The 12th International Symposium on Wireless Personal Multimedia Communications (WPMC'09)

RECEIVER-SELECTION SCHEDULING IN WIRELESS NETWORKS

M. Bama, Andrew Thangaraj and Srikrishna Bhashyam
Deptartment of Electrical Engineering
Indian Institute of Technology Madras
Chennai, India - 600036

ABSTRACT

The problem of scheduling transmissions in a wireless network is central for optimum and efficient use of power and bandwidth. In general, optimal scheduling is a hard problem, and it is typically harder in wireless networks. In this work, we consider the problem of scheduling in wireless networks that exploit the inherent wireless broadcast advantage and employ network coding. In such a scenario, we propose a receiver-selection technique that results in scheduling with optimal throughput in test cases such as rectangular grid networks and random geometric networks. In the proposed method, the complexity of determining the schedule is significantly lesser when compared to optimal scheduling. The proposed method fares well in comparisons with other scheduling methods in the literature.

I INTRODUCTION

One of the central problems in wireless communication networks is the handling of interference. The signal sent by a transmitting node reaches the intended receiver as well as other users in the neighbourhood causing collisions. A standard method to avoid collision is to *schedule* transmissions in slots and avoid interference in every slot. A different set of transmitting nodes is chosen for each slot so that information can be transferred from the source to the sinks through intermediate nodes, if necessary. Scheduling is done in the Medium Access Control (MAC) layer of typical networks, such as those following a IEEE 802.11 standard in a centralized or distributed fashion, to avoid collision completely. Typical goals of schedulers are to minimize the number of slots, end-to-end delay or power of transmissions [2, 9, 3, 11]. Many of these minimization problems are known to be NP-hard, in general [7]. In this work, we propose a scheduling method for maximizing throughput in wireless networks. Specifically, we consider multicast problems in wireless networks that exploit the inherent *wireless broadcast advantage* and employ *network coding* at the network layer.

The omnidirectional reach of the transmitted signal in a wireless network, while resulting in interference, provides an inherent wireless broadcast advantage for multicast or broadcast requirements from one source to multiple sinks. While links of the form (transmitter,receiver) are used in wire-line networks, scheduling in wireless networks can involve *hyperarcs* of the form (transmitter, set of receivers). We consider the scheduling problem in networks where a set of non-interfering hyperarcs is activated in every time slot to exploit the wireless broadcast advantage.

Network coding (allowing intermediate nodes to combine packets before forwarding) has resulted in several simplifications in network communication problems. One such simplification is the formulation of multicast problems as max-flow problems solvable by linear programs [1]. In wireless networks, the problem of scheduling for a multicast requirement can be reduced to a problem of selecting sets of non-interfering hyperarcs under network coding. The time slots (or length of time) in which each subgraph needs to be activated for maximizing multicast throughput can be efficiently determined by a linear program such as the one in [6]. Hence, the problem of efficient scheduling to maximize the multicast throughput essentially reduces to the efficient choice of sets of non-interfering subgraphs.

In this work, we introduce a *Receiver-Selection* (RS) technique to generate hyperarcs that maximize spatial reuse of the network. In words, receiver selection is the ability of a transmitting node to determine a subset of receivers among its neighbors as intended receivers based on interference from other transmitters. Equivalently, receiver selection allows collision at nodes that are not in the subset of selected receivers. The transmitter and the chosen subset of receivers form a hyperarc that can be active in a particular time slot. We propose scheduling algorithms based on the RS technique to generate non-interfering subgraphs that result in optimal multicast throughputs in several typical test cases.

Several heuristic algorithms have been proposed for scheduling in the network-coded wireless scenario in [8, 10, 11]. The heuristics in [8] do not exploit the broadcast advantage, while the procedure described in [10] is a generic framework. Hence, we primarily use the scheduling algorithm of [11] for comparison purposes. When compared to [11], the proposed receiver selection method achieves higher and optimal throughput in the test cases at the cost of moderately higher complexity. However, the scheduling method in [11] involves linear programs and optimizations in the procedure for selecting non-interfering subgraphs, which make it a centralized method. The proposed RS methods are much simpler and ideas from the RS technique could possibly be used in distributed methods.

The rest of the paper is organized as follows: The network model and problem statement are described in Section II. The receiver-selection technique for scheduling is explained in Section III. Numerical results are presented in Section IV, and concluding remarks are provided in Section V.

II NETWORK MODEL AND PROBLEM STATEMENT

We consider wireless networks with n nodes that have uniform transmission range r_n, interference range r'_n and omnidirectional antennas. Such a network is modeled as a graph,

$G = (V, E)$, where $V = \{1, 2, \cdots, n\}$ is the set of nodes and $E = \{(i, j) : d_{ij} \leq r_n, i, j \in V\}$ is the set of edges and d_{ij} is the Euclidean distance between Node i and Node j. Let $N_i = \{j \in V \setminus i : d_{ij} \leq r_n\}$ be the set of neighbors of Node i. Let $I_i = \{j \in V : d_{ij} \leq r'_n\}$ be the set of nodes in the interference range of Node i. Each link $(i, j) \in E$ is loss-less and has a capacity of L packets per unit-time. We further assume that a node can either transmit or receive at a given time (*i.e., half duplex nodes*). We consider a single multicast session from a source $s \in V$ to a set of sinks $\mathcal{T} = \{t_1, t_2, \cdots, t_m\} \subset V$ in G.

Transmissions are scheduled in a time-slotted manner. The broadcast nature of wireless transmission is exploited by each transmitter to potentially reach multiple neighbours in the same slot. This is modeled as a hyperarc and is denoted as (i, J) where $J \subseteq N_i$. Let $\mathcal{A} = \{(i, J) : i \in V, J \subseteq N_i, J \neq \emptyset\}$ be the set of all hyperarcs of G. In each time slot, a set of transmitters is active. The set of transmitters active at the same time is chosen in such a way that the hyperarcs representing the transmissions are non-interfering. A set of non-interfering hyperarcs form a non-interfering subgraph of G. Under the protocol model for interference, two hyperarcs (i_1, J_1) and (i_2, J_2) from $\mathcal{A}$ are said to be non-interfering, if (1) $I_{i_1} \cap J_2 = \emptyset$, and (2) $I_{i_2} \cap J_1 = \emptyset$. Network coding is assumed at the network layer in all nodes for information flow.

II.A Linear program and complexity

Wireless multicast is achieved by choosing an appropriate set of non-interfering subgraphs denoted $\{A_1, A_2, \ldots, A_M\}$, and the fraction of time $(\lambda_1, \lambda_2, \ldots, \lambda_M)$ associated with each subgraph. Given the A_k $(1 \leq k \leq M)$, the optimal λ_k's can be obtained by solving a linear program from [6], which is described below.

$$\text{maximize } f \quad \text{subject to}$$

$$\sum_{k=1}^{M} \lambda_k c_k(i, J) - z_{iJ} \geq 0, \quad \forall (i, J) \in \mathcal{A}. \tag{1}$$

$$z_{iJ} - \sum_{j \in J} x_{iJj}^{(t)} \geq 0, \quad \forall (i, J) \in \mathcal{A}, t \in \mathcal{T}. \tag{2}$$

$$\sum_{\{J:(i,J)\in\mathcal{A}\}} \sum_{j \in J} x_{iJj}^{(t)} - \sum_{\{j:(j,I)\in\mathcal{A}, i\in I\}} x_{jIi}^{(t)} = \begin{cases} f \text{ if } i = s \\ -f \text{ if } i = t \\ 0 \text{ otherwise} \end{cases}, \tag{3}$$

$$\forall i \in V, t \in \mathcal{T}.$$

$$z_{iJ} \geq 0, x_{iJj}^{(t)} \geq 0, \lambda_k \geq 0, \sum_{k=1}^{M} \lambda_k \leq 1. \tag{4}$$

The linear program maximizes a flow f from the source to the sinks by operating a set of non-interfering hyperarcs in each time slot. The variables z_{iJ} (packets per unit time) denote the average rate at which packets are injected by Node i into the hyperarc (i, J) averaged over all slots. The flow variable $x_{iJj}^{(t)}$ denotes the average information flow rate from Node i to Node $j \in J$ along a hyperarc (i, J) towards sink $t \in \mathcal{T}$. The set $\mathcal{A}$ denotes the set of hyperarcs in the non-interfering subgraphs $\{A_1, A_2, \ldots, A_M\}$. The indicator function $c_k(i, J)$ is defined as

$$c_k(i, J) = \begin{cases} L & \text{if } (i, J) \in A_k, \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

The scheduling constraint (1) stipulates that rate of packets injected into each hyperarc (i, J) is less than its average capacity expressed in terms of $c_k(i, J)$. The capacity constraint (2) for each hyperarc (i, J) states that the average flows to each sink along different $j \in J$ lies within the feasible rate region for the broadcast channel from Node i to J under time-sharing of links. The flow constraint (3) ensures that the average in-flow equals the out-flow at every node for each sink.

In summary, the linear program determines the minimum of the max flows from the source to each sink under scheduling constraints in a wireless network. As shown in [1, 5, 6], every feasible solution to the linear program corresponds to a valid network coding solution achieving a throughput of f packets per unit time over M slots with each slot active for λ_k fractional time units.

The complexity of solving the linear program depends on the total number of constraints, which is equal to $n + 2|\mathcal{A}| + 1$, and the total number of variables, which is bounded by $M + |\mathcal{A}|(1 + d|\mathcal{T}|)$ where d is the maximum degree of G. Hence, the total number of hyperarcs $|\mathcal{A}|$ plays a crucial role in the complexity of the linear program. The number of hyperarcs in the network grows rapidly with the number of nodes in the network. Precisely, $|\mathcal{A}| = \sum_{i=1}^{n} (2^{d_i} - 1)$, where d_i is the degree of Node i. In dense random networks, the average degree of a node grows as $\Theta(\log n)$ with the number of nodes. In such networks $|\mathcal{A}| = \mathcal{O}(n^k)$ for a positive integer k. Therefore, the linear program for computing throughput becomes highly complex for a network with large number of nodes (typically more than 15 in dense networks).

II.B Scheduling

An important input to the linear program is the set of non-interfering subgraphs $\{A_1, A_2, \ldots, A_M\}$. Under network coding, determining the A_k's can be termed *scheduling*, since the λ_k's are determined from the A_k's through the linear program.

To obtain maximum possible throughput from the linear program, all non-interfering subgraphs of the network have to be enumerated. As suggested in [4, 6], this enumeration can be done using a conflict graph approach.

Conflict Graph: Non-interfering subgraphs can be formed as independent sets of a conflict graph [4, 6]. Each node in the conflict graph corresponds to a hyperarc from $\mathcal{A}$. Two nodes in the conflict graph are connected if the corresponding hyperarcs interfere in G. It is easy to see that every independent set in such a conflict graph will correspond to a non-interfering subgraph of G. For exact computation of capacity, all maximal independent sets of the conflict graph are enumerated to obtain all possible non-interfering subgraphs in [6]. Since the number of hyperarcs grows fast with n, this enumeration is highly complex in practice. An alternative approach proposed in [11] is described below.

Algorithm in [11]: An iterative optimization procedure proposed in [11] relies on an intelligent heuristic selection of *non-interfering* subgraphs. The procedure starts with n non-interfering graphs such that the i-th subgraph contains the hyperarc (i, N_i). The basic idea is to allow every node to be a transmitter at least once. These subgraphs are recursively updated based on the flow value of the optimization model until the cost function (*i.e.*, power consumption, congestion control) is minimized. Though this method is significantly less complex ($\mathcal{O}(n)$), it may not achieve the optimal throughput. Also, the recursive updates are dependent on the actual choice of source and sinks.

Problem statement: In this work, we are concerned with the scheduling problem of determining the set of non-interfering subgraphs $\{A_1, A_2, \ldots, A_M\}$ so that optimal throughput can be obtained at manageable complexity. The proposed scheduling algorithm uses the idea of *receiver selection*.

III Receiver-Selection Scheduling Algorithm

Receiver Selection (RS) is the ability of a node to select a subset of its neighbors as receivers to avoid interference from other transmitters. Equivalently, RS provides a receiver the option of not receiving packets from a neighbouring transmitter. We propose a scheduling algorithm based on RS to identify hyperarcs that improve the spatial reuse of the network. As shown by our numerical results, this RS-based algorithm results in a significant reduction in the number of hyperarcs and non-interfering subgraphs while providing optimal throughput.

III.A Algorithm

The receiver-selection procedure for selecting one random non-interfering subgraph of the network, denoted A_k, is presented in Algorithm 1. The nodes in a set S are considered one at a time for inclusion as transmitters in A_k. During the i-th step, receivers for s_i are *selected* by removing the neighbours of s_i in the interference range of transmitters chosen so far. The neighbours in the interference range are then removed from existing hyperarcs of A_k. At step i, the set T contains all chosen transmitters up to step $i-1$. Finally, hyperarcs of the form $(t, \emptyset)$ are removed from A_k and the set T is updated. Intuitively, this greedy method attempts to pick hyperarcs that maximize spatial reuse.

Algorithm 1 Find one non-interfering subgraph

INPUT: $G = (V, E)$, $S = \{s_1, s_2, \ldots, s_p\} \subseteq V$
Set $A_k = \emptyset$, $T = \emptyset$
for $i = 1 : p$ **do**
  Consider (s_i, N_{s_i}) for inclusion in A_k
  Receiver-Selection:
  $P = \{j \in N_{s_i} : d_{tj} \leq r'_n$ for some $t \in T\}$ (Neighbors of s_i in interference range of T)
  For each $t \in T$, replace $(t, J_t) \in A_k$ with $(t, J_t \setminus P)$
  Add $(s_i, N_{s_i} \setminus (P \cup S))$ to A_k
  Update $T = T \bigcup s_i$
  Remove hyperarcs of the form (t, ϕ) with empty receiver set from A_k and update T
end for
OUTPUT: A_k, a non-interfering subgraph.

Algorithm 1 is repeated several times with different subsets S and orderings to obtain (say) M_{RS} distinct non-interfering subgraphs $A_1, A_2, \cdots, A_{M_{RS}}$, which are maximal in the sense that $A_i \nsubseteq A_j$. The set of all hyperarcs from the subgraphs obtained by this process are collected into the set

$$\mathcal{A}_{RS} = \bigcup_{k=1}^{M_{RS}} A_k \tag{6}$$

The linear program of Section II is run to compute the multicast throughput with the hyperarc set $\mathcal{A}_{\mathcal{RS}}$ and non-interfering subgraphs $A_1, A_2, \ldots, A_{M_{RS}}$.

III.B Optimality of proposed approach

We now prove that using the subgraphs generated by the receiver-selection algorithm results in optimal throughput asymptotically for a large number of runs. We begin with a description of the non-interfering subgraphs generated by Algorithm 1. Any maximal (in the sense that no other hyperarc can be added) non-interfering subgraph A_k formed from the conflict graph (as described in Section II.B) is of the form

$$A_k = \{(i_1, J_1), (i_2, J_2), \cdots, (i_p, J_p)\}, \tag{7}$$

where $J_m \subseteq \bar{J_m} = N_{i_m} \setminus \bigcup_{l \neq m} N_{i_l}$. We define the completion of A_k to be the non-interfering subgraph

$$\bar{A_k} = \{(i_1, \bar{J_1}), (i_2, \bar{J_2}), \cdots, (i_p, \bar{J_p})\}. \tag{8}$$

We call a maximal non-interfering subgraph complete if $\bar{A_k} = A_k$.

Lemma 1. *Every maximal complete non-interfering subgraph can be generated by Algorithm 1.*

Proof. Consider a complete maximal non-interfering subgraph $\bar{A_k} = \{(i_1, \bar{J_1}), (i_2, \bar{J_2}), \cdots, (i_p, \bar{J_p})\}$, and the subset $S = \{i_1, i_2, \ldots, i_p\}$. It is clear that Algorithm 1 with the subset S will generate $\bar{A_k}$. This completes the proof. □

Let $A_1, A_2, \cdots, A_M$ be an enumeration of all maximal non-interfering subgraphs of G using the conflict graph approach (as in Section II.B). Let $A'_1, A'_2, \cdots, A'_{M'}$ be an enumeration of all maximal non-interfering subgraphs generated by running Algorithm 1 for an arbitrarily large number of times.

Theorem 1. *The linear programming problem of Section II run with the non-interfering subgraphs A'_k, $1 \leq k \leq M'$ produces the same optimal multicast throughput as when run with the set of all non-interfering subgraphs A_k, $1 \leq k \leq M$.*

Proof. Consider $\bar{A_k}$, the completion of the non-interfering subgraph A_k. By Lemma 1, there exists k' such that $\bar{A_k} = A'_{k'} = \{(i, \bar{J}), (i_1, \bar{J_1}), \cdots\}$. Hence, time sharing between the set of subgraphs in $\{A_k : 1 \leq k \leq M\}$ is equivalent to time sharing between the set of subgraphs in $\{A'_{k'} : \bar{A_k} = A'_{k'}, 1 \leq k \leq M\}$. This finishes the proof. □

IV Numerical Results

In this section, we evaluate the proposed receiver-selection scheduling algorithm by comparing throughput performance and complexity with the heuristic scheduling algorithm described in [11] and optimal scheduling. Under optimal scheduling, all maximal independent sets of the conflict graph of Section II.B are chosen as non-interfering subgraphs. The multicast throughput is determined using the linear program in Section II for the set of non-interfering graphs and the set of hyperarcs generated by each scheduling algorithm. We analyse the performance of these algorithms for (i) rectangular grid networks and (ii) random networks.

IV.A Rectangular grid networks

We consider $n_r \times 3$ rectangular grid networks, where the j-th node of the i-th row is placed at the coordinates (i, j) for $i = 1, 2, \ldots, n_r$ and $j = 1, 2, 3$. The nodes are identical with a transmission range, $r_n = r'_n = \sqrt{2}$. The link capacities are taken to be one packet per unit time. The number of rows n_r is varied from 4 to 10. The center node in the first row is considered as the source node. The three nodes in the n_r-th row are taken to be the sinks. For $n_r = 4$, the grid is shown in Fig. 1.

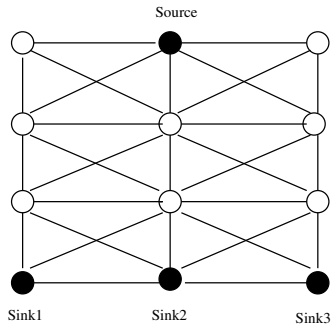

Figure 1: An example - 4 x 3 grid network.

A comparison of multicast throughput for different scheduling algorithms is shown in Fig. 2. The optimal throughput is known to be 2/3 for the above choice of source and sinks [6]. From Fig. 2, we observe that the proposed algorithm achieves optimal throughput, while the algorithm in [11] achieves only half the optimal throughput.

For comparing complexity, we plot the number of hyperarcs used in the three algorithms in Figs. 3. We notice that the receiver-selection algorithm produces a significantly lesser number of hyperarcs, while achieving optimal throughput. The algorithm of [11] is the least complex of the three algorithms. However, as observed earlier, this algorithm fails to achieve the optimal throughput by a significant margin.

The number of runs of Algorithm 1 needed to achieve optimal throughput is plotted in Fig. 4. For comparison, the total number of non-interfering subgraphs needed for optimal scheduling is also shown. We see that receiver selection succeeds in finding the non-interfering subgraphs crucial for high throughput in much fewer runs than the total number of possibilities.

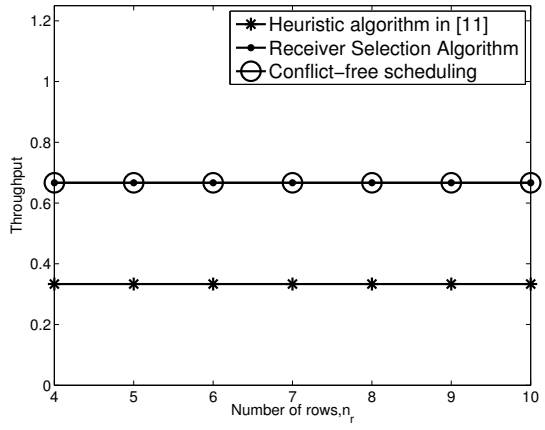

Figure 2: Multicast throughput of $n_r \times 3$ grid networks

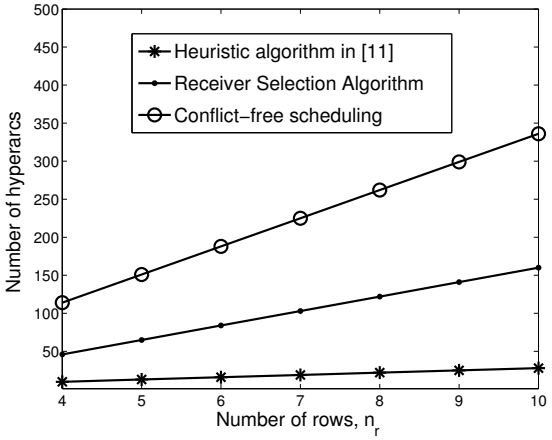

Figure 3: Average number of hyperarcs in $n_r \times 3$ grid networks

IV.B Random networks

We consider an unit square region where n nodes with a transmission range $r_n = r'_n = \sqrt{\frac{2 \log n}{\pi n}}$ are placed uniformly at random and independent of each other. Node 1 is taken as the source node. We consider two other nodes, randomly chosen among the remaining $n - 1$ nodes, as sinks for multicast. The multicast throughput is averaged over 500 different network realizations and over all possible choices of sink for each realization.

The comparison of multicast throughput is shown in Fig. 5 when the link capacities are one unit. We see that the throughput is higher with receiver selection throughout with the difference increasing for larger n. Because of the computation complexity, we could run the optimal scheduling algorithm only for 10-node networks. For 10-node networks, we observed that RS scheduling achieved optimal throughput. Algorithm 1 was run 10000 times with $p = n$ to generate non-interfering subgraphs.

The average number of hyperarcs used for different algorithms is shown in Fig. 6. The proposed RS scheduling algo-

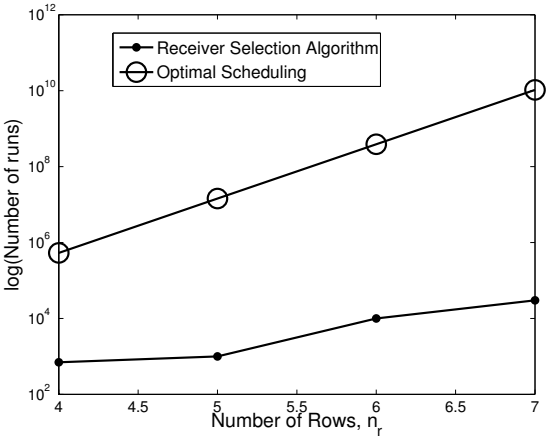

Figure 4: Number of runs required to achieve optimal throughput.

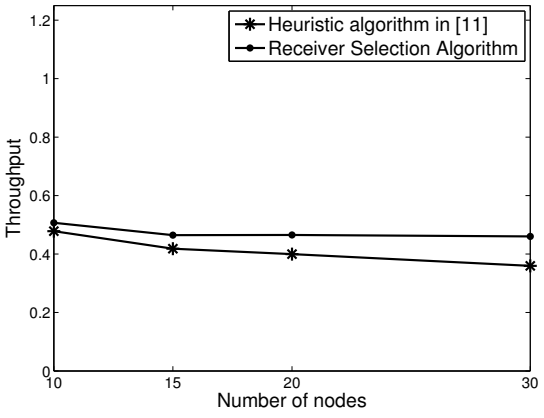

Figure 5: Multicast throughput for random networks

rithm generates a moderate number of hyperarcs and achieves the optimal throughput. The reduction is close to an order of magnitude particularly noticeable for larger number of nodes. The number of hyperarcs generated by the algorithm in [11] is equal to the number of nodes in the network, but the throughput is lesser than RS scheduling.

In summary, we see that receiver selection scheduling for hyperarcs results in optimal throughput in rectangular grid networks and random networks at moderate complexity levels.

V Conclusion

In this paper, we introduced the receiver-selection technique for generating hyperarcs that maximize spatial reuse. The proposed scheduling algorithms based on the receiver selection technique offer optimal throughput under network coding at significantly lesser complexity. Numerical results with random networks and rectangular grid networks demonstrate the effectiveness of the proposed approach.

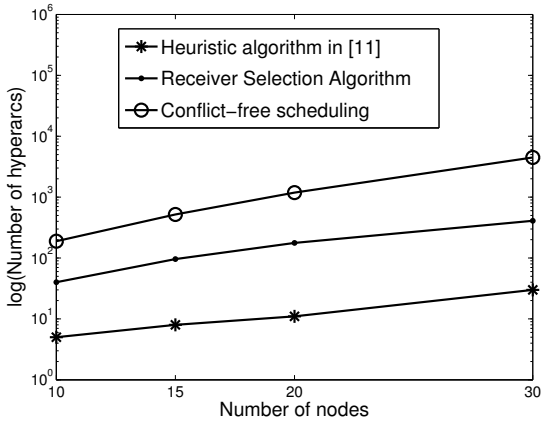

Figure 6: Average number of hyperarcs for random networks

References

[1] R. Ahlswede, N. Cai, S. R. Li, and R. W. Yeung. Network information flow. *IEEE Transactions on Information Theory*, 4(4):1204–1216, July 2000.

[2] A. Behzad and I. Rubin. Optimum integrated link scheduling and power control for multihop wireless networks. *Vehicular Technology, IEEE Transactions on*, 56(1):194–205, Jan. 2007.

[3] R.L. Cruz and A.V. Santhanam. Optimal routing, link scheduling and power control in multihop wireless networks. In *Proceedings INFOCOM 2003*, volume 1, pages 702–711 vol.1, March-3 April 2003.

[4] K. Jain, J. Padhye, V. Padmanabhan, and L. Qiu. Impact of interference on multi-hop wireless network performance. In *Proc. of Mobicom*, 2003.

[5] D. S. Lun, N. Ratnakar, M. Medard, R. Koetter, D. R. Karger, T. Ho, E. Ahmed, and F. Zhao. Minimum-cost multicast over coded packet networks. *IEEE trans. Inform. Theory*, 52(6):2608–2623, June 2006.

[6] J. S. Park, D. S. Lun, F. Soldo, M. Gerla, and M. Medard. Performance of network coding in ad hoc networks. In *Proc. of IEEE MILCOM 2006*, October 2006.

[7] R. Ramanathan. A unified framework and algorithm for channel assignment in wireless networks. *Wireless Networks*, 5:81–94, 1999.

[8] Y. E. Sagduyu and A. Ephremides. On joint MAC and network coding in wireless ad hoc networks. *IEEE trans. Inform. Theory*, 53(10):3697–3713, October 2007.

[9] Jian Tang, Guoliang Xue, C. Chandler, and Weiyi Zhang. Link scheduling with power control for throughput enhancement in multihop wireless networks. *Vehicular Technology, IEEE Transactions on*, 55(3):733–742, May 2006.

[10] L. Wan, J. Luo, and A. Ephermides. An iterative framework for optimizing multicast throughput in wireless networks. In *Proc. of IEEE symposium on Information theory*, pages 196–200, July 2008.

[11] Y. Wu, P.A. Chou, Q. Zhang, K. Jain, W. Zhu, and S. Y. Kung. Network planning in wireless ad-hoc networks: A cross-layer approach. *IEEE Journal on selected areas in communications*, 23(1):136–150, January 2005.