The Case for Addressing the Ordering Effect in Interference-Limited Wireless Scheduling

Xin Che, *Student Member, IEEE*, Hongwei Zhang, *Senior Member, IEEE*, Xi Ju

***Abstract*—Scheduling channel access for interference control is a basic building block of wireless networking. Despite much work in this area, the existing algorithms did not explicitly address the impact of link ordering (i.e., the order in which links are added to the schedule of a time slot) on receiver-side interference accumulation and thus on optimal scheduling. Towards understanding the importance of considering the ordering effect, we formulate the concept of *interference budget*, and, by modeling the scheduling problem as a knapsack problem, we propose the scheduling algorithm *iOrder* that maximizes the schedulability of future channel access when scheduling concurrent transmissions. When selecting concurrent transmitters for a time slot, more specifically, iOrder tries to maximize the additional interference that can be tolerated by all the receivers while satisfying the application requirement on link reliability. We analyze the approximation ratio of iOrder, and, through extensive simulation and testbed-based measurement, we observe that addressing the ordering effect can improve the performance of existing algorithms by a significant margin in the case of both backlogged and online traffic, for instance, improving the throughput and reducing the latency of the well-known algorithm LQF by a factor up to 2 and 24 respectively. Thus our study demonstrates the importance of explicitly addressing the ordering effect in wireless scheduling, which opens up new avenues for future research and for optimizing wireless network performance.**

***Index Terms*—Interference-oriented wireless scheduling; analysis; simulation; measurement**

I. Introduction

With the development of networked embedded sensing and control, wireless networks are increasingly applied to mission-critical applications such as industrial monitoring and control [1]. This is evidenced by the recent industry standards such as WirelessHART [2] and ISA SP100.11a [3] which target wireless networked sensing and instrumentation. In supporting real-time, mission-critical tasks, these wireless networks are required to ensure real-time, reliable data delivery. In data-intensive sensing such as in camera sensor networks, it is also necessary to enable high-throughput data delivery. Nonetheless, wireless communication is subject to various dynamics and uncertainties. Due to the broadcast nature of wireless communication, in particular, concurrent transmissions may interfere with one another and introduce co-channel interference. Co-channel interference not only reduces the reliability and throughput of wireless networks, it also increases the variability and uncertainty in data communication [4]. Therefore, effectively scheduling concurrent transmissions to control co-channel interference has become critical for enabling reliable, predictable wireless communication.

Optimal interference-oriented scheduling in wireless networks has been shown to be NP-complete in general [5], [6], and the research community has proposed different polynomial-time approximation algorithms accordingly. Most approximation algorithms are greedy in nature, and three representatives are Longest-Queue-First (LQF) [7], [8], [9], GreedyPhysical [10], [11], and Maximum-Independent-Set-of-Links (MISL) [12]. When scheduling concurrent transmissions for a time slot, LQF greedily adds to the slot non-interfering links in a decreasing order of their senders' queue lengths; GreedyPhysical selects non-interfering links for the slot in a decreasing order of their interference numbers, where the interference number of a link ℓ is defined as the number of other links that do not share any end-node with ℓ but can be interfered by ℓ; MISL greedily adds to the slot non-interfering links in an increasing order of their lengths. Different from the greedy algorithms, Goussevskaia et al. proposed the algorithm LengthDiversity [5]. In LengthDiversity, the links of a network are grouped into different classes based on their lengths. Links in different classes are scheduled independent of one another, and links in the same class are scheduled using virtual-grid-based coloring. LengthDiversity chooses the link lengths and the size of virtual-grids so that the resulting schedule ensures a certain minimum signal-to-interference-plus-noise-ratio (SINR) at each receiver.

When scheduling transmissions for a time slot, it is usually the interference among the transmissions that limits the number of concurrent transmissions in the slot. In approximation algorithms where links are added to a time slot until reaching the interference limit,[1] the order in which links are added determines the accumulation of interference at the receivers and thus affects the number of concurrent links schedulable in the slot (see Section III for an example). Nonetheless, existing scheduling algorithms either do not take this ordering effect into account (e.g., in LQF) or do not explicitly optimize for the ordering effect (e.g., in GreedyPhysical, LengthDiversity, and MISL).[2] Thus the *open questions* are: 1) how to explicitly optimize the ordering of link addition in interference-limited wireless scheduling? and 2) how does link ordering affect the

This work is supported in part by NSF awards CNS-1136007, CNS-1054634, GENI-1890, and GENI-1633, as well as grants from Ford Research and GM Research. An extended abstract containing some preliminary results of this paper appeared in IEEE ICNP 2011.

Xin Che, Hongwei Zhang, and Xi Ju are with the Department of Computer Science, Wayne State University, U.S.A. E-mail: {chexin,hongwei,xiju}@wayne.edu.

Correspondence author: Hongwei Zhang

[1] For example, the SINR at some receiver falls below a minimum threshold.

[2] By ordering effect, we refer to the impact of link ordering on receiver-side interference accumulation and thus on optimal scheduling.

throughput and delay of data delivery?

To address these open questions for insight into wireless scheduling, we formulate the concept of *interference budget* that, given a set of scheduled transmissions in a time slot, characterizes the additional interference power that can be tolerated by all the receivers without violating the application requirement on link reliability. Then, by modeling the scheduling problem as a knapsack problem, we propose the scheduling algorithm *iOrder* that optimizes link ordering by considering both interference budget and queue length in scheduling. When constructing the schedule for a time slot, iOrder first picks a link with the maximum number of queued packets; then iOrder adds links to the slot one at a time in a way that maximizes the interference budget at each step; this process repeats until no additional link can be added to the slot without violating the application requirement on link reliability.

To understand the impact of link ordering on scheduling, we first analytically prove the approximation ratio of iOrder in Poisson random networks, then we comparatively study the performance of iOrder and existing algorithms via simulation and testbed-based measurement. We observe that optimizing link ordering can improve the performance of existing algorithms by a significant margin in the case of both backlogged and online traffic, for instance, doubling the throughput and reducing the latency of LQF by a factor up to 24, and improving the throughput of LengthDiversity by a factor up to 19.6. Thus our study demonstrates the importance of explicitly optimizing link ordering in wireless scheduling, which opens up new avenues for future research and for optimizing wireless network performance. Our detailed simulation study also discover the surprisingly low performance of LengthDiversity despite its good asymptotic approximation ratio. We find that this is due to the large constant factor hidden in the analysis of LengthDiversity [5]. Therefore, it is important to examine the constant factors when analyzing wireless scheduling algorithms.

As a first step towards addressing the ordering effect in wireless scheduling, our objective in this paper is to characterize the benefits of optimizing link ordering in TDMA scheduling, and we do not focus on distributed algorithm design. Nonetheless, our scheduling algorithm iOrder is amenable to distributed implementation, and we discuss potential approaches in Section VI; we also discuss in Section VI the scenarios when centralized TDMA scheduling is also applicable without having to resort to distributed implementation.

The rest of the paper is organized as follows. We present the system models and problem definition in Section II. We present algorithm iOrder in Section III. Then we perform detailed simulation and measurement study of iOrder and existing algorithms in Sections IV and V respectively. We discuss potential distributed approaches to implementing iOrder in Section VI. We present related work in Section VII and concluding remarks in Section VIII.

II. Preliminaries

Here we present the wireless channel and radio models used in the analytical and simulation parts of this paper, and we define the problem of wireless TDMA scheduling.

A. Wireless channel and radio models

Channel model. To characterize signal attenuation in wireless networks, we use the log-normal path loss model [13] which is widely adopted in protocol design and analysis. By this model, the power P_r (in dBm) of the received signal at a node distance d away from the transmitter is computed as

$$P_r = P_{tx} - PL(d_0) - 10\alpha log_{10}\frac{d}{d_0} + N(0,\sigma^2) \qquad (1)$$

where P_{tx} is the transmission power, $PL(d_0)$ is the power decay at the reference distance d_0, α is the path loss exponent, $N(0,\sigma)$ is a Gaussian random variable with mean 0 and variance σ. In our study, we assume that all the nodes use the same transmission power, and we use different instantiations of α and σ to represent different wireless environments.

Radio model. The reception capability of a radio can be characterized by the bit error rate (BER) and the packet delivery rate (PDR) in decoding signals with specific signal-to-interference-plus-noise-ratios (SINR). Our study considers the CC2420 radios [14], which are compatible with the IEEE 802.15.4 standard and are widely used in wireless sensor network platforms such as TelosB and Tmote Sky motes. For CC2420 radio, the BER for a SINR of γ is computed as follows [15]:[3]

$$\text{BER}(\gamma) = \frac{8}{15} \times \frac{1}{16} \times \sum_{k=2}^{16} (-1)^k \binom{16}{k} e^{(20\times\gamma\times(\frac{1}{k}-1))} \qquad (2)$$

Accordingly, the PDR for a SINR of γ is computed as follows:

$$\text{PDR}(\gamma, f) = (1 - \text{BER}(\gamma))^{8f} \qquad (3)$$

where f is the packet length (in units of bytes) including overhead such as packet header.

Remarks. The aforementioned models are among the most general models commonly used in the literature, even though they do not capture all the real-world phenomena such as the irregularity of wireless communication [16]. We use these models in our analysis and simulation to gain insight into wireless scheduling, and then we verify the analytical and simulation results through testbed-based measurement which captures complex real-world phenomena as we discuss in Section V.

B. Problem definition

We consider wireless networks where, for the purpose of reliable data delivery, the scheduling is required to ensure a minimum signal-to-interference-plus-noise-ratio (SINR) γ_t for all the receivers involved in any transmission. Note that reliable data delivery is important not only for reliability but also for predictable data delivery latency in mission-critical networks [17].

We consider a network $G(V, E)$ where V is the set of nodes, and E is the set of directed links $\{\langle T_i, R_i\rangle : i = 1, 2, \ldots, |E|\}$; each link $\ell_i = \langle T_i, R_i\rangle$ is such that, when node T_i transmits,

[3]Note that the CC2420 radio and many wireless sensor network radios only have one transmission rate, and we do not consider link transmission rate control in this study.

the signal-to-noise-ratio (SNR) at receiver R_i is no less than γ_t in the absence of interference. We assume a time-slotted system where a node can finish transmitting a packet in each time slot. We define a *slot-schedule* $\mathcal{S}_j$ for a time slot j as the set of concurrent transmitting links in slot j. Given a link ℓ_i and a slot-schedule $\mathcal{S}_j$, we define the indicator variable $I(\ell_i \in \mathcal{S}_j)$ whose value is 1 if $\ell_i \in \mathcal{S}_j$, and 0 otherwise. A slot-schedule $\mathcal{S}_j$ is *valid* if, in the presence of the concurrent transmissions of the schedule, the SINRs at all the receivers of the schedule is no less than γ_t and there is no primary interference between the concurrent transmissions.[4] γ_t can be chosen based on the desired packet delivery rate (PDR) and formula (3). To ensure a high PDR (e.g., close to 100%), we use a SINR threshold $\gamma_t = 5dB$ in most places of this paper unless specified otherwise. A *schedule* $\mathbb{S}$ consists of a sequence of slot-schedules $\mathcal{S}_j, j = 1, 2, \ldots$, and $\mathbb{S}$ is valid if $\mathcal{S}_j$ is valid for every time slot j.

We consider two types of traffic models: backlogged traffic and online traffic. In the case of backlogged traffic, every transmitter T_i has L_i ($L_i \geq 0$) number of queued packets to be delivered to the receiver R_i. In the case of online traffic, packets arrive at each transmitter T_i over time and need to be delivered to the receiver R_i.

For backlogged traffic, we define the following *backlog-scheduling* problem:

Problem $\mathbb{P}_{bl}$: Given L_i queued packets at each transmitter T_i ($i = 1, \ldots, |E|$), find a valid schedule $\mathbb{S}_{bl} = \{\mathcal{S}_1, \mathcal{S}_2, \ldots\}$ such that $\sum_{\mathcal{S}_j \in \mathbb{S}_{bl}} I(\ell_i \in \mathcal{S}_j) = L_i$ for every i and that $|\mathbb{S}_{bl}| \leq |\mathbb{S}'|$ for every other valid schedule $\mathbb{S}'$ with $\sum_{\mathcal{S}_{j'} \in \mathbb{S}'} I(\ell_i \in \mathcal{S}_{j'}) = L_i$ for every i.

For online traffic, we define the following *slot-scheduling* problem such that scheduling online traffic becomes solving the slot scheduling problem for each time slot:

Problem $\mathbb{P}_s$: Given a link $\ell_i \in E$, find a valid slot-schedule $\mathcal{S}_{\ell_i}$ such that $\ell_i \in \mathcal{S}_{\ell_i}$, and $|\mathcal{S}_{\ell_i}| \geq |\mathcal{S}'|$ for every other valid slot-schedule $\mathcal{S}'$ with $\ell_i \in \mathcal{S}'$.

Note that, by combining the backlog and online traffic models, we can model hybrid traffic patterns where there is both backlog and online traffic; but, to be concise, we skip the detailed discussion on hybrid traffic models in this paper.

III. Algorithm iOrder

In what follows, we first demonstrate the drawbacks of existing scheduling algorithms by examining their behavior in solving an example slot-scheduling problem, then we present our algorithm iOrder and analyze its approximation ratio and time complexity.

A. A motivating example

We consider a simple convergecast network as shown in Figure 1, where every node has a CC2420 radio and uses a transmission power of $-25dBm$ (a.k.a. power level 3 [14]), and each link is 3.06 meters long. The environment is a typical indoor environment with path loss exponent $\alpha = 3.5$,

[4]There is primary interference between two links ℓ_i and ℓ_j if ℓ_i and ℓ_j share any common end-node.

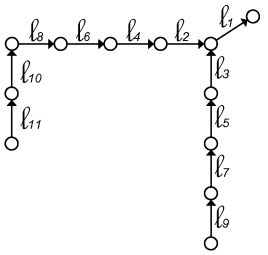


Fig. 1. A simple network

reference distance $d_0 = 1$ meter, and $PL(d_0) = 55dBm$; the mean background noise power is $-105dBm$. This network setup ensures, in the absence of concurrent transmissions, a signal-to-noise ratio (SNR) of $8dB$ for every receiver of a transmitting link. We consider a state of the network where the number of queued packets, denoted by L_i, for each link ℓ_i ($i = 1, 2, \ldots, 11$) satisfies the following:

$$\begin{array}{ll} L_1 > L_k, & 2 \leq k \leq 11 \\ L_{2k} > L_{2k+2}, & 1 \leq k \leq 3 \\ L_{2k} = L_{2k+1}, & 1 \leq k \leq 4 \\ L_8 > L_{10} > L_{11} & \end{array}$$

We consider the slot-scheduling problem $\mathbb{P}'_s$ where we want to find a largest valid schedule that includes link ℓ_1 and ensures a minimum SINR γ_t of $6dB$ at all the receivers. For convenience, we define the *hop distance* between two links ℓ_i and ℓ_j ($i \neq j$) as the number of links in the shortest path connecting any end-node of ℓ_i to any end-node of ℓ_j. With the network setup and minimum SINR requirement of $6dB$, any two links whose hop distance is 0 or 1 can interfere with each other, but any two links whose hop distance is no less than 2 do not interfere with each other in the absence of other concurrent transmissions. For instance, if links ℓ_1 and ℓ_4 transmit together, the SINR at the receiver of ℓ_4 is $0.64dB$ which is below the SINR threshold of $6dB$; on the other hand, if ℓ_1 and ℓ_6 transmit together in the absence of other transmissions, the SINR at the receiver of ℓ_1 and ℓ_6 is 7.75dB and 6.08dB respectively, thus ℓ_1 and ℓ_6 do not interfere with each other in this case.

For problem $\mathbb{P}'_s$, an optimal solution $\mathcal{S}_{opt}$ is $\{\ell_1, \ell_{11}, \ell_9\}$ where the SINR at the receiver of ℓ_1, ℓ_{11}, and ℓ_9 is $7.81dB$, $7.71dB$, and $7.28dB$ respectively. That is, three concurrent links can be scheduled together for $\mathbb{P}'_s$. In what follows, we examine how the existing algorithms LQF [7], GreedyPhysical [10], MISL [12], and LengthDiversity [5] will behave for problem $\mathbb{P}'_s$.

LQF. When constructing a slot-schedule $\mathcal{S}_{gms}$, LQF selects links in a decreasing order of the number of packets queued at their senders. After selecting link ℓ_1, therefore, LQF will consider links ℓ_6 or ℓ_7 first. Without loss of generality, we assume that LQF picks ℓ_6 at random. As discussed earlier, ℓ_1 and ℓ_6 can transmit concurrently in the absence of other transmitters. Once ℓ_6 is added into the slot-schedule, however, no other link can be added without making the SINR at the receiver of ℓ_6 lower than $6dB$ (which is the minimum SINR allowed by problem $\mathbb{P}'_s$). For instance, if we also add ℓ_9 to

the slot-schedule $\{\ell_1, \ell_6\}$, the SINR for the receiver of ℓ_6 will become $5.98dB$. Therefore, LQF will generate the slot-schedule $\mathcal{S}_{gms} = \{\ell_1, \ell_6\}$, which is one link less than the optimal slot-schedule $\mathcal{S}_{opt} = \{\ell_1, \ell_{11}, \ell_9\}$ in concurrency.

GreedyPhysical. When constructing a slot-schedule $\mathcal{S}_{gp}$, GreedyPhysical selects links in a decreasing order of their interference number. Among all the links at least 2-hop away from ℓ_1, ℓ_6 has the largest interference number. Thus GreedyPhysical first selects ℓ_6 to add to the partial slot-schedule $\{\ell_1\}$, after which no more links can be added (similar to the case in LQF). Therefore, $\mathcal{S}_{gp} = \{\ell_1, \ell_6\}$, which is one link less than the optimal slot-schedule $\mathcal{S}_{opt}$ in concurrency.

MISL. When constructing a slot-schedule $\mathcal{S}_{misl}$, MISL selects links in an increasing order of their lengths. In our example, all the links are of equal length; given that we need to find a largest valid schedule that includes link ℓ_1, we first select link ℓ_1. For the network shown in Figure1, MISL prevents the transmission along a link $\ell_i (i \neq 1)$ from occurring concurrently with the transmission along ℓ_1 if the distance between the senders of ℓ_i and ℓ_1 is less than 6.9966 times the length of ℓ_1. Accordingly, $\mathcal{S}_{misl} = \{\ell_1\}$, which is two links less than the optimal slot-schedule $\mathcal{S}_{opt}$ in concurrency.

LengthDiversity. LengthDiversity divides the network links into different classes and then schedule the individual link classes independent of one another. The k-th class $\mathcal{C}_k$ $(k = 0, 1, ...)$ consists of links whose length is in the range $[2^k, 2^{k+1})$. When scheduling links of a class $\mathcal{C}_k$, LengthDiversity first partitions the deployment area of those links into a square grid where each cell has a side length $\mu 2^k$, with $\mu = 4\left(8\gamma_t \cdot \frac{\alpha-1}{\alpha-2}\right)^{\frac{1}{\alpha}}$; it then 4-color the cells so that no neighboring cells have the same color and only the links whose receivers are inside the cells of the same color can transmit concurrently. For the example problem $\mathbb{P}'_s$, $\mu = 12.4$ meters. Therefore, only one link is allowed to transmit at a time in LengthDiversity, and the slot-schedule for $\mathbb{P}'_s$ is $\mathcal{S}_{ld} = \{\ell_1\}$, which is two links less than the optimal slot-schedule $\mathcal{S}_{opt}$ in concurrency.

Goussevskaia et al. [5] proved an approximation ratio $\delta_{ld} = C_{ld} g(L) = O(g(L))$ for LengthDiversity, where $g(L)$ is the logarithm of the largest ratio between the lengths of any two links in the network, and the constant factor $C_{ld} = \frac{4(2(\sqrt{2}\mu+1))^\alpha}{\gamma_t}$. Note that LengthDiversity does not consider the inner structure within cells in scheduling and the constants μ and C_{ld} are usually large (see Table I), which lead to its low performance as shown in the above example and in Section IV.

	μ		C_{ld}	
α	$\gamma_t = 5dB$	$\gamma_t = 8dB$	$\gamma_t = 5dB$	$\gamma_t = 8dB$
2.5	22.6	29.8	4.5e+4	4.4e+4
3.5	11.7	14.1	3.2e+5	3.1e+5
4.5	8.8	10.3	3.5e+6	3.3e+6

TABLE I
CONSTANTS μ AND C_{ld} IN LENGTHDIVERSITY

B. Scheduling for maximal interference budget

From Section III-A, we see that interference is a major factor for limiting the number of concurrent transmissions in a time slot. Existing algorithms LQF, GreedyPhysical, and LengthDiversity do not explicitly consider or minimize interference accumulation during scheduling, thus leading to performance loss. To understand the impact of considering interference accumulation in scheduling, we propose the algorithm *iOrder* that tries to minimize interference accumulation and maximize the additional interference that can be tolerated by all the receivers of a time slot without violating application requirement on link reliability. To this end, we first define the *interference budget of a valid slot-schedule* $\mathcal{S}_{\ell_i}$, denoted by $I_b(\mathcal{S}_{\ell_i})$, as the maximum extra interference that can be tolerated by all the receivers of $\mathcal{S}_{\ell_i}$ without invalidating $\mathcal{S}_{\ell_i}$. Formally, let P_{ℓ_j} be the strength of the signal from the sender to the receiver of link ℓ_j, P_{noise,ℓ_j} be the background noise power at the receiver of ℓ_j, and I_{ℓ_k,ℓ_j} be the strength of the signal reaching the receiver of ℓ_j from the sender of ℓ_k. Then the maximum extra interference that can be tolerated by the receiver of a link $\ell_j \in \mathcal{S}_{\ell_i}$, denoted by $I_b(\ell_j)$, satisfies

$$\frac{P_{\ell_j}}{P_{noise,\ell_j} + \sum_{\ell_k \in \mathcal{S}_{\ell_i}, \ell_k \neq \ell_j} I_{\ell_k,\ell_j} + I_b(\ell_j)} = \gamma_t.$$

Thus

$$I_b(\ell_j) = \frac{P_{\ell_j}}{\gamma_t} - P_{noise,\ell_j} - \sum_{\ell_k \in \mathcal{S}_{\ell_i}, \ell_k \neq \ell_j} I_{\ell_k,\ell_j}.$$

Therefore,

$$\begin{aligned} I_b(\mathcal{S}_{\ell_i}) &= min_{\ell_j \in \mathcal{S}_{\ell_i}} I_b(\ell_j) \\ &= min_{\ell_j \in \mathcal{S}_{\ell_i}} (\tfrac{P_{\ell_j}}{\gamma_t} - P_{noise,\ell_j} - \textstyle\sum_{\ell_k \in \mathcal{S}_{\ell_i}, \ell_k \neq \ell_j} I_{\ell_k,\ell_j}). \end{aligned} \quad (4)$$

Then, for the slot-scheduling problem $\mathbb{P}_s$, we can add to the schedule $\mathcal{S}_{\ell_i}$ one link at a time, and pick the link that maximizes the interference budget of the extended schedule each time a link is to be added. We denote this algorithm as *iOrder-slot* and present it in Algorithm 1. This algorithm

Algorithm 1 iOrder-slot(ℓ_i, E)

Input: starting link ℓ_i, a set E of links where $\ell_i \notin E$
Output: a valid slot-schedule $\mathcal{S}_{\ell_i}$ such that $\ell_i \in \mathcal{S}_{\ell_i}$

```
S_{ℓ_i} = {ℓ_i}, E' = E;
Compute the set of schedulable links: E_c = {ℓ_k : ℓ_k ∈ E', S_{ℓ_i} ∪ {ℓ_k} is a valid schedule};
while E_c ≠ ∅ do
    ℓ_j = arg max_{ℓ_k ∈ E_c} I_b(S_{ℓ_i} ∪ ℓ_k);
    S_{ℓ_i} = S_{ℓ_i} ∪ ℓ_j, E' = E' \ {ℓ_j};
    E_c = {ℓ_k : ℓ_k ∈ E', S_{ℓ_i} ∪ {ℓ_k} is a valid schedule};
end while
Return schedule S_{ℓ_i}.
```

design is similar to approximation algorithms to the knapsack problem [18] in the sense that the interference budget serves as the constraint on the total amount of interference and thus the total number of links allowed for a time slot.

For online traffic, we can apply iOrder-slot to generate a schedule for every time slot. As we will show in Section IV, the choice of the starting link ℓ_i does not significantly affect the size/optimality of the resulting schedule $\mathcal{S}_{\ell_i}$. For the purpose of stabilizing queueing in the network, therefore, we can choose a link with the maximum number of queued packets as the starting link. We denote this algorithm for online traffic as *iOrder-ol* and present it in Algorithm 2.

Algorithm 2 iOrder-ol(m, E_m)

Input: a set E_m of non-empty links where each link ℓ_i has L_i queued packets at the beginning of the m-th time slot
Output: a valid slot-schedule $\mathcal{S}_m$ for the m-th time slot

1: $\ell_j = \arg\max_{\ell_k \in E_m} L_k$;
2: Return iOrder-slot($\ell_j, E_m \setminus \{\ell_j\}$).

For backlogged traffic, we can also apply iOrder-slot to generate the schedule for each time slot in an iterative manner until all the packet transmissions are scheduled. For stabilizing packet queues, we select the link with the maximum number of remaining packets to be scheduled as the starting link for each time slot. We denote this algorithm for backlogged traffic as *iOrder-bl* and present it in Algorithm 3.

Algorithm 3 iOrder-bl(E)

Input: a set E of non-empty links where each link ℓ_i has L_i queued packets
Output: a valid schedule $\mathbb{S}_E$ for transmitting all the queued packets

1: $\mathbb{S}_E = \emptyset$, $E' = E$;
2: **while** $E' \neq \emptyset$ **do**
3: $\quad \ell_j = \arg\max_{\ell_k \in E'} L_k$;
4: $\quad \mathcal{S}_{\ell_j}$ = iOrder-slot($\ell_j, E' \setminus \{\ell_j\}$);
5: $\quad \mathbb{S}_E = \mathbb{S}_E \cup \{\mathcal{S}_{\ell_j}\}$;
6: $\quad$ **for** all $\ell_k \in \mathcal{S}_{\ell_j}$ **do**
7: $\quad\quad L_k = L_k - 1$;
8: $\quad\quad$ **if** $L_k = 0$ **then**
9: $\quad\quad\quad E' = E' \setminus \{\ell_k\}$;
10: $\quad\quad$ **end if**
11: $\quad$ **end for**
12: **end while**
13: Return $\mathbb{S}_E$.

In what follows, we use the general term *iOrder* to refer to any of the algorithms iOrder-slot, iOrder-ol, or iOrder-bl; the context will clarify which one we refer to exactly.

Now let's revisit the example problem $\mathbb{P}'_s$ discussed in Section III-A. iOrder will first select link ℓ_{11} to add to the schedule $\{\ell_1\}$, and the resulting SINR at the receiver of ℓ_1 and ℓ_{11} is $7.93dB$ and $7.81dB$ respectively, both of which are more than $1dB$ above the required $\gamma_t = 6dB$. Accordingly, iOrder can add a third link ℓ_9 to the slot-schedule $\{\ell_1, \ell_{11}\}$ while still maintaining its validity. In the end, iOrder generates an optimal schedule $\{\ell_1, \ell_{11}, \ell_9\}$ for problem $\mathbb{P}'_s$, and the resulting SINR at the receiver of ℓ_1, ℓ_{11}, and ℓ_9 is $7.81dB$, $7.71dB$, and $7.28dB$ respectively. Thus iOrder outperforms existing algorithms for this example problem $\mathbb{P}'_s$.

Approximation ratio. Given that iOrder-slot is a basic element of the iOrder algorithm family (i.e., iOrder-slot, iOrder-ol, iOrder-bl), we analyze the optimality of iOrder-slot in solving the slot-scheduling problem $\mathbb{P}_s$, which will also shed light on the potential effectiveness of iOrder-ol and iOrder-bl. As in the literature [10], we consider the following Poisson network $\mathbb{G}$: n nodes are uniformly distributed on a 2D plane at random with a density of λ nodes per unit area; for connectivity of the network, the transmission range $r_0(n) = 2\sqrt{\frac{log(n)}{\pi\lambda}}$ for every node, and that the transmission power $P_{tx}(n) = \frac{(\gamma_t+\gamma_b)P_{noise}(r_0(n))^\alpha}{G_0}$ such that the signal-to-noise-ratio (SNR) is $\gamma_t + \gamma_b$ at receivers in the absence of interference, where P_{noise} is the background noise power, α is the path loss exponent of the wireless channel, and G_0 is the signal power gain at a reference distance d_0. Then we have

Theorem 1: For network $\mathbb{G}$, the approximation ratio $\delta(n)$ of algorithm iOrder is no more than

$$\widehat{\delta}(n) = \frac{P_{tx}G_0(\pi + \frac{2\pi}{\alpha-2}(1 - (\frac{n}{\pi\lambda})^{\frac{2-\alpha}{2}}))U_{opt}}{I_b\pi(r_{ci}(n))^2}, \tag{5}$$

where $U_{opt} = min\{1 + \frac{I_b(r_{ci}(n))^\alpha}{P_{tx}G_0}, \pi(r_{ci}(n))^2\lambda\}$, $I_b = \frac{\gamma_b}{\gamma_t}P_{noise}$, $r_{ci}(n) = r_0(n)(\frac{n}{\ln n})^{\frac{1}{2}-\frac{1}{\Gamma(\alpha)+\epsilon}}$, $\Gamma(\alpha) = \frac{1}{2} + \frac{\sqrt{9\alpha^2-20\alpha+4}}{2(\alpha-2)}$, and ϵ is any arbitrarily small positive number. $\widehat{\delta}(n) = O((log(n))^{\frac{\alpha}{2}})$.

Proof: For convenience, we denote the optimal scheduling algorithm as OPT. To characterize the approximation ratio of iOrder, we consider an arbitrary link $\langle n_t, n_r\rangle$ (with n_r being the receiver) in a large network $\mathbb{G}$, and compare the maximum single-slot schedule that includes $\langle n_t, n_r\rangle$ in iOrder and OPT. According to Brar et al. [10], there exists a "close-in" region $\mathbb{R}_{ci}$ of radius $r_{ci}(n)$ that is centered at n_r such that, due to power attenuation of wireless signals, interference outside the region is negligible in scheduling for large networks. Thus we focus on the number of concurrent transmitters in $\mathbb{R}_{ci}$ that are allowed in the schedules of iOrder and OPT.

In the schedule $\mathcal{S}_o$ that is generated by OPT, let I_b be the largest interference power that can be introduced to receiver n_r by all the transmitters other than n_t such that the SINR at n_r is no less than the required threshold γ_t. Then

$$\frac{P_{tx}G_0(r_0(n))^{-\alpha}}{P_{noise} + I_b} = \gamma_t$$

Thus

$$I_b = \frac{P_{tx}G_0(r_0(n))^{-\alpha}}{\gamma_t} - P_{noise}$$

Since $P_{tx}(n) = \frac{(\gamma_t+\gamma_b)P_{noise}(r_0(n))^\alpha}{G_0}$,

$$I_b = \frac{\gamma_b}{\gamma_t}P_{noise} \tag{6}$$

The expected number of concurrent transmitters in $\mathcal{S}_o$, denoted by N_{opt}, is maximized if all the transmitters except n_t (called *interferers* hereafter) is on the boundary of the close-in region $\mathbb{R}_{ci}$. In this case, the interference power P_{ci} introduced by each of the interferer is such that

$$P_{ci} = \frac{P_{tx}G_0}{(r_{ci}(n))^\alpha}$$

$\widehat{\delta}(n)$	n=50	n=100	n=200
α=2.5	6.6	6.3	11.2
α=3.5	11.1	11.7	11.5
α=4.5	15	16.9	18.1

TABLE II
$\widehat{\delta}(n)$: UPPER BOUND ON THE APPROXIMATION RATIO OF IORDER

$\widehat{\delta}(n)$	n=50	n=100	n=200
α=2.5	50	79.2	118.4
α=3.5	32.8	45	60.8
α=4.5	27.4	36.2	47

TABLE III
APPROXIMATION RATIO OF GREEDYPHYSICAL

$\widehat{\delta}(n)$	n=50	n=100	n=200
α=2.5	664	664	664
α=3.5	425	425	425
α=4.5	618	618	618

TABLE IV
APPROXIMATION RATIO OF MISL

Thus the total number of interferers N'_{opt} is such that

$$N'_{opt} \leq \frac{I_b}{P_{ci}} = \frac{I_b(r_{ci}(n))^\alpha}{P_{tx}G_0}$$

Thus $N_{opt} = 1 + N'_{opt} \leq 1 + \frac{I_b(r_{ci}(n))^\alpha}{P_{tx}G_0}$. Since the total expected number of nodes in region $\mathbb{R}_{ci}$ is $\pi(r_{ci}(n))^2\lambda$, thus $N_{opt} \leq \pi(r_{ci}(n))^2\lambda$. Therefore,

$$N_{opt} \leq min\{1 + \frac{I_b(r_{ci}(n))^\alpha}{P_{tx}G_0}, \pi(r_{ci}(n))^2\lambda\} \quad (7)$$

In the schedule $\mathcal{S}'_i$ generated by iOrder for the Poisson network $\mathbb{G}$, concurrent transmitters are uniformly distributed in region $\mathbb{R}_{ci}$. This is because, in the process of generating $\mathcal{S}'_i$ in a uniform geometric network $\mathbb{G}$, iOrder will pick a transmitter that is of the maximum minimum distance to the receivers of all the links already scheduled in $\mathcal{S}'_i$. In a large network, this selection process will lead to a set of statistically uniformly distributed transmitters. From the results of Che et al. [17] on the spatial distribution of concurrent transmitters, the spatial process of transmitters in $\mathcal{S}'_i$ can be approximated by a Matern-core-type process which is then modeled as a thinning process Φ of $\mathbb{G}$ [19] where a typical node in $\mathbb{G}$ is retained in Φ with some probability p, and the set of transmitters in region $\mathbb{R}_{ci}$ are the nodes of Φ that lie in $\mathbb{R}_{ci}$. That is, the set of transmitters in schedule $\mathcal{S}'_i$ can be approximated as a spatial Poisson process Φ with density $p\lambda$, and the schedule $\mathcal{S}_i$ that iOrder generates for region $\mathbb{R}_{ci}$ consists of the nodes of Φ that lie in $\mathbb{R}_{ci}$. According to [20] and given that the radius of $\mathbb{G}$ is $\sqrt{\frac{n}{\pi\lambda}}$, the interference I'_b that is incurred to receiver n_r can be calculated as

$$I'_b = P_{tx}G_0p\lambda(\pi + \frac{2\pi}{\alpha - 2}(1 - (\frac{n}{\pi\lambda})^{\frac{2-\alpha}{2}}))$$

Since network $\mathbb{G}$ is large, we expect iOrder to find enough concurrent transmitters such that the resulting SINR at the receiver n_r is close to the required threshold γ_t and thus $I'_b \approx I_b$. Therefore

$$p \approx \frac{I_b}{P_{tx}G_0\lambda(\pi + \frac{2\pi}{\alpha-2}(1 - (\frac{n}{\pi\lambda})^{\frac{2-\alpha}{2}}))}$$

(Note that the path loss exponent $\alpha \geq 2$, thus $0 \leq p \leq 1$.) Therefore, the expected number of transmitters, denoted by N_{iOrder}, in the schedule $\mathcal{S}_i$ of iOrder for region $\mathbb{R}_{ci}$ is calculated as follows:

$$N_{iOrder} \approx p\lambda\pi(r_{ci}(n))^2 = \frac{I_b\pi(r_{ci}(n))^2}{P_{tx}G_0(\pi + \frac{2\pi}{\alpha-2}(1 - (\frac{n}{\pi\lambda})^{\frac{2-\alpha}{2}}))} \quad (8)$$

Therefore, the approximation ratio $\delta(n)$ of iOrder calculates as follows:

$$\delta(n) = \frac{N_{opt}}{N_{iOrder}} \leq \widehat{\delta}(n) = \frac{P_{tx}G_0(\pi + \frac{2\pi}{\alpha-2}(1-(\frac{n}{\pi\lambda})^{\frac{2-\alpha}{2}}))min\{1 + \frac{I_b(r_{ci}(n))^\alpha}{P_{tx}G_0}, \pi(r_{ci}(n))^2\lambda\}}{I_b\pi(r_{ci}(n))^2}. \quad (9)$$

Since

$$min\{1 + \frac{I_b(r_{ci}(n))^\alpha}{P_{tx}G_0}, \pi(r_{ci}(n))^2\lambda\} = O(\pi(r_{ci}(n))^2\lambda),$$

$$\begin{aligned} \widehat{\delta}(n) &= O(\frac{P_{tx}G_0(\pi + \frac{2\pi}{\alpha-2}(1-(\frac{n}{\pi\lambda})^{\frac{2-\alpha}{2}}))\lambda}{I_b}) \\ &= O(P_{tx}) = O((\sqrt{\frac{log(n)}{\pi\lambda}})^\alpha) = O((log(n))^{\frac{\alpha}{2}}). \end{aligned} \quad (10)$$

■

For a setting of $\lambda = 3$, $\gamma_t = 5dB$, $\gamma_b = 3dB$, $P_{noise} = -95dBm$, $G_0 = 1$, and $\epsilon = 0.1$, Table II shows the upper bounds $\widehat{\delta}(n)$ for the approximation ratios of iOrder with different network size n and wireless path loss exponent α. We see that the approximation ratio of iOrder tends to be small, especially in small networks; as we will show through simulation study in Section IV, iOrder actually achieves a throughput very close to (e.g., up to 93.78%) what is feasible in optimal scheduling. Note also that our approved approximation ratio of iOrder is orders-of-magnitude smaller than the proved approximation ratio of LengthDiversity as shown in Table I. For the purpose of comparison, Tables III and IV show the approximation ratios of GreedyPhysical and MISL respectively. We see that the approximation ratios of GreedyPhysical and MISL are significantly greater than that of iOrder too. The approximation ratio of LQF also tends to be large because it may happen that only two concurrent transmissions are allowed in the whole network when the two most queued transmitters are close to one another and can transmit concurrently.[5] To corroborate these analytical results in a wide range of scenarios, we next study the performance of iOrder and the existing algorithms through simulation and testbed-based measurement in Section IV and Section V respectively.

Time complexity. Assuming the basic operation in computing time complexity is computing the SINR at a receiver in the

[5]The presented approximation ratios of LengthDiversity, MISL, GreedyPhysical, and LQF are for the same Poisson networks as those considered for iOrder.The approximation ratios approved for these algorithms (including iOrder) are not necessarily the tightest approximation ratios. Theoretical analysis of the tightest approximation ratios of these algorithms is worthwhile but beyond the scope of this paper; in Sections IV and V, we experimentally analyze the exact performance of these algorithms and demonstrate the benefits of explicitly optimizing the ordering effect in wireless transmission scheduling, which is the main focus of this study.

presence of concurrent transmissions, it is not difficult to find that the time complexity for algorithm iOrder-slot is $O(|E|^3)$. Accordingly, the time complexity for iOrder-ol is $O(|E_m|^3)$, similar to that of iOrder-slot. To compute the time complexity for iOrder-bl, we can replace each link ℓ_i (having L_i queued packets) with L_i virtual links $\ell_{i,1}, \ldots, \ell_{i,L_i}$, each of which has only one queued packet. Then the system will consist of $\sum_{\ell_i \in E} L_i$ number of virtual links. The time complexity for the lines 3-11 of iOrder-bl is dominated by that of line 4 (i.e., iOrder-slot), thus their time complexity is $O((\sum_{\ell_i \in E} L_i)^3)$. Accordingly, it is easy to see that the time complexity for iOrder-bl is $O((\sum_{\ell_i \in E} L_i)^4)$.

IV. Simulation

To gain insight into the impact of link ordering on wireless scheduling, we comparatively study different scheduling algorithms via simulation. We first discuss the simulation methodology and then the simulation results. The insight gained through this simulation study will be verified via testbed-based measurement in Section V.

A. Simulation methodology

We have built a custom simulation package using Matlab. To understand the potential impact that the environmental and network settings have on the behavior of different algorithms, we use networks of different scales and different wireless channel parameters. More specifically, we consider the set $\{2.5, 3, 3.5, 4, 4.5, 5, 5.5, 6\}$ of wireless path loss exponents αs, which represent a wide range of real-world environments [21]; we also set the shadowing variance σ based on measurement data from [21]. We consider 2D Poisson networks with the node distribution density $\lambda = 1$ node$/m^2$ and the background noise power $P_{noise} = -95dBm$. Given an environmental setting (i.e., specific values for α and σ), we use the same transmission power P_{tx} for all the nodes such that every node has, on average, 10 neighbors to which the signal-to-noise-ratio (SNR) is above the threshold $\gamma_t = 5dB$ in the absence of interference. Then we choose the average link length r_0 such that receivers r_0 distant from their transmitters have a SNR of $\gamma_t + \gamma_b$. We set γ_t as $5dB$ to ensure 100% link reliability during scheduling. We call γ_b the *starting SINR budget*, and $\gamma_b = 1dB$ in this paper unless mentioned otherwise;

We consider networks that are deployed in squares with side lengths kr_0, where $k = 5, 7, 9, 11$. For convenience, we denote the network deployed in a square of side length kr_0 as a $k \times k$ network. For $k = 5, 7, 9, 11$, the average number of nodes in a $k \times k$ network is $70, 140, 237$, and 346 respectively in our simulation.

Given an average link length r_0, we consider one-hop unicast where the receiver n_r of a node n_t is the node whose distance to n_t is the closest to r_0, with ties broken at random. (We will study multi-hop convergecast in Section V.) For each node, we consider the case of backlogged traffic as well as online traffic. In the backlogged traffic pattern, each node has m number of packets queued for transmission, where m is a Poisson random variable with mean being 30. In the online traffic pattern, packets arrive at each node in a Poisson manner with an average rate of 0.15 packet/time-slot,[6] and we assume packet arrivals happen at the very beginning of the individual time slots. Backlogged traffic represents the case where nodes need to upload stored data, and network throughput (i.e., total number of packets delivered per time slot in the network) is an important performance metric. Online traffic can represent the case of real-time sensing and control, and packet delivery latency as well as queue length are important performance metrics. When simulating online traffic, we assume that packet queues are managed in a FIFO manner. We set the queue size to be large enough (e.g., 40) to avoid queue overflow.

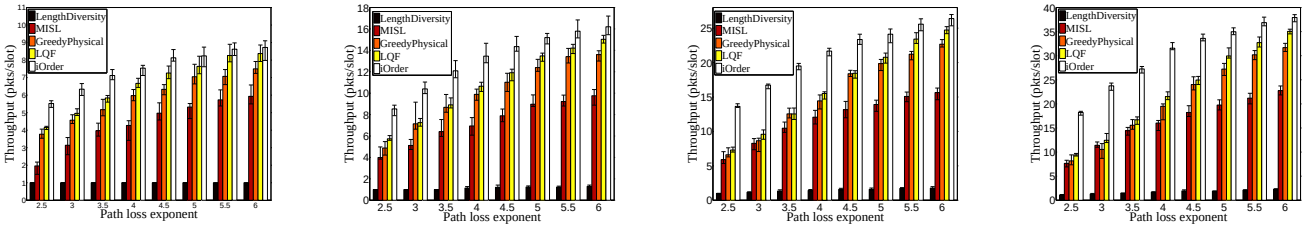
Using the above experiment design, we evaluate the performance of LengthDiversity, MISL, GreedyPhysical, LQF, and iOrder.For each scheduling algorithm and each configuration of environmental, network, and traffic parameters, we repeat the experiment 20 times such that we can analyze the variability and confidence intervals of performance metrics.

B. Simulation results

In what follows, we first discuss the results for backlogged traffic and then for online traffic. The data presented in most of the figures include the medians and their 95%confidence intervals of the corresponding metrics (e.g., throughput and delay).

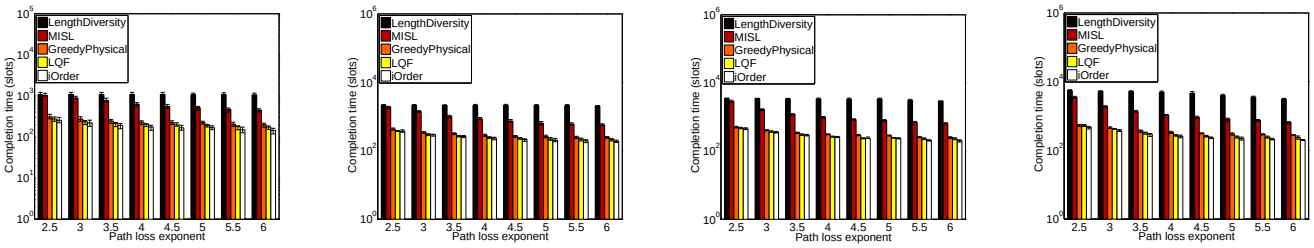
1) Backlogged traffic: For networks of different scales and different wireless path losses, Figure 2 shows the network throughput of different scheduling algorithms. We see that, except for LengthDiversity, there is a clear throughput increase in all the algorithms as network size or path loss exponent increases. This is because larger network size or path loss exponent implies higher degree of spatial reuse possible in scheduling. Among all the algorithms, iOrder always performs better than the other algorithms. For instance, iOrder may double the throughput of LQF in large networks (e.g., 11×11 network) with small path loss exponents (e.g., 2.5 or 3), and iOrder may improve the throughput of LengthDiversity by a factor up to 19.6. Among the existing algorithms, LQF performs better than GreedyPhysical, MISL, and LengthDiversity. One reason for this is because LQF is based on queue length whose stochastic temporal behavior prevents LQF from getting stuck at low-performance slot-schedules. Surprisingly, the throughput of LengthDiversity is very low (even compared with other existing protocols) despite its good asymptotic approximation ratio. For instance, LengthDiversity only allows for one transmission per slot in 5×5 networks and at most 3 concurrent transmissions per slot in 11×11 networks. As discussed in Section III-A, this is due to the large μ value in LengthDiversity that schedules transmissions based on virtual grids of cell side lengths $\mu 2^k$ ($k = 0, 1, \ldots$). The higher throughput in iOrder makes it take less time to complete the transmission of all the backlogged traffic. For instance, Figure 3 shows that it takes the fewest number of time slots to deliver all the backlogged traffic in iOrder.

[6]The average arrival rate of 0.15 packet/slot is chosen so that it does not exceed the capacity of many of the networks we study.



(a) 5×5 networks (b) 7×7 networks (c) 9×9 networks (d) 11×11 networks

Fig. 2. Network throughput in different algorithms



(a) 5×5 networks (b) 7×7 networks (c) 9×9 networks (d) 11×11 networks

Fig. 3. Time taken to deliver all the backlogged traffic

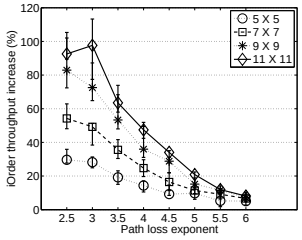
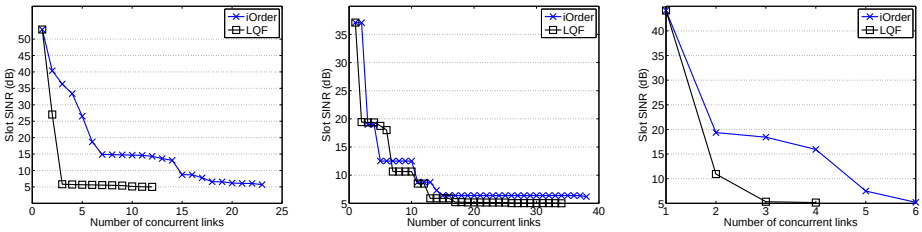


Fig. 4. Throughput increase in iOrder



(a) 11×11 networks, $\alpha = 2.5$ (b) 11×11 networks, $\alpha = 6$ (c) 5×5 networks, $\alpha = 2.5$

Fig. 5. Time series of slot SINR

To understand the optimality of iOrder, we also compute, through exhaustive search, the throughput of the optimal scheduling OPT that maximizes the number of concurrent transmissions while ensuring the required receiver-side SINR. Due to the NP-completeness of optimal scheduling [5], computing optimal schedules takes extremely long time, and it would even take many years to compute the optimal schedule of a 7×7 network. Hence we only compare the throughput of iOrder with that of OPT in 5×5 networks. Figure 6 shows the

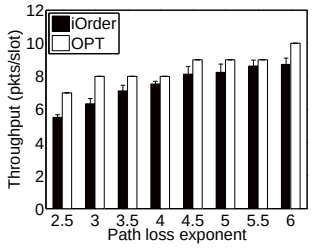


Fig. 6. Network throughput of iOrder and OPT in 5×5 networks

throughput of iOrder and OPT in 5×5 networks with different wireless path losses. We see that iOrder achieves a throughput very close to what OPT enables. When the path loss exponent is 6, for instance, the throughput in iOrder is 93.78% of that in OPT.

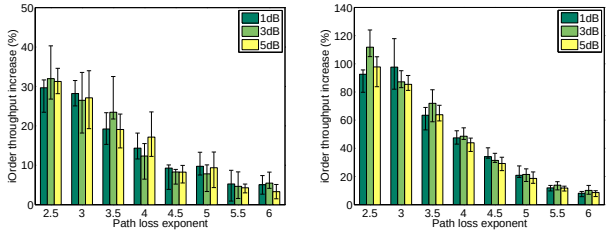
Since LQF performs better than GreedyPhysical, MISL, and LengthDiversity, we only focus on the comparative study of iOrder and LQF hereafter. To gain insight into the impact of link ordering on wireless scheduling in different settings, Figure 4 shows the throughput increase in iOrder, as compared with LQF, in networks of different scales and different path loss exponents. We see that the throughput increase (and thus the importance of optimal link ordering) drops as the path loss exponent α increases. This is because, when adding a new link to a partial slot-schedule, the interference between the new link and the existing links of the slot-schedule decreases as α increases, which makes the interference budget less sensitive to the different choices of new link addition and thus the benefit of optimal ordering/choice of link addition less significant. For a typical 11×11 network with $\alpha = 2.5$ and $\alpha = 6$ respectively, for instance, Figures 5(a) and 5(b) show how the *slot SINR* evolves in LQF and iOrder as new links are added to the schedule of a typical slot, where the slot SINR is defined as the minimum SINR at all the receivers of a slot. We see that, with smaller α, the slot SINR in iOrder decreases much slower than that in LQF. Despite the impact of α, we observe that the

throughput enhancement as a result of optimized link ordering in iOrder is still significant (e.g., up to 115%) for typical indoor and outdoor environments where α is usually less than 4.5 [21]. We also see that the throughput enhancement in iOrder increases with network size. This is because larger networks give more opportunity of spatial reuse and thus more room for optimization. For instance, Figures 5(a) and 5(c) show that, for the same extra SINR budget in iOrder, more concurrent links can be scheduled in larger networks since they provide more schedulable candidates.

Figures 2 and 4 show that the trend how network size affect algorithm performance is monotonic. For conciseness, therefore, we only present data for 5×5 and 11×11 networks hereafter.

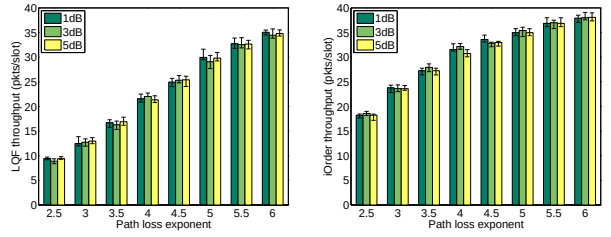
SINR budget. The starting SINR budget γ_b determines the interference that can be tolerated during scheduling without violating application requirement on link reliability. Thus it tends to affect allowable concurrent transmissions. In general, we can control γ_b in two ways: 1) control the link length r_0 (e.g., in routing), and 2) control the transmission power P_{tx} (e.g., in power control). To increase γ_b, for instance, we can decrease r_0 or increase P_{tx}. To understand the potential impact that γ_b may have on the relative goodness between LQF and iOrder as well as the importance of link ordering, we comparatively study the performance of LQF and iOrder when $\gamma_b = 1dB, 3dB$, and $5dB$ respectively.

Figure 7 shows, when γ_b is controlled by changing r_0, the throughput increase in iOrder as compared with LQF. We see that the enhancement in iOrder does not change much at the 95% confidence level. After examining the scheduling data in LQF and iOrder, we find that this is mostly due to the fact that controlling γ_b by changing r_0 does not have statistically significant impact on the throughput of LQF and iOrder (as shown in Figure 8), which is due to the fact that nodes' transmission



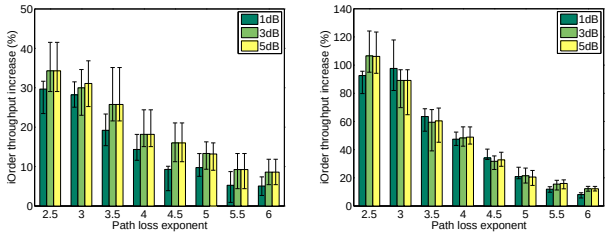
(a) 5×5 networks (b) 11×11 networks

Fig. 7. Impact of SINR budget γ_b: control by r_0



(a) LQF (b) iOrder

Fig. 8. Network throughput in 11×11 networks when γ_b is controlled by r_0

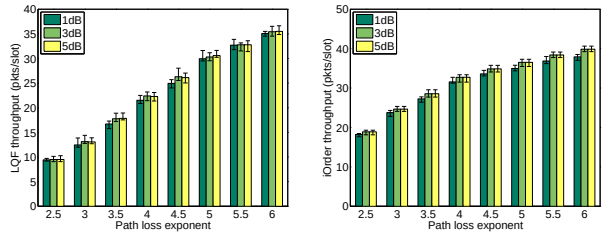


(a) 5×5 networks (b) 11×11 networks

Fig. 9. Impact of SINR budget γ_b: control by P_{tx}

power P_{tx} do not change and thus the interference field does not change significantly. One implication of this fact is that, in routing, a node does not need to intentionally choose shorter links for the purpose of controlling transmission concurrency.

Figure 9 shows, when γ_b is controlled by changing P_{tx}, the throughput increase in iOrder as compared with LQF. We see that, as γ_b increases, the throughput increase in iOrder does not change much at the 95% confidence level. This is again due to the fact that the actual throughput does not change with statistical significance in PRK and iOrder (as shown in Figure 10). Note that this result holds even for

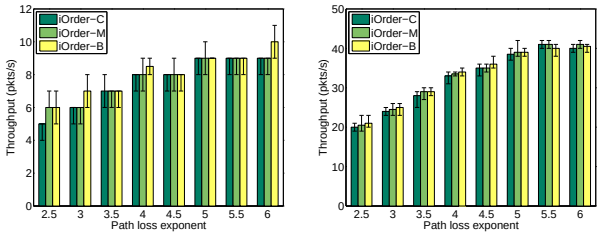


(a) LQF (b) iOrder

Fig. 10. Network throughput in 11×11 network when γ_b is controlled by P_{tx}

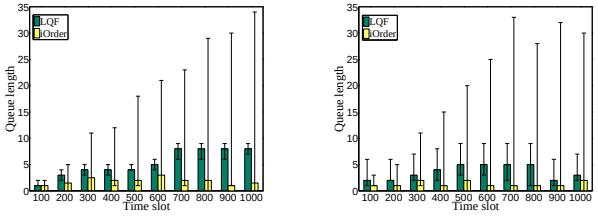
very large γ_b's (e.g., 10dB and 100dB). The reason for this is that, given the same set of concurrent transmitters, increasing their transmission power by the same factor does not lead to significant change of the SINR at the receivers (since the background noise power is usually small compared with the power of the data and interference signal). One implication of this fact is that, in power control, a node does not need to intentionally use higher power level for the purpose of controlling transmission concurrency.

iOrder & starting link location. For the purpose of stabilizing packet queueing, iOrder chooses the link with the highest queue length as the first link of a slot-schedule. It is possible that this is not optimal since the location of the starting link of a slot-schedule may affect the total number of the links that can be scheduled for the slot. To characterize the potential impact of starting link location, we study the performance of three variants of iOrder with different strategies on choosing the starting link of a slot-schedule: *iOrder-C* that chooses the link closest to the geometric center of the network deployment area as the starting link, *iOrder-B* that chooses the link farthest away from the network geometric center as the starting link, and *iOrder-M* that chooses as the starting link the link closest



(a) 5×5 network (b) 11×11 network

Fig. 11. Impact of starting link location on the throughput of iOrder

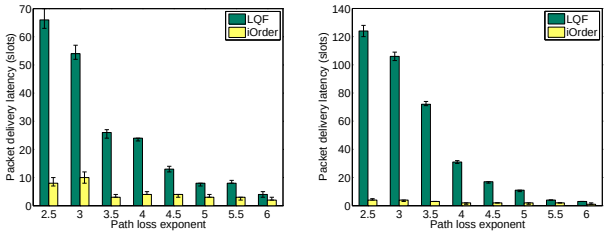


(a) $\alpha = 3.5$ (b) $\alpha = 4.5$

Fig. 13. Time series of median queue length and its 95% confidence interval

to the middle point between the network geometric center and the point in the deployment area that is farthest from the center. Figure 11 shows the performance of these variants of iOrder. We see that iOrder-C tends to enable the lowest throughput, and iOrder-B tends to enable the highest throughput. But the differences between the throughput of these algorithms tend to be small and, in many cases, not statistically significant. This observation can help facilitate distributed implementation of iOrder since we may choose not to worry about starting link location in iOrder scheduling. We will explore this direction in our future work.

2) Online traffic: For online traffic, we measure the delivery latency of a packet as the interval between the packet arrival time at a node and the moment when the node finishes transmitting the packet. Figure 12 shows the delivery latency



(a) 5×5 networks (b) 11×11 networks

Fig. 12. Packet delivery latency

in LQF and iOrder for different network and wireless path loss settings. We see that, in both LQF and iOrder, the median latency decreases with increasing path loss exponent α. This is because, when α increases, the network throughput increases as a result of increased spatial reuse, and thus the median waiting/queueing time for each packet decreases. Compared with LQF, iOrder reduces packet delivery latency significantly, for instance, by a factor up to 24 in 11×11 networks with $\alpha = 3.5$ (as in some typical indoor environments [21]). This shows that iOrder can better support applications such as real-time sensing and control.

To gain insight into the queueing behavior which will also help explain packet delivery latency in LQF and iOrder, we examine the evolution of queue length (i.e., number of queued packets at nodes) in LQF and iOrder. For 5×5 networks, for instance, Figure 13 shows, for a typical indoor and outdoor environment with α being 3.5 and 4.5 respectively, how the queue length evolves over time (measured in time slot number) in LQF and iOrder.[7] We see that iOrder usually has much smaller queue length than LQF does, which, together with the fact that iOrder has higher throughput than LQF, explains why the packet delivery latency is much smaller in iOrder (as shown in Figure 12). The higher throughput in iOrder also supports higher application traffic load while ensuring finite queueing. For instance, Figure 14 shows that, for ensuring finite queueing in 5×5 networks, the maximum supportable per-node packet arrival rate in iOrder is higher than that in LQF.

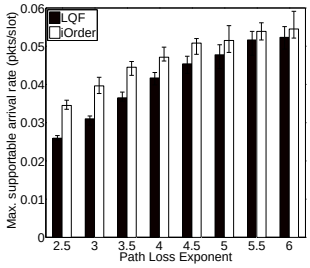


Fig. 14. Maximum supportable per-node packet arrival rate

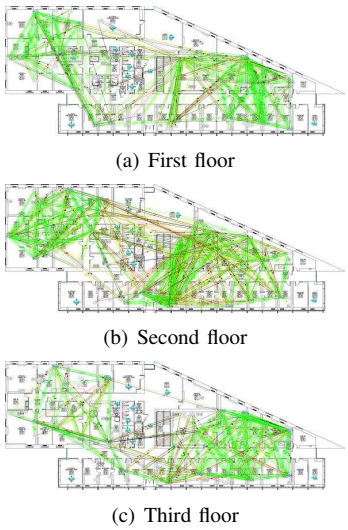
V. Testbed Measurement

Our simulation results show that, by optimizing link ordering in scheduling, iOrder outperforms existing scheduling algorithms. To corroborate these results, we experimentally compare the performance of iOrder and LQF using the *MoteLab* testbed [22].

A. Measurement methodology

We use the MoteLab wireless sensor network testbed at Harvard University [22]. MoteLab is deployed at three floors of the EECS building of Harvard as shown in Figure 15. In our experiments, we use all of the 101 operational Tmote Sky motes, with 32, 39, and 30 motes distributed at the first, second, and third floors respectively. We use two radio transmission powers $0dBm$ and $-3dBm$ (a.k.a. power level 31 and 23 respectively) to generate networks of different connectivity.

[7]Note that the cross-node variation in queue length tends to be more in iOrder than in LQF. This is expected since, unlike LQF which focuses on balancing queue length across nodes, iOrder focuses more on improving spatial reuse and concurrency in scheduling. Nonetheless, the maximum queue length in iOrder is still small (e.g., less than 40) and can be implemented even in resource constrained platforms such as TelosB motes.



(a) First floor

(b) Second floor

(c) Third floor

Fig. 15. *MoteLab* testbed

Part of the input to iOrder and LQF are the radio model, background noise at every node, and strength of signals from any node to every other node. To collect these information for MoteLab, we perform the following experiment: let the 101 motes take turns to be a transmitter one at a time; when a mote is a transmitter, it broadcasts 600 128-byte packets with an inter-packet interval of 100ms (note: each packet transmission takes ∼4ms); while a mote is transmitting packets, every other mote keeps sampling its radio RSSI once every 2ms whether or not it can receive packets from the transmitter, and, if a mote can receive packets from the transmitter, it logs the received packets. We execute the above experiment for both the transmission power of $0dBm$ and $-3dBm$. Using the data collected in these experiments, we can derive the background noise power at each node, the strength of signals from any node to every other node when the transmission power is $0dBm$ or $-3dBm$, and the packet delivery rate (PDR) from any node to every other node as well as the associated SINR. These data also enable us to derive the empirical radio model for the Tmote Sky motes in MoteLab. Figure 16 shows the

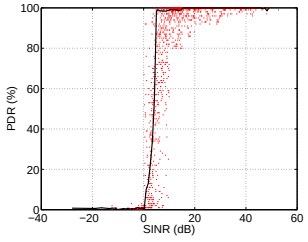
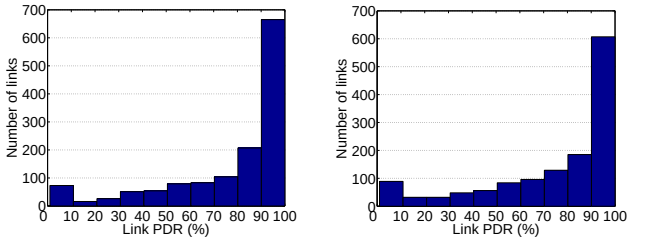


Fig. 16. PDR vs. SINR for the CC2420 radio in MoteLab: scatter plot and mean-PDR curve

measured radio model, as represented by the relation between packet delivery rate (PDR) and SINR at a receiver, for a typical mote in MoteLab. We will use this radio model in our scheduling algorithms for two purposes: 1) to choose the SINR threshold for satisfying a certain link reliability, and 2) to compute the expected PDR for a given SINR at a receiver.

Figure 17 shows the histograms of the PDRs of all the wireless links in MoteLab when the transmission power is $0dBm$ and $-3dBm$, and Figure 18 shows the histogram of background noise power in MoteLab. We see that there is a high degree of spatial variability in link PDRs and background noise power. Thus the testbed enables us to do experiments in non-uniform settings.



(a) Transmission power: $0dBm$ (b) Transmission power: $-3dBm$

Fig. 17. Histogram of link PDRs in MoteLab

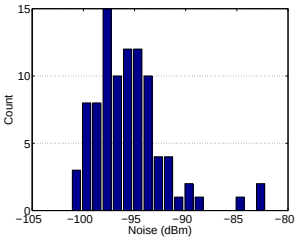
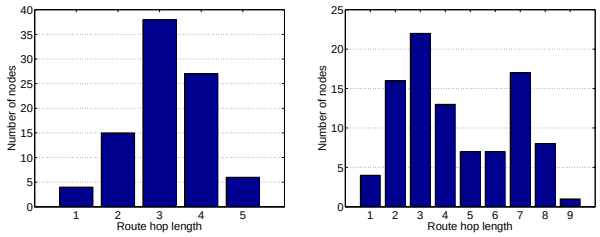


Fig. 18. Histogram of background noise power in MoteLab

To generate the traffic load for scheduling, we consider convergecast[8] where data packets generated by all the nodes need to be delivered to a base station node. More specifically, we let mote #115 at the center of the second floor be the base station to which the remaining 100 motes deliver their packets (mostly via multi-hop paths). Then we build the routing tree by identifying, for each non-base-station mote, a reliable minimum-hop path to the base station where each link of the path has a receiver-side SNR of no less than $\gamma_t + \gamma_b$ in the absence of interference. Similar to simulation, we set γ_t and γ_b as $5dB$ and $1dB$ respectively. Figure 19 shows the histogram



(a) Transmission power: $0dBm$ (b) Transmission power: $-3dBm$

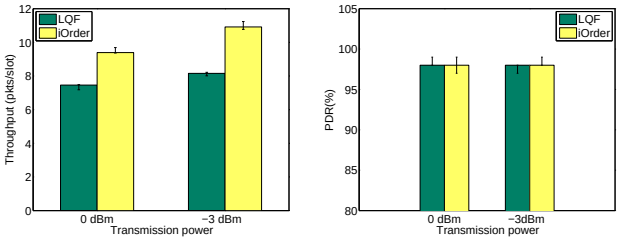
Fig. 19. Histogram of routing hop length

[8]Convergecast enables us to examine properties related to both single-hop transmissions (as in Section IV) and multi-hop data delivery, with the former corroborating observations in simulation and the latter shedding light on the benefits of iOrder in multi-hop, end-to-end data delivery.

of the routing hop length when different transmission power is used. We see that the maximum hop count is 5 and 9 when the transmission power is $0dBm$ and $-3dBm$ respectively.

Given a routing tree, we generate the traffic load as follows: each mote generates 30 packets, and then the number of packets that need to be delivered across a link is the number of packets generated in the subtree rooted at the transmitter of the link. Then the traffic load is used as the input to iOrder and LQF to generate the transmission schedule. Experiment with each schedule is repeated 10 times to gain statistical insight. To experiment with a schedule in MoteLab, we use the TinyOS Flooding Time Synchronization Protocol (FTSP) [23] to synchronize transmissions such that the links in the same time lot of the schedule transmit at the same time; each slot is repeated 30 times before moving onto the next slot so that we can get 30 samples on the transmission status (i.e., success or failure) along each link of the slot to understand the behavior of each slot.

B. Measurement results



(a) Median throughput (b) Mean link PDR

Fig. 20. MoteLab-based measurement

Figure 20(a) shows the network-wide throughput (measured in the total number of packets delivered per time slot) in LQF and iOrder when different transmission powers and thus different routing trees are used. We see that iOrder consistently outperforms LQF, by 22.6% and 28.9% when the transmission power is $0dBm$ and $-3dBm$ respectively. Given that MoteLab is housed in an indoor environment where the path loss exponent $\alpha \approx 3.5$ and that the size (i.e., 101 motes) of our measurement network is between those of the 5×5 and 7×7 simulation networks (which have 70 and 140 nodes respectively), the MoteLab-based measurement data very much agrees with the simulation data shown in Figure 4, thus validating the simulation results. When the transmission power decreases from $0dBm$ to $-3dBm$, the diameter of the network connectivity graph and the depth of the routing tree increases (as shown in Figure 19), and thus the opportunity of optimizing spatial reuse increases. Therefore, the throughput enhancement increases from 22.6% to 28.9% when the transmission power decreases. The higher throughput in iOrder makes it take fewer number of time slots to deliver all the packets to the base station, as shown in Figure 21.

The intention of setting γ_t as $5dB$ is to ensure a $\sim$100% link reliability in scheduling. To verify the correctness of our schedule, Figure 20(b) presents the actual link PDRs during experiments. We see that the links are very reliable and

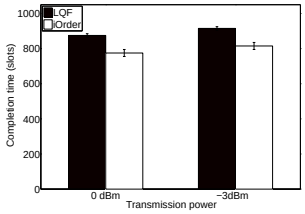


Fig. 21. Time taken to deliver all the packets in MoteLab

have a mean PDR of $\sim$98%, which implies that the TDMA scheduling is able to ensure the required link reliability. From Figures 20(a) and 20(b), we also see that iOrder improves network throughput without sacrificing link reliability.

VI. Discussion: distributed implementation

As a first step towards addressing the ordering effect in wireless scheduling, our objective in this paper is to characterize the impact of link ordering on the throughput of TDMA scheduling, thus we have focused on centralized TDMA scheduling. In terms of implementation, even though distributed scheduling algorithms are preferred in general, centralized scheduling has also found its use in settings of slowly-varying network and traffic conditions. In wireless networked sensing for oil field instrumentation, for instance, the wireless link properties tend to be slowly-varying only, and the monitoring nodes generate packets at fixed, pre-specified frequencies; in these cases, centralized TDMA scheduling (possibly together with frequency scheduling) has been employed in both engineering practice [24] and industry standards such as WirelessHART [2]. That said, for cases where distributed scheduling is more desirable, the scheduling algorithm iOrder can be implemented in a distributed manner, and we discuss the potential approaches as follows.

To implement iOrder in a distributed manner, we need to design mechanisms that generate transmission schedules similar to those in iOrder; more specifically, we need to generate the effect of interference-budget- and queue-length-based scheduling of iOrder. The effect of *interference-budget-based scheduling* is that the SINRs at the receivers of a slot-schedule are close to the required threshold γ_t. This fact lends iOrder to distributed implementation via the physical-ratio-K (PRK) interference model [17], because PRK-based scheduling makes the SINRs at the receivers close to the desired threshold γ_t. The PRK model also integrates the high-fidelity of the physical interference model with the locality of the protocol interference model, thus PRK-based scheduling enables reliable, high-throughput communication while only requiring coordination among nodes close-by (i.e., no global information is needed in PRK-based scheduling) [17], [25].

For the effect of *centralized, queue-length-based scheduling*, Le et al. [7] and Ni et al. [26] have recently developed distributed, queue-length-based priority scheduling mechanisms that, only with localized, asynchronous coordination among neighboring nodes, achieve a performance close to the centralized, queue-length-based scheduling. In iOrder, the link with the highest queue length is selected as the first link of

a slot-schedule, but, as we have discussed in Section IV-B, the number of links that can be scheduled for a time slot is *insensitive* to the location of the first link picked for the slot.

From the above observations, we believe that iOrder can be implemented in a distributed manner by leveraging the PRK model [17] and the distributed queue-length-based priority scheduling [7], [26], and we will investigate this in detail in our future work.

VII. Related work

As a basis of wireless scheduling, different interference models have been studied extensively [27], [28], [29], [30]. It was found that the physical interference model (a.k.a. the SINR model) is more accurate than the basic protocol interference model and can enable higher scheduling performance in general [27], [28]. For distributed protocol design, we also discovered that it is possible to effectively instantiate the protocol interference model via local feedback control [17]. Focusing on gaining insight into the impact of link ordering on wireless scheduling instead of distributed protocol design, we employ the physical model in this paper for the sake of its high accuracy.

Different interference-oriented TDMA scheduling algorithms have been proposed in the literature. They include LQF and its variants [7], [8], [31], [6], [32], [9], GreedyPhysical and its variants [10], [11], MISL and its variants [12], [33], [34], as well as LengthDiversity [5]. The throughput and delay performance [31], [35], [36] as well as the distributed implementation [31], [37], [38] of these algorithms have also been studied, and scheduling based on dominant interferers has been considered by Badia et al. [32]. Nonetheless, these work have not focused on the impact of link ordering on interference-limited scheduling; the analysis of these algorithms has also mostly focused on the asymptotic behavior without characterizing the impact of potentially large constants in the analysis. Our study fills this gap by proposing the algorithm iOrder for optimizing link ordering in wireless scheduling and by examining the detailed behavior of different protocols through analysis, simulation, and testbed-based measurement.

Interference-oriented distributed scheduling has also been well studied [39], [40], [41], [42], [43], [44], [45], but the existing work has mostly focused on contention resolution and collision avoidance without explicitly optimizing network-wide spatial reuse. Yang et al. [46] have studied how to induce spatial clustering for the purpose of improving spatial reuse in densely deployed spread-spectrum networks. Spatial clustering may not be generically applicable since it is most useful only when network deployment is dense and the spreading factor of the spread spectrum radios is large (e.g., a spreading factor of 512 was considered in [46]). The performance of spatial clustering has only been compared with the traditional CSMA/CA random access schemes too.

Scheduling has also been considered together with transmission power control in wireless networks [47], [48], where nodes adapt their transmission power to further increase network capacity and to reduce delay. Focusing on characterizing the ordering effect in wireless scheduling, we have assumed fixed transmission power in this paper. How to leverage controllable transmission power in addressing the ordering effect in scheduling will be an interesting topic to pursue, but detailed study of this is beyond the scope of this paper. Full-duplex radio [49] has also been developed recently, and it can transmit and receive at the same time by eliminating self-interference at transmitters. With full-duplex radios, interference between concurrent transmitters still need to be controlled; detailed study of this is an interesting topic to pursue but beyond the scope of this paper.

VIII. Concluding remarks

Co-channel interference control is important for the reliability and predictability of wireless networks. Towards understanding the importance of explicitly addressing the ordering effect in interference-limited wireless scheduling, we have formulated the notion of interference budget to characterize a schedule's tolerance of additional interference, and we have designed the algorithm iOrder that schedules links in a decreasing order of the enabled interference budget. Through analysis, simulation, and testbed-based measurement, we have demonstrated the benefits of explicitly addressing the ordering effect by showing the significantly better performance of iOrder as compared with the well-known algorithms such as LQF, GreedyPhysical, MISL, and LengthDiversity. By discovering and understanding the surprisingly low performance of LengthDiversity, our study has also demonstrated the importance of examining the constant factors involved in asymptotic analysis. As a side result, our study has characterized the relative goodness of the existing algorithms, which is of independent interest. The findings of this paper shed new insight into the behavior of wireless scheduling and, via algorithm iOrder, constructively shows the benefit of optimizing link ordering in scheduling. Therefore, the findings of this paper open up new avenues for future research and for optimizing wireless network performance; one of the future directions is to investigate mechanisms for realizing iOrder (or interference budget optimization in general) in a distributed manner, and we will explore the methods discussed in Section VI as well as their possible variants.

References

[1] K. Chintalapudi and L. Venkatraman, "On the design of mac protocols for low-latency hard real time latency applications over 802.15.4 hardware," in *ACM/IEEE IPSN*, 2008.

[2] D. Chen, M. Nixon, and A. Mok, *Wireless*HART*: Real-Time Mesh Network for Industrial Automation*. Springer, 2010.

[3] ISA SP100.11a, http://www.isa.org//MSTemplate.cfm?MicrositeID=1134\&CommitteeID=6891.

[4] H. Zhang, A. Arora, and P. Sinha, "Link estimation and routing in sensor network backbones: Beacon-based or data-driven?" *IEEE Transactions on Mobile Computing*, vol. 8, no. 5, pp. 653 – 667, May 2009.

[5] O. Goussevskaia, Y. A. Oswald, and R. Wattenhofer, "Complexity in geometric SINR," in *ACM MobiHoc*, 2007.

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

[7] L. B. Le, E. Modiano, C. Joo, and N. B. Shroff, "Longest-queue-first scheduling under SINR interference model," in *ACM MobiHoc*, 2010.

[8] C. Joo, X. Lin, and N. B. Shroff, "Understanding the capacity region of the greedy maximal scheduling algorithm in multi-hop wireless networks," in *IEEE INFOCOM*, 2008.

[9] D. M. Blough, S. Das, G. Resta, and P. Santi, "A framework for joint scheduling and diversity exploitation under physical interference in wireless mesh networks," in *IEEE MASS*, 2008.

[10] G. Brar, D. M. Blough, and P. Santi, "Computationally efficient scheduling with the physical interference model for throughput improvement in wireless mesh networks," in *ACM MobiCom*, 2006.

[11] W. Wang, Y. Wang, X.-Y. Li, W.-Z. Song, and O. Frieder, "Efficient interference-aware TDMA link scheduling for static wireless networks," in *AMC MobiCom*, 2006.

[12] P.-J. Wan, X. Jia, and F. Yao, "Maximum independent set of links under physical interference model," in *WASA*, 2009.

[13] T. Rappaport, *Wireless Communications*. Prentice-Hall, Upper Saddle River, NJ, 2002.

[14] CC 2420 radio, http://focus.ti.com/docs/prod/folders/print/cc2420.html.

[15] IEEE 802.15.4, "Wireless medium access control (MAC) and physical layer (PHY) specifications for low-rate wireless personal area networks (WPANs)," IEEE Std 802.15.4-2006, 2006.

[16] G. Zhou, T. He, S. Krishnamurthy, and J. Stankovic, "Models and solutions for radio irregularity in wireless sensor networks," *ACM Transactions on Sensor Networks*, vol. 2, no. 2, 2006.

[17] H. Zhang, X. Che, X. Liu, and X. Ju, "Adaptive instantiation of the protocol interference model in wireless networked sensing and control," *ACM Transactions on Sensor Networks (TOSN)*, vol. 10, no. 2, 2014.

[18] D. S. Hochbaum, *Approximation Algorithms for* NP*-hard Problems*. PWS Publishing Company, 1997.

[19] D. Stoyan, W. S. Kendall, and J. Mecke, *Stochastic Geometry and its applications*. Wiley, 1995.

[20] M. Haenggi and R. K. Ganti, "Interference in large wireless networks," *Foundations and Trends in Networking*, vol. 3, no. 2, pp. 127–248, 2009.

[21] K. Sohrabi, B. Manriquez, and G. J. Pottie, "Near ground wideband channel measurement," in *IEEE VTC*, 1999.

[22] G. Werner-Allen, P. Swieskowski, and M. Welsh, "Motelab: A wireless sensor network testbed," in *IEEE/ACM IPSN/SPOTS*, 2005.

[23] M. Maroti, B. Kusy, G. Simon, and A. Ledeczi, "The flooding time synchronization protocol," in *ACM SenSys*, 2004.

[24] Dust Networks, "Time synchronized mesh protocol (TSMP)," http://www.dustnetworks.com/cms/sites/default/files/TSMP_Whitepaper.pdf.

[25] H. Zhang, C. Li, X. Liu, Y. Chen, X. Che, F. Lin, L. Y. Wang, and G. Yin, "PRK-based scheduling for predictable link reliability in wireless networked sensing and control," Wayne State University, Tech. Rep. DNC-TR-14-01 (https://sites.google.com/site/dnctrs/DNC-TR-14-01.pdf), 2014.

[26] 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," in *IEEE INFOCOM*, 2010.

[27] R. Maheshwari, S. Jain, and S. Das, "A measurement study of interference modeling and scheduling in low-power wireless networks," in *ACM SenSys*, 2008.

[28] T. Moscibroda, R. Wattenhofer, and Y. Weber, "Protocol design beyond graph based models," in *ACM HotNets*, 2006.

[29] D. Chafekar, V. A. Kumar, M. V. Marathe, S. Parthasarathy, and A. Srinivasan, "Approximation algorithms for computing capacity of wireless networks with sinr constraints," in *IEEE INFOCOM*, 2008.

[30] Y. Shi, Y. T. Hou, J. Liu, and S. Kompella, "How to correctly use the protocol interference model for multi-hop wireless networks," in *ACM MobiHoc*, 2009.

[31] M. Leconte, J. Ni, and R. Srikant, "Improved bounds on the throughput efficiency of greedy maximal scheduling in wireless networks," in *ACM MobiHoc*, 2009.

[32] L. Badia, A. Erta, L. Lenzini, F. Rossetto, and M. Zorzi, "A physical model scheduler for multi-hop wireless networks based on local information," in *IEEE MASS*, 2008.

[33] G. Pei and V. S. A. Kumar, "Distributed approximation algorithms for maximum link scheduling and local broadcasting in the physical interference model," in *IEEE INFOCOM*, 2013.

[34] T. Kesselheim and B. Vocking, "Distributed contention resolution in wireless networks," in *DISC*, 2010.

[35] K. Kar, X. Luo, and S. Sarkar, "Delay guarantees for throughput-optimal wireless link scheduling," in *IEEE INFOCOM*, 2009.

[36] M. J. Neely, "Delay analysis for maximal scheduling with flow control in wireless networks with bursty traffic," *IEEE/ACM Transactions on Networking*, vol. 17, no. 4, pp. 1146–1159, 2009.

[37] A. Gupta, X. Lin, and R. Srikant, "Low-complexity distributed scheduling algorithms for wireless networks," *IEEE/ACM Transactions on Networking*, vol. 17, no. 6, 2009.

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

[39] L. X. Cai, L. Cai, X. Shen, J. W. Mark, and Q. Zhang, "MAC protocol design and optimization for multi-hop ultra-wideband networks," *IEEE TWC*, vol. 8, no. 8, 2009.

[40] Y. Yi, G. D. Veciana, and S. Shakkottai, "On optimal MAC scheduling with physical interference model," in *IEEE INFOCOM*, 2007.

[41] G. Brar, D. M. Blough, and P. Santi, "The SCREAM approach for efficient distributed scheduling with physical interference in wireless mesh networks," in *ICDCS*, 2008.

[42] G. Zhou, T. He, J. A. Stankovic, and T. Abdelzaher, "RID: Radio interference detection in wireless sensor networks," in *IEEE INFOCOM*, 2005.

[43] M. Sha, G. Xing, G. Zhou, S. Liu, and X. Wang, "C-MAC: Model-driven concurrent medium access control for wireless sensor networks," in *IEEE INFOCOM*, 2009.

[44] M. Cesana, D. Maniezzo, P. Bergamo, and M. Gerla, "Interference aware (IA) MAC: an enhancement to ieee 802.11b DCF," in *IEEE VTC*, 2003.

[45] P. Wang and W. Zhuang, "A collision-free MAC scheme for multimedia wireless mesh backbone," *IEEE Transactions on Wireless Communications*, vol. 8, no. 7, 2009.

[46] X. Yang and G. de Veciana, "Inducing spatial clustering in MAC contention for spread spectrum ad hoc networks," in *ACM MobiHoc*, 2005.

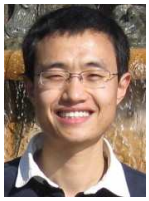
[47] T. Moscibroda and R. Wattenhofer, "The complexity of connectivity in wireless networks," in *IEEE INFOCOM*, 2006.

[48] T. Moscibroda, Y. A. Oswald, and R. Wattenhofer, "How optimal are wireless scheduling protocols," in *IEEE INFOCOM*, 2007.

[49] M. Jain, J. I. Choi, T. Kim, D. Bharadia, S. Seth, K. Srinivasan, P. Levis, S. Katti, and P. Sinha, "Practical, real-time, full duplex wireless," in *ACM MobiCom*, 2011.

Xin Che (S'10) received his B.S. and M.S. in Electrical Engineering as well as his B.S. in Computer Science from Huazhong University of Science and Technology. He is currently pursuing his Ph.D. degree in Computer Science at Wayne State University. His primary research interests lie in modeling, algorithmic, and systems issues in wireless, embedded, and sensor networks.

Hongwei Zhang (S'01-M'07-SM'13/ACM S'01-M'07) received his B.S. and M.S. degrees in Computer Engineering from Chongqing University, China and his Ph.D. degree in Computer Science and Engineering from The Ohio State University. He is currently an associate professor of computer science at Wayne State University. His primary research interests lie in the modeling, algorithmic, and systems issues in wireless, vehicular, embedded, and sensor networks. His research has been an integral part of several NSF, DARPA projects such as the KanseiGenie and ExScal projects. He is a recipient of the NSF CAREER Award. (URL: http://www.cs.wayne.edu/~hzhang).



Xi Ju received his M.S. and Ph.D. degrees in Computer Science from Southeast University, China. He is currently a visiting scholar of computer science at Wayne State University. His research focuses on wireless, vehicular, and embedded networked sensing. He has been involved in the development of the Chinese Next Generation Internet and the US NSF GENI project.