Fair and Efficient Scheduling in Wireless Networks with Successive Interference Cancellation

Mohsen Mollanoori and Majid Ghaderi
Department of Computer Science, University of Calgary
Emails: {mmollano, mghaderi}@ucalgary.ca

Abstract—**This paper considers the problem of uplink scheduling in wireless networks supporting *successive interference cancellation (SIC)* at the physical layer. By allowing concurrent interfering transmissions, SIC enables multi-packet reception at the receiver resulting in increased network throughput. Specifically, we consider *maximum throughput scheduling* and *proportional fair scheduling* problems and study optimal and heuristic algorithms for these problems. We prove that the maximum throughput scheduling problem is NP-hard and develop a throughput efficient polynomial time greedy algorithm for the problem. While being throughput efficient, the maximum throughput scheduling can lead to highly unfair rates among the users. The proportional fair scheduling, on the other hand, is not throughput optimal but achieves proportional fairness among the users. For scheduling multiple users in a single time-slot, we show that there exists an algorithm that solves the proportional fair scheduling problem in polynomial time. For scheduling in multiple time-slots, we develop a greedy algorithm that computes a highly fair schedule in polynomial time. Numerical results are also provided to show the utility and efficiency of the proposed scheduling algorithms in various simulated networks.**

I. Introduction

Wireless communications are subject to interference and noise. While increasing the transmission power can help reduce the noise effect and hence increase the transmission rate, it cannot mitigate the effect of interference. On the contrary, increasing the transmission power, when not carefully controlled, can increase the interfere among nearby nodes resulting in reduced network throughput. While noise is usually structureless, interference has a specific structure as it is caused by other transmissions in the network or nearby networks. With *interference cancellation*, the structured nature of interference is exploited in order to increase the throughput of wireless networks.

Scheduling mechanisms are widely used in wireless networks to prevent multiple simultaneous transmissions thereby eliminating interference. While the lack of interference is a desirable feature from the perspective of individual users, preventing simultaneous transmissions results in a smaller network throughput. *Successive interference cancellation (SIC)* is a multiuser detection technique that can greatly increase the throughput of wireless networks by allowing simultaneous transmissions. In contrast to the traditional *single user at a time* transmission paradigm, using this physical layer technique, multiple users can simultaneously transmit if a proper transmission rate is assigned to each user.

While conventional receivers treat interference as random noise, a SIC receiver treats the interference as a structured signal and tries to decode and remove it from the original signal [1]. In this technique, the receiver, decodes one of the signals first considering the rest of the signals as noise. After decoding the first signal, it reconstructs the corresponding analog signal and subtracts it from the original composite signal. At this stage, the remaining signal is free from the interference of the first decoded signal. The process continues until all the signals are decoded. Since at every stage, the remaining signals are treated as noise, the maximum rate achievable by a user depends not only on its received signal power but also on the *order* at which its signal is decoded.

The key advantage of SIC compared to other multi-user detection techniques such as *joint detection* [2] is that SIC receivers are architecturally similar to traditional non-SIC receivers in terms of hardware complexity and cost [3]. This makes SIC more practical for implementation in wireless systems. A SIC receiver uses the same decoder to decode the composite received signal at different stages of decoding. As a result, neither a complicated decoder nor multiple antennas is required to increase the throughput of the network [1]. It is also known that other multiple access techniques such as CDMA and OFDMA are no more efficient than SIC [4, Ch. 6].

SIC can be employed for both uplink [5] and downlink [6] transmissions. However, at the uplink direction, SIC is able to achieve a higher throughput since the total received power is higher. In this paper, we study the problem of uplink scheduling with successive interference cancellation. Although, scheduling and MAC issues have been extensively studied in traditional wireless networks [7], there are only a few works that have explicitly considered interference cancellation.

In the broader context of multi-packet reception, there exist several work on scheduling and MAC protocols [8]–[10]. Nevertheless, non of them are applicable to SIC-based networks. For instance, Zhao and Tong [8] develop a MAC protocol with multi-packet reception capability assuming that a *reception matrix* is available for the network. A reception matrix specifies the probability that k packets are successfully received when n overlapping transmissions are made in the network. While this model provides a useful abstraction for a general multi-packet reception network, it does not accurately capture the underlying mechanisms of SIC-based networks such as selection of the transmission rates. Other works on multi-packet reception, such as [9] and [10], assume even a simpler model in which a receiver is capable of receiving up to k packets simultaneously regardless of the transmission rates and channel conditions. The closest work to ours is due to Kumaran and Qian [11] where the authors consider the uplink scheduling problem with SIC. However, their work differs from ours in that they only consider the case of scheduling multiple transmissions in a *single* time-slot in order to maximize throughput.

Our main contributions in this paper include the following:

- We mathematically model *maximum throughput scheduling (MTS)* and *proportional fair scheduling (PFS)* problems in wireless networks with SIC. Our models are reasonably abstract, yet capture the key features of SIC-based systems.
- We prove that the maximum throughput scheduling problem is NP-hard by reducing the problem to the well-known *number partitioning problem*. A greedy algorithm is then developed that computes a throughput efficient schedule in polynomial time.
- For scheduling multiple users in a single time-slot, we show that there exists a polynomial time algorithm for the proportional fair scheduling problem. For scheduling in multiple time-slots, we develop a greedy algorithm that computes a highly fair schedule in polynomial time.

The rest of the paper is organized as follows. In Section II, we describe our system model and assumptions. Section III studies the maximum throughput scheduling problem, while Section IV is devoted to the proportional fair scheduling problem. Numerical results are presented in Section V. Our concluding remarks are discussed in Section VI.

II. System Model and Assumptions

A. Network Model

The network consists of a set of wireless devices (referred to as *nodes* or *users*) communicating with a single receiver (*e.g.*, an access point in a wireless LAN or a base station in a cellular network). The time is divided into *scheduling frames*. Scheduling is done once every scheduling frame, at the beginning of the frame (this is similar to the scheduling structure of WiMAX networks [12]).

In every scheduling frame F, a set of users $N_F = \{1, 2, \ldots, n_F\}$ are scheduled for uplink transmission. Every scheduling frame is divided into k time-slots. The aim of the scheduler is to schedule the set of nodes N_F in the k times-lots so that a *system utility function* is maximized. In this work, we consider two utility functions, namely the sum of the user rates and the sum of the logarithm of the user rates. A detailed description of these utility functions and the justification for considering them is presented in Section II-E.

We only consider the case of $|N_F| = n_F > k$, since the solution for the case of $n_F \leq k$ is trivial. Each node is scheduled exactly once in a scheduling frame, thus more than one node might be scheduled in a time-slot (recall that scheduling simultaneous transmissions is allowed with SIC).

In an ideal world, to maximize the system throughput, all nodes have to be scheduled in a single time slot [4]. However, due to practical limitations, only a few overlapping signals can be decoded successfully [1]. One reason is that the decoding time in SIC linearly increases in the number of overlapping signals. The linear decoding time causes practical difficulties with large number of users [4]. In our model, such limitations can be readily captured by controlling k and n_F.

B. Throughput Model

The set of received powers of users at the receiver is denoted by $P_F = \{p_{1,F}, p_{2,F}, \ldots, p_{n,F}\}$, where $p_{i,F}$ is the received power of user i at scheduling frame F. For notational simplicity and wherever clear, we will omit the subscript F from the equations. Note that we do not consider any specific signal propagation model, hence our results are very general. Without loss of generality, we assume $p_1 < p_2 < \cdots < p_n$.

We define the *throughput* of a user as the maximum error-free rate achievable by that user, and consider the Shannon capacity formula in our throughput analysis.

Definition 1 (Single user throughput): The shannon capacity function $\tau : R^+ \times R^+ \rightarrow R^+$ is defined as follows:

$$\tau(p, N) = \log\left(1 + \frac{p}{N}\right) \; bits/s/Hz, \tag{1}$$

where p is the received power, and N is the noise plus interference power at the receiver.

C. Interference Cancellation

Let X_i denote the received signal of user i at the receiver. Assume that $m \geq 1$ users simultaneously transmit in the considered time slot. Then the composite signal received at the receiver is expressed as

$$X = \sum_{i=1}^{m} X_i + Z, \tag{2}$$

where Z denotes the noise at the receiver. Note that the amplitude and phase of transmitted signals is affected by the wireless channel. The received signal X_i includes such effects. Let p_i and N_0 denote, respectively, the signal power of X_i and noise power at the receiver.

A SIC receiver decodes the composite signal X in different stages. Assume that the signals are decoded by the receiver in the order $X_m, X_{m-1}, \ldots, X_1$. That is, when decoding X_m, the rest of the signals are treated as noise. Therefore, the maximum rate achievable by user m is given by $\tau(p_m, N_0 + \sum_{i=1}^{m-1} p_i)$, whereas the maximum achievable rate by user 1 is given by $\tau(p_1, N_0)$. Since the interference power for each signal X_i depends on the order of decoding, it is apparent that the throughput achieved by each user depends not only on its corresponding received power but also on the order at which it is decoded.

To cancel out the signal X_i from the composite signal X, the SIC receiver first encodes the decoded bits using the same modulation technique used at the transmitter of user i to reproduce the analog signal transmitted by the user. Then, it applies the channel transfer function to the reproduced analog signal to obtain an estimation of X_i. Obviously, the receiver should have an accurate estimation of the channel transfer function, otherwise, it cannot reconstruct X_i correctly [1].

D. Sum Throughput

We define the *sum throughput* of a set of received signal powers as the maximum rate achievable in a single time slot subject to some noise power N_0.

Definition 2 (Sum throughput): For a given vector of received powers $P = \langle p_1, \ldots, p_m \rangle$, and some noise power N_0, the sum throughput function $\xi : R \times R^m \to R$ is defined as

$$\xi\left(N_0, \langle p_1, \ldots, p_m \rangle\right) = \sum_{i=1}^{m} \tau\Big(p_i, N_0 + \sum_{j=1}^{i-1} p_j\Big) . \qquad (3)$$

The sum throughput function defines the throughput of the system in a single time-slot. While, as we have already seen, the throughput of individual users depends on the order of decoding, the sum throughput of a set of users is independent of the order of decoding. The following lemma states this property.

Lemma 1: The order of decoding, does not affect the sum throughput. More precisely, we have the following result

$$\xi\left(N_0, \langle p_1, \ldots, p_m \rangle\right) = \tau\Big(\sum_{i=1}^{m} p_i, N_0\Big) . \qquad (4)$$

Proof: The proof is simply done by expanding the right hand side of (3) and using well-known properties of the logarithm function (please refer to our technical report for details [13]). ■

Since the order of decoding does not change the sum throughput, thereafter, instead of writing $\xi(N_0, \langle p_1, \ldots, p_m \rangle)$ we will use the notation $\xi(N_0, \{p_1, \ldots, p_m\})$. In general, p_i's can be any real number. However, in practice, only a finite set of power levels are implemented. Thus, p_i's can be properly scaled and represented by positive integers. Without loss of generality, in the remaining of the paper, we assume that the set of powers are positive integers. Although, our results hold without this assumption, it simplifies some of the proofs and derivations presented in Sections III and IV.

E. Proportional Fairness

In addition to the system throughput, fairness is an important factor in wireless networks. Fairness can be defined in several ways. In this paper, we consider *proportional fairness* [14], which is widely implemented in existing wireless systems [7]. In general, the system throughput is affected by the particular consideration of fairness in the system. While some fairness criteria may sacrifice the system throughput for fairness, the proportional fairness achieve a reasonable tradeoff between fairness and throughput [7].

Definition 3 (Proportional fairness): Assume that schedule Γ gives the throughput vector $\nu = \langle \nu_1, \ldots, \nu_m \rangle$, where ν_i denotes the throughput of user i. A schedule Γ^* with throughput vector $\nu^* = \langle \nu_1^*, \ldots, \nu_m^* \rangle$ is proportionally fair if and only if the following condition holds for any other schedule Γ with throughput vector ν:

$$\sum_{i=1}^{m} \frac{\nu_i - \nu_i^*}{\nu_i^*} \leq 0 . \qquad (5)$$

It has been shown that a proportionally fair schedule maximizes the sum of users' utilities (*i.e.*, the system utility) where the utility of each user is defined as the logarithm of its throughput [14]. The following definition and lemma states this property formally.

Definition 4 (Fairness index): For a given sequence of received powers $P = \langle p_1, \ldots, p_m \rangle$, and some noise power N_0, the proportional fairness index $\phi : R \times R^n \to R$ is defined as

$$\phi\left(N_0, \langle p_1, \ldots, p_m \rangle\right) = \sum_{i=1}^{m} \log\left(\tau\Big(p_i, N_0 + \sum_{j=1}^{i-1} p_j\Big)\right) . \qquad (6)$$

Lemma 2: A schedule Γ is proportionally fair if it maximizes the fairness index ϕ.

Please see [14] for a proof.

III. Maximum Throughput Scheduling

A. Problem Formulation

In this section, we study the problem of maximum throughput scheduling (MTS) in a network with SIC. From Lemma 1, we already know that the order of decoding in a single time slot does not affect the sum throughput. Consequently, for scheduling in multiple time slots, we only need to assign the users to the proper time slots in order to maximize the throughput (*i.e.*, we don't need to specify the order of decoding in each time slot). Later, we will show that MTS, even for the special case of having only two time slots, is NP-hard.

In the maximum throughput scheduling problem, we want to schedule m nodes in k time slots so that: (i) every node is scheduled exactly once, and (ii) the sum throughput (see definition 3) over all time slots is maximized. The following definition states the problem formally.

Problem 1 (MTS): For a given set of received powers $P = \{p_1, \ldots, p_m\}$, an integer $1 \leq k < m$, and some noise power N_0, partition P into k subsets $P_1, P_2, \ldots, P_k$ such that $\cup_{i=1}^{k} P_i = P$ and $\forall i, j\ P_i \cap P_j = \emptyset$, so that $\sum_{i=1}^{k} \xi(N_0, P_i)$ is maximized.

B. Computational Complexity

Lemma 3: Problem 1 is NP-hard.

Proof: To prove the NP-hardness of Problem 1 we will show that there exists a polynomial reduction from the well-known *number partitioning* problem to Problem 1. First, we define the number partitioning problem.

Problem 2 (Number partitioning): Given a set of positive integers $\{a_1, \ldots, a_n\}$, find a subset $A \subset \{1, 2, \ldots, n\}$ so that the following is minimized:

$$E(A) = \Big| \sum_{i \in A} a_i - \sum_{j \notin A} a_j \Big| . \qquad (7)$$

Problem 2 is a well-known NP-complete problem [15].

For a given set of positive integers $P = \{p_1, \ldots, p_n\}$, we will show that the solution of Problem 1 for an arbitrary $N_0 > 0$ and $k = 2$ is the same as the solution of the number partitioning problem. That is, if P is partitioned into

P_1 and P_2 so that $\xi(N_0, P_1) + \xi(N_0, P_2)$ is maximized, then $|(\sum_{a\in P_1} a) - (\sum_{b\in P_2} b)|$ is minimized:

$$\begin{aligned}\xi(N_0, P_1) &+ \xi(N_0, P_2) \\ &= \tau\Big(N_0, \sum_{a\in P_1} a\Big) + \tau\Big(N_0, \sum_{b\in P_2} b\Big) \\ &= \log\Big(\frac{(N_0 + \sum_{a\in P_1} a)(N_0 + \sum_{b\in P_2} b)}{N_0^2}\Big) .\end{aligned} \tag{8}$$

It is well known that if the summation of two numbers is constant, their product is maximized when their difference is minimized. Since $(N_0 + \sum_{a\in P_1} a) + (N_0 + \sum_{b\in P_2} b) = 2N_0 + \sum_i p_i$ is constant, (8) is maximized when $|(\sum_{a\in P_1} a) - (\sum_{b\in P_2} b)|$ is minimized. ■

C. Efficient Greedy Scheduling

Since the MTS problem is NP-hard, we develop an efficient polynomial time greedy algorithm for it called *GreedyMax*. Alg. 1 concisely describes GreedyMax. The algorithm, first sorts the received powers in a descending order. Then, starting from the node with the largest received power, it assigns the node to the time slot that maximizes the system throughput that can be achieved so far. Finally, it sorts the nodes in each time slot with respect to their received powers in an ascending order. The purpose of the final sorting is to make the algorithm as fair as possible without reducing the sum throughput of the system (as stated in Lemma 1).

Alg. 1 GreedyMax

```
$P \leftarrow$ list of received powers
Sort $P$ in descending order
$S \leftarrow \emptyset$
for $i \leftarrow 0$ to $length(P) - 1$ do
    Schedule $P_i$ at the time slot that maximizes $\xi(N_0, S)$
end for
Sort $P_i$'s in each time slot of $S$ in ascending order
return $S$
```

IV. Proportional Fair Scheduling

A. Problem Formulation

In this section, we study the problem of proportional fair scheduling (PFS) with SIC. This problem is similar to the MTS problem, but the objective function is different. The following formally defines the problem.

Problem 3 (PFS): Given a set of received powers $P = \{p_1, \ldots, p_m\}$, an integer $1 \le k < m$, and some noise power N_0, partition P into k vectors $P_1, \ldots, P_k$ so that $\cup_{i=1}^k P_i = P$ and $\forall i, j\ P_i \cap P_j = \emptyset$ in order to maximize $\sum_{i=1}^k \phi(N_0, P_i)$.

Since the order of decoding affects the fairness index $\phi(.)$, we partition the set P into vectors, not sets. PFS, in contrast to the MTS problem, not only assigns the users to time slots but also specifies the order of decoding in each time slot. We next prove that in PFS, the strongest signal is decoded first, the second strongest signal is decoded next, and so on until the weakest signal is decoded. We show that any other decoding sequence is not proportionally fair.

Lemma 4: Assume $\{P_1, \ldots, P_k\}$ is a solution of problem 3, where $P_i = \langle p_1^i, p_2^i, \ldots, p_{n_i}^i \rangle$. The following holds

$$p_1^i \le p_2^i \le \ldots \le p_{n_i}^i . \tag{9}$$

Note that (9) is a necessary but not sufficient condition.

Proof: The proof is by contradiction. Assume for some index j, $p_j^i > p_{j+1}^i$. It can be shown that [13]:

$$\begin{aligned}&\phi(N_0, \langle p_1^i, \ldots, p_j^i, p_{j+1}^i, \ldots, p_{n_i}^i \rangle) \\ &\quad < \phi(N_0, \langle p_1^i, \ldots, p_{j+1}^i, p_j^i, \ldots, p_{n_i}^i \rangle),\end{aligned} \tag{10}$$

which completes the proof. Please refer to our technical report for the details of the proof [13]. ■

B. Efficient Greedy Scheduling

Utilizing Lemma 4, we develop a polynomial time greedy algorithm for the PFS problem. The algorithm called *Greedy-Fair* is listed in Alg. 2. The algorithm sorts the received powers in an ascending order. Then, starting from the smallest received power, it assigns the nodes to the time slot that currently has the minimum sum of received powers (*i.e.*, minimum noise power for the currently being scheduled node).

Alg. 2 GreedyFair

```
$P \leftarrow$ list of received powers
Sort $P$ in ascending order
$S \leftarrow \emptyset$
$k \leftarrow$ number of time slots (i.e., frame size)
for $i \leftarrow 0$ to $length(P) - 1$ do
    Schedule $P_i$ at the time slot $t = (i \mod k)$ of $S$
end for
return $S$
```

Though we do not have a formal proof yet, we conjecture that GreedyFair computes the optimal solution of the PFS problem. The reasoning is that: (i) GreedyFair is compatible with the result of Lemma 4, and (ii) in our simulations over millions of test cases, the schedule computed by GreedyFair always matched the schedule computed with a brute force search. We leave this as a topic for future research.

V. Numerical Results

In this section, we provide numerical results to show the utility and efficiency of the proposed scheduling algorithms in various simulated network scenarios.

A. Simulation Setup

Simulations were conducted using a custom built simulator written in the Java programming language. For simulations, we consider a disk-shape network with radius 1000, where the access point is places at the center of the disk. In each simulation run, n nodes are positioned in the network following a uniform distribution over the disk area. To compute the received powers at the access point, we use a free-space path loss model in which the received power is inversely proportional to the square of the distance from the access point. Recall that our results and algorithms are insensitive to the specific propagation environment as we have assumed that the access point has knowledge about users' channel information (as is the case in 3G/4G networks). The noise power at the receiver is set to unity in every simulation run. Each point in the plots is computed by averaging over 50 simulation runs with different seeds.

We compare both the throughput and the fairness of the proposed algorithms. We use the sum throughput (Definition 3) over all time slots as the measure of throughput. Furthermore, we use the Jain's fairness index [16] as the measure of fairness for comparison. For a given set of throughputs $\{x_1, \dots, x_n\}$, the Jain's fairness index is defined as follows:

$$J(\{x_1, \dots, x_n\}) = \frac{\left(\sum_{i=1}^{n} x_i\right)^2}{n \sum_{i=1}^{n} x_i^2} .$$

B. Simulated Scheduling Algorithms

In addition to the GreedyMax and GreedyFair algorithms presented in subsections III-C and IV-B, we simulate two other scheduling algorithms for comparison purposes:

- **MaxThroughput:** This is a brute force search algorithm to find the schedule that maximizes the system throughput. Once an schedule is found, the nodes that are scheduled in the same time slot are sorted by their received powers in an ascending order. The sorting step improves the fairness of the algorithm without affecting its sum throughput (see Lemma 4).
- **Random:** This algorithm randomly assigns the nodes to the time slots such that at least one node is assigned to each time slot. Once an schedule is found, the nodes that are scheduled in the same time slot are sorted by their received powers in an ascending order.

C. Simulation Results

Since MaxThroughput takes exponential amount of time to complete, running the algorithm for more than a few number of nodes is extremely time consuming. Thus, in our initial set of results, where all four scheduling algorithms are compared, we vary the number of nodes in the network from 5 to 10. In the subsequent simulations, we do not simulate MaxThroughput, and instead vary the number of nodes from 10 to 50.

1) Initial Set of Results: Fig. 1 shows the throughput performance of different scheduling algorithms. In this experiment, frame size is set to $K = 2$ time-slots. The y-axis shows the sum throughput achieved by each algorithm as a percentage of the optimal sum throughput achieved by MaxThroughput. It is observed that the greedy algorithms achieve throughput values that are very close to that of MaxThroughput, while clearly outperforming Random. Moreover, due to the logarithmic behavior of our throughput function, the sum throughput is relatively insensitive to the scheduling algorithms when the total received power at the receiver is large. [1]

[1] **Please note that the non-uniform fluctuations seen in some plots (*e.g.*, the throughput of Random for 7 nodes in Fig. 1) are the artifact of our relative performance measures. Both the throughput percentage and fairness index are relative metrics. For all algorithms, the absolute throughput values are uniformly increasing by increasing the number of nodes.**

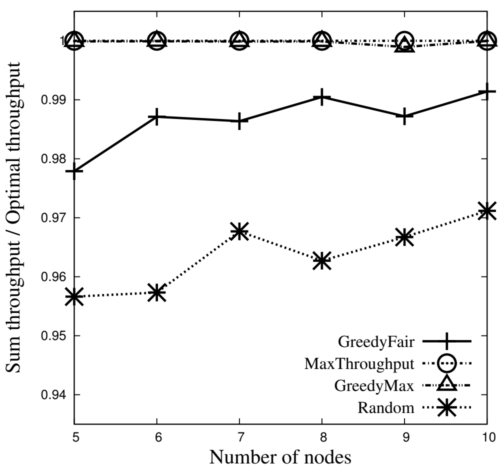

Fig. 1. Throughput performance of different scheduling algorithms relative to MaxThroughput (frame size $K = 2$ time-slots).

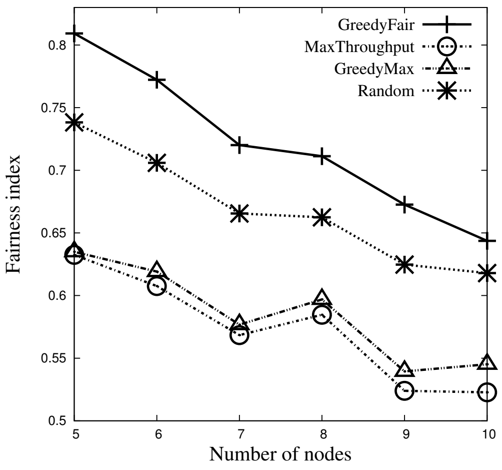

Fig. 2. Fairness index of different scheduling algorithms (frame size $K = 2$ time-slots).

Fig. 2 shows the Jain's fairness index for the four algorithms. It is observed that the GreedyFair algorithm results in the highest fairness index indicating a fair allocation of transmission rates among the users. Interestingly, GreedyMax and MaxThroughput perform quite similarly in terms of fairness; both performing worse than Random.

2) Extended Set of Results: In this subsection, we study the performance of scheduling algorithms GreedyMax, GreedyFair and Random for different network configurations in terms of the frame size and number of users. Simulation results for throughput and fairness are presented in Figs. 3 and 4, respectively. The number of nodes varies from 10 to 50, while the scheduling frame size K is set to 5 and 10 time-slots. The sum throughput is normalized by the frame size in each case resulting in a higher throughput when frame size is $K = 5$.

These results essentially confirm the behavior that was observed in Figs. 1 and 2. That is, GreedyFair and GreedyMax are both throughput efficient but GreedyFair achieves a higher fairness index as well. An interesting observation is that by

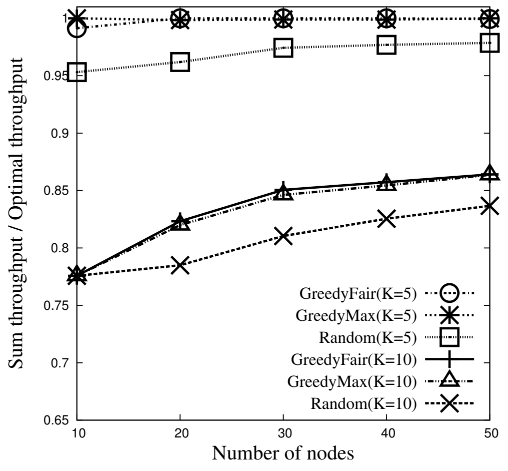

Fig. 3. Throughput performance of different scheduling algorithms relative to MaxThroughput for frame sizes $K = 5$ and $K = 10$ time-slots.

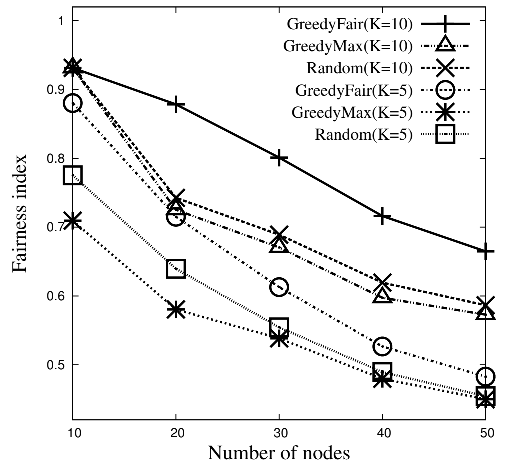

Fig. 4. Fairness index of different scheduling algorithms for frame sizes $K = 5$ and $K = 10$ time-slots.

increasing the frame size from $K = 5$ to $K = 10$, the fairness of all algorithms improves. This behavior is expected as a larger frame size distributes users more sparsely; scheduling fewer users in each time slot.

To summarize, our simulation results indicate that the GreedyFair algorithm is considerably efficient in terms of throughput, while achieving a high degree of fairness among the users. This is a polynomial time algorithm that can potentially be implemented in practice.

D. Discussion

In practice, there are several issues with SIC that prevent real systems to reach the theoretical limits. For example, we assumed that signals are decoded in an error-free manner. However, in case an error occurs while decoding a signal, it is unlikely that the signal is correctly removed from the composite signal. Thus, at later stages the signals cannot be decoded correctly [4]. Also, due to imperfect channel estimation and analog-to-digital quantization errors, the analog version of a decoded signal cannot be perfectly reconstructed. Thus, the interference removal might be imperfect, which causes lower throughput gains at later stages [17].

VI. Conclusion

In this paper, we considered the problem of uplink scheduling in wireless networks supporting SIC at the physical layer. We presented theoretical and simulation results on efficient and fair uplink scheduling. We proved that the maximum throughput scheduling problem is NP-hard. We then developed GreedyMax and GreedyFair, two greedy scheduling algorithms with polynomial time complexity, to solve the max throughput and proportional fair scheduling problem, respectively. Our simulation results indicate that GreedyMax and GreedyFair both achieve throughput that is close to the optimal, while GreedyFair is highly fair as well. For the future work, we plan to investigate our conjecture that the GreedyFair algorithm is in fact proportionally fair.

References

[1] D. Halperin, J. Ammer, T. Anderson, and D. Wetherall, "Interference cancellation: Better receivers for a new wireless MAC," in *Proc. ACM HotNets*, Atlanta, Georgia, Nov 2007.

[2] J. Blomer and N. Jindal, "Transmission capacity of wireless ad hoc networks: Successive interference cancellation vs. joint detection," in *Proc. IEEE ICC*, Dresden, Germany, Jun. 2009.

[3] J. Andrews, "Interference cancellation for cellular systems: A contemporary overview," *IEEE Wireless Commun. Mag.*, vol. 12, no. 2, Apr. 2005.

[4] D. Tse and P. Viswanath, *Fundamentals of Wireless Communication*. Cambridge University Press, 2005.

[5] A. Zubow, M. Grauel, M. Kurth, and J. Redlich, "On uplink superposition coding and multi-user diversity for wireless mesh networks," in *Proc. Mobile Ad-hoc and Sensor Networks*, China, Dec. 2009.

[6] L. Li *et al.*, "Superposition coding for wireless mesh networks," in *Proc. ACM MobiCom*, Montréal, Canada, Sep. 2007.

[7] T. Bu, L. Li, and R. Ramjee, "Generalized proportional fair scheduling in third generation wireless data networks," in *Proc. IEEE INFOCOM*, Barcelona, Spain, Apr. 2006.

[8] Q. Zhao and L. Tong, "A dynamic queue protocol for multiaccess wireless networks with multipacket reception," *IEEE Trans. Wireless Commun.*, vol. 3, no. 6, Nov. 2004.

[9] M. Ghanbarinejad, C. Schlegel, and P. Gburzynski, "Adaptive probabilistic medium access in MPR-capable ad-hoc wireless networks," in *Proc. IEEE GLOBECOM*, Honolulu, USA, Dec. 2009.

[10] S. Nagaraj, D. Truhachev, and C. Schlegel, "Analysis of a random channel access scheme with multi-packet reception," in *Proc. IEEE GLOBECOM*, New Orleans, USA, Nov. 2008.

[11] K. Kumaran and L. Qian, "Scheduling on uplink of CDMA packet data network with successive interference cancellation," in *Proc. IEEE WCNC*, New Orleans, USA, Mar. 2003.

[12] S. Deb, S. Jaiswal, and K. Nagaraj, "Real-time video multicast in wimax networks," in *Proc. IEEE INFOCOM*, Phoenix, USA, Apr. 2008.

[13] M. Mollanoori and M. Ghaderi, "Fair and efficient scheduling in wireless networks with successive interference cancellation," Dept. Computer Science, University of Calgary, Tech. Rep., Oct. 2010, http://www.ucalgary.ca/~mghaderi/docs/sic.pdf.

[14] F. Kelly, "Charging and rate control for elastic traffic," *European Trans. on Telecommunications*, vol. 8, no. 1, Jan-Feb 1997.

[15] M. Garey and D. Johnson, *Computers and intractability: A guide to the theory of NP-completeness*. W. H. Freeman & Co., 1979.

[16] R. Jain, D. Chiu, and W. Hawe, "A quantitative measure of fairness and discrimination for resource allocation in shared computer systems," *DEC Research Report TR-301*, 1984.

[17] J. Andrews and T. Meng, "Optimum power control for successive interference cancellation with imperfect channel estimation," *IEEE Trans. Wireless Commun.*, vol. 2, no. 2, Mar. 2003.