receiver is helpful to keep overall channel states similar for each experiment run. We have measured the queue lengths and throughputs[3] of all the three stations, when each of them transmits data at 950 Kbps. We present their time-averaged values (using exponential weighted moving average) in Fig. 9.

We first comment on Figs. 9(a) and 9(c), which show the behavior of IEEE 802.11 DCF. Interestingly, although the protocol specification of IEEE 802.11 DCF does not directly take channel condition into consideration, its behavior shows some level of adaptivity to changing channel conditions. For example, at the 60-th second, the receiver is located at position C, where the channels of STA 1 and STA 3 are much better than the channel of STA 2. In Fig. 9(a) (marked by an arrow), it shows that STA 1 and STA 3 achieve good throughput ($\geq$ 950 Kbps) while STA 2 suffers from low throughput. We conjecture that this channel adaptivity has something to do with the binary exponential backoff of DCF. Specifically, when a station experiences a packet loss and exponentially backs off its contention window, the others will be able to take advantage of the opportunity and transmit data more greedily. Since the station with worse channel (i.e., STA 2) is likely to experience more packet losses, it achieves lower throughput.

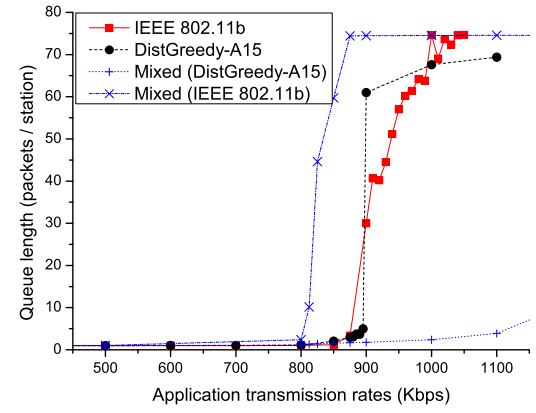
However, there are two limitations in the way IEEE 802.11 DCF adapts to changing channel conditions. First, it does not take into account the backlog, which results in frequent buffer overflows [37]. For example, we have observed from Figs. 9(a) and 9(c) that STA 2 is starved from service at the

[3]The throughput is measured at the MAC layer, and the average can be greater than the rate transmitted from the application (since retransmissions are counted). In our experiments, the channel of STA 3 is worse than those of STA 1 and STA 2 overall. As a result, STA 3 achieves higher throughput than STA 1 and STA 2 (on average) due to frequent retransmissions, as shown in Fig. 9(b).

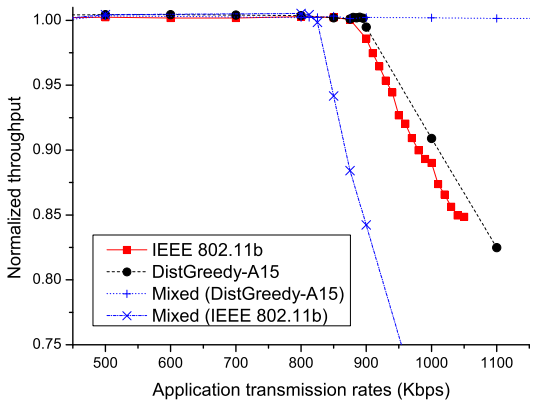
60-th second and its backlog quickly grows. Also, afterwards when the receiver moves to position B, the channel of STA 2 becomes better than that of STA 3 (around the 65-th second). STA 2 now starts transmitting at higher rate and draining its queue, while STA 3 is suffering from very low throughput and its backlog increases significantly. Second, IEEE 802.11 DCF fails to exploit short-term opportunistic gains. For example, even though STA 2 on average has poor channel around the 60-th second, our traces show that its channel condition still fluctuates rapidly. However, under IEEE 802.11 DCF, STA 2 is still prevented from accessing the medium even when its channel state occasionally becomes good.

In contrast, we can observe from Figs. 9(b) and 9(d) that DistGreedy-A15 does not suffer from such pitfalls. First, DistGreedy-A15 is able to exploit short-term opportunistic gains: if stations with a worse channel on average have a good channel state temporarily, they would have a higher priority due to larger queue lengths and can opportunistically transmit data. This property prevents the medium from being occupied by one or two stations. For example, around the 60-th second in Fig. 9(b), even though STA 2 has worse channel on average, it still achieves similar throughput as the others. This short-term opportunistic gain can be confirmed indirectly by comparing the average throughput of the three stations around the 60-th second: 917 Kbps for IEEE 802.11 DCF and 953 Kbps for DistGreedy-A15. Second, Fig. 9(d) indicates the fact that DistGreedy-A15 takes into account the backlog. Specifically, the station with smaller queue length yields transmission opportunity to those with larger queue lengths, and thus the queue lengths of all the stations are somewhat synchronized (unlike IEEE 802.11 DCF). Finally, we highlight that DistGreedy-A15 successfully supports the transmission rate-vector regardless of the positions of the receiver. In contrast, IEEE 802.11 DCF cannot support the input rate-vector in certain positions (especially when one station has a better channel than the other). This implies that DistGreedy-A15 achieves a larger throughput region than IEEE 802.11 DCF.

Next, we test backward compatibility of the proposed scheme. Unless we create a brand-new service network, new scheduling schemes will likely have to coexist with the IEEE 802.11 DCF standard. To this end, the old scheme and the new scheme should be able to work together in the same network, and none of them starves out of service. Further, it is desirable that the new one may perform slightly better than the old one as an incentive for upgrading to new MAC protocol. We have conducted our experiments with four transmitters: two with IEEE 802.11 DCF and two with DistGreedy-A15. Each station transmits to a common stationary receiver at the same transmission rate, which is set at the application level. As we increase the application-level transmission rates, we measure throughput and queue length. Fig. 10 shows the normalized throughput and queue length, which are averaged across stations with the same scheduling scheme. Red and black marks are the results when either all the four stations use IEEE 802.11 DCF or they all use DistGreedy-A15, respectively. The curves with blue marks show the results when two stations use IEEE 802.11 DCF and the other two use DistGreedy-A15. In particular, the curve with blue "x" marks represents the performance of the stations using DistGreedy-A15 and the curve with blue "+" marks represents the performance of the stations using IEEE 802.11 DCF. The results demonstrate that DistGreedy-A15 outperforms IEEE 802.11 DCF when they coexist. The stations using IEEE 802.11 DCF do not starve either, while they experience some performance degradation compared to the case when all the stations use IEEE 802.11 DCF. On the other hand, the stations using DistGreedy-A15 make use of the medium more aggressively, suggesting an incentive for upgrade to our proposed new protocol.



(a) Queue length



(b) Normalized throughput

Fig. 10. Performance when IEEE 802.11 DCF and DistGreedy-A15 coexist in a network (two stations for each scheme). Though DistGreedy-A15 outperforms IEEE 802.11 DCF, the performance of IEEE 802.11 DCF slightly degrades compared to the case when all the stations are IEEE 802.11 DCF.

VI. Conclusion

In this paper, we develop a distributed scheduling scheme that is provably efficient under wireless fading. In a network graph with maximum conflict degree Δ^*, our local greedy scheme achieves arbitrarily close-to $\frac{1}{\Delta^*}$ fraction of the optimal throughput. Further, it incurs $O(\log |V|)$ complexity (or polylogarithmic complexity), where $|V|$ is the number of links,

provided that the conflict graph has bounded maximum vertex degree.

We evaluate our scheme through simulations in grid networks and random networks. The results show that our distributed scheduling scheme is insensitive to parameter settings, and achieves throughput and delay performance similar to those of the optimal solution. We also implement our scheme with hardware by modifying the existing IEEE 802.11 DCF. The experimental results show that our modification results in better throughput performance with low queue length by taking into account time-varying link capacities.

REFERENCES

[1] C. Joo, X. Lin, J. Ryu, and N. B. Shroff, "Distributed Greedy Approximation to Maximum Weighted Independent Set for Scheduling with Fading Channels," in *ACM MobiHoc*, 2013.

[2] G. Sharma, N. B. Shroff, and R. R. Mazumdar, "On the Complexity of Scheduling in Wireless Networks," in *ACM MOBICOM*, September 2006.

[3] L. Tassiulas and A. Ephremides, "Stability Properties of Constrained Queueing Systems and Scheduling Policies for Maximal Throughput in Multihop Radio Networks," *IEEE Trans. Autom. Control*, vol. 37, no. 12, pp. 1936–1948, December 1992.

[4] E. Modiano, D. Shah, and G. Zussman, "Maximizing Throughput in Wireless Networks via Gossiping," *Sigmetrics Performance Evaluation Review*, vol. 34, no. 1, pp. 27–38, 2006.

[5] L. Bui, S. Sanghavi, and R. Srikant, "Distributed Link Scheduling with Constant Overhead," *IEEE/ACM Trans. Netw.*, vol. 17, no. 5, pp. 1467–1480, October 2009.

[6] L. Jiang and J. Walrand, "A Distributed CSMA Algorithm for Throughput and Utility Maximization in Wireless Networks," *IEEE/ACM Trans. Netw.*, vol. 18, no. 13, pp. 960–972, June 2010.

[7] J. Ni, B. Tan, and R. Srikant, "Q-CSMA: Queue-Length Based CSMA/CA Algorithms for Achieving Maximum Throughput and Low Delay in Wireless Networks," *IEEE/ACM Trans. Netw.*, vol. 20, no. 3, June 2012.

[8] M. Lotfinezhad and P. Marbach, "Delay Performance of CSMA Policies in Multihop Wireless Networks: A New Perspective," in *Information Theory and Application Workshop*, Feburary 2010.

[9] J. Ghaderi and R. Srikant, "Effect of Access Probabilities on the Delay Performance of Q-CSMA Algorithms," in *IEEE INFOCOM*, April 2012, pp. 2068–2076.

[10] C. Joo and N. B. Shroff, "Local Greedy Approximation for Scheduling in Multi-hop Wireless Networks," *IEEE Trans. Mobile Computing*, vol. 11, no. 3, pp. 414–426, March 2012.

[11] X. Lin and S. Rasool, "Constant-Time Distributed Scheduling Policies for Ad Hoc Wireless Networks," *IEEE Trans. Autom. Control*, vol. 54, no. 2, pp. 231–242, Feburary 2009.

[12] C. Joo and N. B. Shroff, "Performance of Random Access Scheduling Schemes in Multi-hop Wireless Networks," *IEEE/ACM Trans. Netw.*, vol. 17, no. 5, October 2009.

[13] A. Gupta, X. Lin, and R. Srikant, "Low-Complexity Distributed Scheduling Algorithms for Wireless Networks," *IEEE/ACM Trans. Netw.*, vol. 17, no. 6, pp. 1846–1859, December 2009.

[14] C. Joo, "On Random Access Scheduling for Multimedia Traffic in Multihop Wireless Networks," *IEEE Trans. Mobile Computing*, vol. 12, no. 4, pp. 647–656, April 2013.

[15] B. Li and A. Eryilmaz, "A Fast-CSMA Algorithm for Deadline Constraint Scheduling over Wireless Fading Channels," in *Workshop on Research Allocation and Cooperation in Wireless Networks (RAWNET)*, May 2011.

[16] B. Birand, M. Chudnovsky, B. Ries, P. Seymour, G. Zussman, and Y. Zwols, "Analyzing the Performance of Greedy Maximal Scheduling via Local Pooling and Graph Theory," *IEEE/ACM Trans. Netw.*, vol. 20, no. 1, Feburary 2012.

[17] X. Wu and R. Srikant, "Scheduling Efficiency of Distributed Greedy Scheduling Algorithms in Wireless Networks," in *IEEE INFOCOM*, April 2006, pp. 1–12.

[18] X. Liu, E. K. P. Chong, and N. B. Shroff, "A Framework for Opportunistic Scheduling in Wireless Networks," *Computer Networks*, vol. 41, no. 4, pp. 451–474, March 2003.

[19] X. Lin and N. B. Shroff, "The Impact of Imperfect Scheduling on Cross-Layer Congestion Control in Wireless Networks," *IEEE/ACM Trans. Netw.*, vol. 14, no. 2, pp. 302–315, April 2006.

[20] S. Sakai, M. Togasaki, and K. Yamazaki, "A Note on Greedy Algorithms for the Maximum Weighted Independent Set Problem," *Discrete Appl. Math.*, vol. 126, no. 2-3, pp. 313–322, March 2003.

[21] P.-J. Wan, O. Frieder, X. Jia, F. Yao, X. Xu, and S. Tang, "Wireless Link Scheduling under Physical Interference Model," in *IEEE INFOCOM*, April 2011, pp. 838–845.

[22] J.-H. Hoepman, "Simple Distributed Weighted Matchings," eprint, October 2004. [Online]. Available: http://arxiv.org/abs/cs/0410047v1

[23] S. Basagni, "Finding a Maximal Weighted Independent Set in Wireless Networks," *Telecommunication Systems*, vol. 18, pp. 155–168, 2001.

[24] P. Gupta and P. R. Kumar, "The Capacity of Wireless Networks," *IEEE Trans. Inf. Theory*, vol. 46, no. 2, pp. 388–404, March 2000.

[25] S. Sarkar and L. Tassiulas, "End-to-end Bandwidth Guarantees Through Fair Local Spectrum Share in Wireless Ad-hoc Networks," in *IEEE CDC*, December 2003, pp. 564–569.

[26] M. J. Neely, E. Modiano, and C. E. Rohrs, "Dynamic Power Allocation and Routing for Time-varying Wireless Networks," *IEEE J. Sel. Areas Commun.*, vol. 23, no. 1, 2005.

[27] E. Leonardi, M. Mellia, F. Neri, and M. A. Marsan, "On the Stability of Input-Queued Switches with Speed-Up," *IEEE/ACM Trans. Netw.*, vol. 9, no. 1, Feburary 2001.

[28] P. Chaporkar, K. Kar, X. Luo, and S. Sarkar, "Throughput and Fairness Guarantees Through Maximal Scheduling in Wireless Networks," *IEEE Trans. Inf. Theory*, vol. 54, no. 2, pp. 572–594, February 2008.

[29] X. Lin and S. B. Rasool, "Distributed and Provably Efficient Algorithms for Joint Channel-assignment, Scheduling, and Routing in Multichannel Ad Hoc Wireless Networks," *IEEE/ACM Trans. Netw.*, vol. 17, no. 6, pp. 1874–1887, Dec. 2009.

[30] S.-Y. Yun, J. Shin, and Y. Yi, "CSMA over Time-varying Channels: Optimality, Uniqueness and Limited Backoff Rate," in *ACM MobiHoc*, 2013, pp. 137–146.

[31] M. Luby, "A Simple Parallel Algorithm for the Maximal Independent Set Problem," *SIAM Journal on Computing*, vol. 15, no. 4, pp. 1036–1055, November 1986.

[32] G. Sharma, C. Joo, N. B. Shroff, and R. R. Mazumdar, "Joint Congestion Control and Distributed Scheduling for Throughput Guarantees in Wireless Networks," *ACM Trans. Model. and Comput. Simul.*, vol. 21, no. 1, pp. 5:1–5:25, December 2010.

[33] "Atheros Linux Wireless Drivers," http://wireless.kernel.org/en/users/Drivers/ath5k.

[34] "Voyage Linux," http://linux.voyage.hk.

[35] "ALIX system boards," http://pcengines.ch/alix.htm.

[36] "Alternative for measuring TCP and UDP bandwidth," http://iperf.sourceforge.net.

[37] K. Jagannathan and E. Modiano, "The Impact of Queue Length Information on Buffer Overflow in Parallel Queues," *IEEE Trans. Inf. Theory*, vol. 59, no. 10, pp. 6393–6404, Oct. 2013.

Changhee Joo (S'98-M'05-SM'13) received his Ph.D degree from Seoul National University, Korea. From 2005, he had been with Purdue University and the Ohio State University, USA. In 2010, he joined Korea University of Technology and Education, Korea, and now he is with Ulsan National Institute of Science and Technology (UNIST), Korea. His research interests include resource allocation in wireless networks, network optimization, wireless sensor networks, and Internet protocols. He is an editor of Journal of Communications and Networks, and has served several primary conferences as a committee member, including IEEE INFOCOM, ACM MobiHoc, and IEEE SECON. Dr. Joo received the IEEE INFOCOM 2008 best paper award and the KICS Haedong Young Scholar Award (2014).

Xiaojun Lin (S'02-M'05-SM'12) received his B.S. from Zhongshan University, Guangzhou, China, in 1994, and his M.S. and Ph.D. degrees from Purdue University, West Lafayette, Indiana, in 2000 and 2005, respectively. He is currently an Associate Professor of Electrical and Computer Engineering at Purdue University.

Dr. Lin's research interests are in the analysis, control and optimization of communication networks. He received the 2005 best paper of the year award from *Journal of Communications and Networks* and the IEEE INFOCOM 2008 best paper award. His paper was also one of two runner-up papers for the best-paper award at IEEE INFOCOM 2005. He received the NSF CAREER award in 2007.

Jiho Ryu (S'07-M'12) received his Ph.D degree from the school of Computer Science and Engineering, Seoul National University, Korea, 2012. He has joined Ulsan National Institute of Science and Technology (UNIST), Korea, as a postdoctoral researcher. His research interests include wireless scheduling, wireless LANs, sensor networks, wireless mesh networks, and real-time systems.

Ness B. Shroff (S'91-M'93-SM'01-F'07) received his Ph.D. degree from Columbia University, NY in 1994 and joined Purdue university immediately thereafter as an Assistant Professor. At Purdue, he became Professor of the school of Electrical and Computer Engineering in 2003 and director of CWSA in 2004, a university-wide center on wireless systems and applications. In July 2007, he joined Ohio State University as the Ohio Eminent Scholar of Networking and Communications. University.

His research interests span the areas of wireless and wireline communication networks. He is especially interested in fundamental problems in the design, performance, control, and security of these networks. Dr. Shroff is a past editor for IEEE/ACM Trans. on Networking and the IEEE Communications Letters. He currently serves on the editorial board of the Computer Networks Journal. He has served on the technical and executive committees of several major conferences and workshops. He was the TPC co-chair of IEEE INFOCOM'03, ACM Mobihoc'08, and general chair of IEEE CCW'99 and WICON'08. Dr. Shroff is a fellow of the IEEE.

He has received numerous awards for his work, including two best paper awards at IEEE INFOCOM (in 2006 and 2008), the flagship conference of the field. He has also received the IEEE WiOPT 2013, IEEE WiOpt 2012, and the IWQoS best student paper award, the 2005 best paper of the year award for the Journal of Communications and Networking, the 2003 best paper of the year award for Computer Networks, and the NSF CAREER award in 1996 (his IEEE INFOCOM 2013 and IEEE INFOCOM 2005 papers ware selected as runner-up papers). Dr. Shroff is among the list of highly cited researchers from Thomson Reuters ISI and in Thomson Reuters Book on The World's Most Influential Scientific Minds in 2014. In 2014, he received the IEEE INFOCOM achievement award for seminal contributions to scheduling and resource allocation in wireless networks.