

Maximal Throughput Scheduling Based on Physical Interference Model Using Learning Automata

Ziaeddin Beheshtifard,
Department of Computer Engineering and Information Technology, Islamic Azad University, Qazvin Branch,
Qazvin, Iran
Email: beheshti@qiau.ac.ir

Mohammad Reza Meybodi
Department of Computer Engineering, Amirkabir University of Technology, Tehran, Iran
Email: mmeybodi@aut.ac.ir

Abstract Wireless link scheduling is one of the major challenging issues in multi hop wireless networks when they need to be designed in distributed fashion. In this work we improve the general randomized scheduling method by using learning automata based framework that allows throughput optimal scheduling algorithms could be developed in distributed fashion. We propose a distributed scheduling algorithm that operates on the more realistic conflict graph based on physical interference model. It uses a combination of a distributed learning automata based pick algorithm and an algorithm that compares successive scheduling solutions. The comparison is done by creating spanning tree on the conflict graph of the two consecutive schedules. Briefly we propose a distributed scheduling schemes that (i) is throughput optimal (ii) intelligently choose links for new schedule (iii) message and time complexity is in $O(n^3)$.

Keywords Wireless Network, Maximal Scheduling, Physical Interference, Learning Automata

1 Introduction

MAC level wireless link scheduling has been one of the most challenging problems in multi-hop wireless networks over the last decades. Scheduling in wireless networks involves allocating network resources among competing network users in uncertain environments that generated by fully stochastic network dynamics. Also scheduling is a main part of joint scheduling, routing and congestion control policies in network utility maximization frameworks and efficient algorithms are presented in this section, will pave the way for more effective methods of reaching the optimal solution for joint resource allocation problems.

Basic studies on scheduling, is mainly developed in seminal work by Tassiulas and Ephremides in constraint queuing systems [1]. They studied the throughput of scheduling policy, which is characterized by stability region C_π , that is, the set of all vectors of arrival rates for which the system is stable. They characterized optimal policy that its stability region dominates all other stability regions. Although their method reaches all achievable throughput regions, the nature of the proposed method is based on centralized computations that may have some issue in realistic applications. They proposed a low complexity randomized algorithm that greedily reach optimal throughput according to simple pick and compare algorithms, but still execute on centralized fashion[2]. The approach presented in [2] works as follows. In each time slot, a candidate solution to the maximum weighted matching problem is obtained randomly. If the sum of the weights of selected links is higher than the value of the current solution, they are replaced. Using this approach guarantees achieving 100% throughput under certain conditions on the way in which the matching is obtained.

Impact of imperfect scheduling on cross layer rate control studied by Lin and Shroff[3]. They showed that using a maximal weight matching algorithm in distributed fashion distributed along with congestion control may achieve up to 50% of achievable throughput region. They designed fully distributed cross layer congestion control and scheduling algorithm for the node-exclusive interference model.

Recent studies by Eryilmaz et al. that inspired by pick and compare algorithm developed scheduling problem in two hop interference model[4]. They proposed a general framework that allows some distributed algorithms with polynomial communication and computation complexity to be developed in two-hop interference model in wireless networks. Another work presented based on physical interference model that use a randomized algorithm for link

activation based on transmission successfulness and two level priority consideration. They proved their work in the multi-hop wireless networks context, guarantees full throughput efficiency in a distributed manner[5].

Recently, there have been several efforts towards analysis and design of wireless multi-hop networks under more general interference models than the graph-based interference model. Authors of [6] develop mixed-integer linear programming formulation for minimizing the schedule length over a TDMA wireless multi-hop networks, based on the joint MAC scheduling and power control under the physical interference model. Our work is different from proposed distributed algorithm in [6] in that the mentioned work is sub-optimal, and is based on a relaxed physical interference model, where it only considers the impact of single transmitter on interference. The similar work in [7] also uses this relaxed physical interference model to study the performance of the scheduling mechanism of the MAC protocol. The physical model used in this paper considered only pairwise interference and also neglected noise.

Some recent studies results [8],[9] and [10] have addressed some challenges related to scheduling problem under physical interference model, but their solutions only guarantee sub-optimal approximation of the maximum achievable throughput. The later work, develops interference-aware scheduling protocols under arbitrary physical interference model such that consider different transmission power control settings, i.e., uniform power control and monotone power control.

In this paper we develop an adaptive learning automata based distributed randomized scheduling algorithm that can fully utilize the throughput region of a multi-hop wireless network under realistic SINR model. In section 2 we present the system model from three different perspectives, including interference model, network model and stochastic automata game model of the network. In section 3 we propose collection of distributed algorithms based on randomized pick and compare method that practically could be developed in more realistic wireless network. Section 4 provides some simulation results and finally we summarize the results of the work in section 5.

2 Models

2.1 Interference Model

According to intrinsic difficulty in modeling of wireless network physical layer, in contrast with wired link networks, wireless links capacity are dependent of one another, so it suffer from mutual interference required to model the wireless channel interference accurately. Several studies within the literatures have considered simplified graph based interference model. In a comparative study presented in [11], the authors investigate two such model, the interference range model which uses fixed distance ranges, and the physical SINR model which specifies ability of packet reception in receiver by comparing the desired sender signal strength with cumulative interference generated by other senders. From their results, it can be concluded that different physical layer model will lead to different result. Specifically the interference range model misleadingly predict the throughput as function of the transmit power. More precisely for an N node grid mesh network, the interference range model predicts a $\Theta(1/N)$ trend for the throughput. In contrast, under the SINR additive interference model, the capacity achieved for an N node grid mesh network is actually $O(1/N^{3/2})$ [11]. These conflicting results reveal the importance of choosing more realistic physical layer model.

In terms of wireless resource allocation, interference model is important significantly in wireless link scheduling algorithms. Many studies consider simple models of interference such that main issues of their classes of algorithms are faced with conflicting pairs of links constraints. In this kind of constraints certain pairs of wireless links, are specified that no two links that constitute a conflicting pairs can be activated simultaneously. Scheduled links set is any set of wireless links that does not include conflicting pairs of links. So the solution of wireless link scheduling problem is equivalent to the computation of maximum weight matching of graph, whereas it become NP-Hard under more general interference model in wireless networks[12]. The numerous variety of solutions have significant result gained for the problem, but most of them easily use simple binary interference model, e.g., hop-based, range-based or protocol interference model[12]. Under these types of interference models conflicting links predetermined by conflict graph, but in realistic wireless networks interference on links determined by global additive noise rather than distance metrics of transmission range.

The interference model can be formulated as follow. Consider a network of n wireless nodes, n_1, n_2, \dots, n_N . A message from a transmitter n_s can be successfully decoded by a receiver n_r if and only if $\frac{P_r}{I_r + N} \geq \beta$ for a hardware

dependent ratio β . In this equation, P_r is the signal strength of the message at the receiver n_r , I_r is the sum of all interferences at n_r and N is the ambient noise. In the physical model of signal propagation, the signal strength P_r is modeled as a decreasing function depending of the distance between n_s , n_r more precisely $P_r = \frac{1}{d(n_s+n_r)^\alpha}$ where α is called the path-loss exponent, a constant dependent on the medium, typically between 2 and 6 [13].

Some of the factors that affect on rate of wireless links are application protocol level parameters like bit error rate(BER) and coding error, so information are transmitted from the upper layer at rate $(1-BER)\theta_l R_l$ where $0 < \theta_l \leq 1$ is coding error and R_l is rate of wireless link l [14]. Physical level factors are the second type of factors that realistic rate of wireless links are depend. These factors constitute a probabilistic model for channel state of physical wireless links[15]. The channel states are modeled by gain matrix $G = [G_{ij}]_{L \times L}$ where G_{ij} is the power gain from the transmitter on link j to the receiver on link i . The vector of transmitter powers is given by $S = [S_l]_L$. The link rate function is assumed to be of the form[15].

$$R_l(S, G) = \log \left(1 + \frac{\phi K G_{ll} S_l}{\sum_{j \neq l} G_{lj} S_j + N} \right) \quad (1)$$

Where $K = -1.5/\log(5BER)$, is a scaling parameter for the received power, ϕ is the coding gain associated with a choice of convolutional code and N is ambient noise. The probability distribution of G is unknown to the network. Because the channel is randomly varying, the link rates can also vary, resulting in congestion and queuing delay at the link buffers.

2.2 Network Model

Consider a wireless mesh network with $N = \{1, 2, \dots, n\}$ is the set of nodes and $L = \{(i, j) : i, j \in N\}$ is the set of directed links in mesh routers backbone. Due to environmental limitations and also different power levels, it is even possible that two nodes i and j belonging to N are linked only one-way, i.e., (i, j) belongs to L but (j, i) not. Each link corresponds to a pair of transmitter node and receiver. We assume that all radio devices use half-duplex transmission where each radio has only one transceiver. Let $b(l)$ and $e(l)$ denote the transmitter node and the receiver node, respectively, of link l and $\mathcal{V}(i)$ is a set of nodes transmitting concurrently with node i . We assume that time is divided into slots of unit length, and synchronized between nodes, denoted by t , and the duration is adequately chosen so one fixed size packet to be transmittable in a unit time slot.

Let $q_i^d(t)$ represent the current amount of packets in node i that should be served to destination d , also called the queue backlog. Every timeslot, nodes can transmit data to others so $q_d^d(t) = 0$. Data that is transmitted from one node to another node is removed from the queue of the first node and added to the queue of the second. Data that is transmitted to its destination is removed from the network. $A_i^d(t)$ is defined as the amount of data that arrives to node i on slot t that must eventually be delivered to destination node d .

Definition 1: A schedule $\vec{S}(t) = (S_l(t) \in \{0, 1\} : l = 1, 2, \dots, |L|)$, $S_l(t)$, is an indicator variable representing the set of link scheduled for transmission intended at time slot t where $S_l(t) = 1$ if link l is scheduled at timeslot t , and 0 otherwise. We define collection of all scheduling vector as Π .

Because of unexpected time varying network dynamics that commonly shares resources within neighbors, successful transmission between two adjacent nodes, depends on physical conditions in the network such as channel frequency interference and error probability conditions that we can model it by SINR parameters.

A scheduling algorithm, in this paper, selects $\vec{S}(t)$ based on the queue-sizes, $\vec{Q}(t)$, $t \leq t$ thus the dynamics of the system can be described as evolution of queues size may be written as:

$$Q_i(t+1) = [Q_i(t) - S_i(t)]^+ + A_i(t) \quad (2)$$

Definition 2: A scheduling algorithm is called Stable if for any feasible rate; the average queue-size is bounded as following:

$$\limsup_{t \rightarrow \infty} (E[Q_i(t)]) < \infty$$

Definition 3: Throughput region Λ represent the set of all feasible transmission scheduling vectors that available under physical state $p(t)$.

$$\Lambda = \{\bar{\delta} : \bar{\delta} = \sum_{S_i \in \bar{S}} \alpha_i S_i, \quad 0 \leq \alpha_i \leq 1, \quad \sum_{i=1}^{|\bar{S}|} \alpha_i = 1\} \quad (3)$$

Node scheduling controller in every slot t , observes $p(t)$ and choose transmission scheduling $S_l(t)$ within the set $\Lambda_{p(t)}$. So $x_l(t)$ directly depends to physical state of network that affects on transmission rate of link l .

2.3 Stochastic Automata Game

Here, we present an overview of the concept driving automata games [16]. Abstractly, learning automata are adaptive decision making unit situated in a probabilistic environment. LA's widely used as a useful artificial intelligent tools in multi-agent reinforcement learning algorithms. A learning automaton learns the optimal action through repeatedly choosing of actions from a finite number of allowable actions. For every action that it chooses, the random environment in which it operates evaluates that action. A corresponding stochastic feedback is sent to the automaton based on which the next action is chosen. As this process progresses the automaton learns to choose the optimal action for that unknown probability distribution of environment asymptotically. An important property of the learning automaton is its ability to improve its performance with time while operation in an unknown environment. In the context of wireless networks, learning automata has been used as an intelligent optimization tool that can model stochastic aspects of network dynamics and unknown probability distribution in some challenging problems such as dynamic channel assignment[17], congestion control[18], routing[19] and power control[20].

The operation of a LA is based on the probability updating algorithm, also known as the reinforcement scheme. This algorithm uses the environmental response that was received as a result of performing the action selected at cycle n (action $a(n)$), in order to update the probability distribution vector p . After the updating is performed, the LA selects the action to perform at cycle $n+1$, according to the updated probability distribution vector $p(n+1)$. The Learning algorithm of the automata can be obtained from ordinary differential equation (ODE) that well approximates the behavior of the algorithm. The stochastic algorithm can be state by following update equation:

$$p_{k+1} = p_k + bG(p_k, \xi_k) \quad (4)$$

Where $p_k \in \mathfrak{R}^N$, is called the state vector and $\xi_k \in \mathfrak{R}^N$, is the noise vector and b is step size or learning parameter. A general reinforcement scheme has the form of Equations (5) and (6):

$$p_i(n+1) = p_i(n) - (1 - \beta(n))g_i(p(n)) - \beta(n)h_i(p(n)) \quad \text{if } \alpha(n) \neq \alpha_i \quad (5)$$

$$p_i(n+1) = p_i(n) - (1 - \beta(n)) \sum_{j \neq i} g_j(p(n)) - \beta(n) \sum_{j \neq i} h_j(p(n)) \quad \text{if } \alpha(n) = \alpha_i \quad (6)$$

The functions g_i and h_i are associated with reward and penalty for the selected action a_i , respectively, while $\beta(n)$ is a parameter expressing the received environmental response at cycle n , normalized in the interval $[0,1]$. The lower the value of $\beta(n)$, the more favorable the response is. According to functions g and h , different schemes of reinforcement can be used. The simplest and most commonly used of them is reward-inaction (L_{R-I}). The L_{R-I} algorithm updates the action probabilities as described below:

$$p_i(k+1) = p_i(k) + \lambda\beta(k)(1 - p_i(k))$$

$$p_j(k+1) = p_j(k) - \lambda\beta(k)p_j(k) \quad \forall j \neq i$$

The above algorithm can be rewritten in vector notation as:

$$P(k+1) = P(k) + \lambda\beta(k)(e_i - P(k)) \quad (7)$$

Here e_i is the unit vector with i^{th} component unity where the index i correspond to the action selected at k and λ is the learning (or step-size) parameter satisfying $0 < \lambda < 1$. Compared to L_{R-P} , the main difference here is that, even the reinforcement is unfavorable (that is, $\beta(k) = 0$) the action probabilities are not updated.

3 Algorithms Design

In this section we establish the scheduling algorithm for obtaining maximum throughput in a distributed fashion that can be applied in large class of physical interference model. Our algorithm is motivated by low complexity algorithm, known as pick and compare approach that introduced by Tassiulas at [2] that presented here as GSA in algorithm 1. This is based on the idea that find max-weight schedule not necessary at each time slot. They showed that for stability guarantee, it is enough to find a good one with probabilistic guarantee of finding an optimal schedule.

Algorithm 1: GENERIHC SCHEDULING ALGORITHM (GSA)

- a. $t = 0$
 - b. Do
 1. $\vec{S}(t) = \text{pick}()$
 2. $[\vec{S}(t+1), \varepsilon] = \text{compare}(\vec{S}(t), \vec{\hat{S}}(t))$
 3. $t = t+1$
 - c. while($\varepsilon < \text{threshold}$)
-

We state the following theorem according to the general property of mentioned algorithm.

Theorem 1. *Let the algorithm GSA satisfy the following property:*

P1. *For any time t define $S(t) = (S_i(t), i \in \{1..N\})$ represent link scheduling binary vector and $S^*(t)$ represent maximum weight link scheduling vector. Let $S(t)$ be independent random variable, there exist a $\delta > 0$ such that $\Pr(S(t) = S^*) > \delta$, for some $\delta > 0$, for all t*

Then the sequence of $\{S(t)\}_{t=0}^{\infty}$ converge to S^ with probability of 1.*

Proof 1. *According to GSA it is sufficient to show that the algorithm visit the maximal schedule with probability of 1 in a limited time, so because of i.i.d property of sequence $\{S(t)\}_{t=0}^{\infty}$ we have:*

$$\Pr(S(t) = S^*, t < \infty) = 1 - \Pr(S(t) \neq S^*, t \in [0, \infty)) = 1 - \lim_{t \rightarrow \infty} (1 - \delta)^t = 1$$

Theorem 2: *For any rate vector $\vec{\lambda}$ under rate region Λ , process $\{Q(t)\}_{t=0}^{\infty}$ weakly converge to random variable \tilde{Q} such that $E[\tilde{Q}] < \infty$.*

Proof: The detailed proof is given in [2].

In section 1 above we describe the characteristics of physical interference model and general randomized scheduling algorithm that satisfy important stability property of queuing network. In this section we establish by using of those theoretical results to design an intelligent low-complexity distributed for real world physical interference model.

Following the general scheduling mechanism, on each time slot, nodes uses learning automata based game theoretic approach to exploit contention pattern on its neighborhoods. RTS/CTS following by DATA/ACK establish signaling mechanism to resolve contention, which probabilistically lead to successful transmission.

Definition 4: A link $l = (i, j)$ $i, j \in N$ is said to be potentially schedulable, if SINR property of physical interference model satisfies RTS from I to j and CTS from j to i .

Definition 5: A transmission on link $l = (i, j)$ $i, j \in N$ is said to be successful, if SINR property of physical interference model satisfies DATA from I to j and ACK from j to i .

Our objective in this section is to develop our two generic *Pick* and *Compare* algorithm that consecutively operate in repeated time slots. In the pick step, a stochastic learning automata based algorithm attains a feasible schedule in distribute fashion. The algorithm gradually reaches to optimal feasible schedule with a positive probability for property P.1 to retain. In the compare step, the total weight of scheduled queues is calculated in distributed manner. The result of this step used to decision about whether an old schedule should be hold or not.

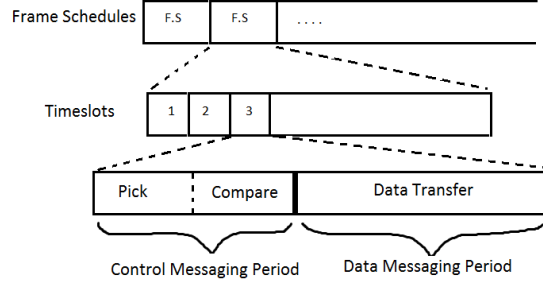


Figure 1 Timeslot configuration for data and control messaging

Pick and *Compare* algorithms operate consecutively in each time slot. Link scheduling for data transmission is updated at beginning of each time slot. According to same shared wireless medium and to prevent data and control message collision, the time slot is divided into two separated intervals, we called *Control Messaging Period* and *Data Messaging period* (

Figure 1 Timeslot configuration for data and control messaging. Both pick and compare algorithms required to operate with exact coincidence in time and so node should be synchronized in time. Time synchronization can be relaxed by using a buffer for accommodate propagation delay.

2.4 Pick Algorithm

This section describe Pick algorithm that gradually learn contention patterns in neighborhood of each node. Every node in this network uses a controller that equipped by two independent learning automata. First automaton support decision for nodes to start an activation for link scheduling, and the second one support decision for neighbor selection if the first decision was positive for participation in link activation. Figure 2 represent logical orders of steps required for activation of a link. After pick algorithm, two possible condition could be occurred. First condition is when node withdraw participation in link activation (i.e. endpoint W) in which case the automaton should be penalized. In the second case the automaton's operation is postponed to the of next compare step.

The algorithm ensures that if two end nodes of a link try to transmit, they sense each other during RTS transmission and terminate attempting to link activation. Another event is that more than one neighbors of node are attempting to establish a connection. In this case receiving node does not get sending node's message due to collision, so it does not respond via CTS message and finally terminates the communication. Respectively Sending node does not receive CTS message and terminate subsequently. In this case it results that both side of link withdraw link activation. Finally if receiving node senses another transmission during CTS sending phase, it withdraw. This ensures that two interfering node does not become end point of two different links. Thus, all of the events that lead to physical interferences, resolved in pick algorithm and the final allocation must be feasible.

The aim of this section is to analyze the stability of learning method in the game of multi-automata. A play $a(t) = (a_1(t), \dots, a_N(t))$ of n automata is a set of strategies chosen by the automata at stage t , such that $a^i(t)$ is the action selected by i th automata. Also $\beta(t) = (\beta_1(t), \dots, \beta_N(t))$ is the payoff vector such that $\beta^i(t)$ is the payoff to i^{th} automata. Let the action set of i^{th} automata denoted by A_i with $|A_i| = r_i$ such that r_i is the number of neighbors of node i . Define function $F^i: \{a_1, \dots, a_N\} \rightarrow [0,1], 1 \leq i \leq N$

$$F^i(a) = E[\beta_i | i^{th} \text{ automata chose } a_i \in A]$$

F^i is the payoff function for player i . The players only receive the payoff such that they have no knowledge of these functions. We say $a^* = (a_1^*, \dots, a_N^*)$ is an optimal point of the game if for each $i, 1 \leq i \leq N$,

$$F^i(a^*) \geq F^i(a) \text{ for all } a = (a_1^*, \dots, a_{i-1}^*, a_i, a_{i+1}^*, \dots, a_N^*), a_i \neq a_i^*$$

Remark 1: In the above definition, the condition implies that a^* is a Nash equilibrium of the game matrix $F(.)$ indexed by $a_i, 1 \leq i \leq N$.

Two known theorems that we used for the stability of systems are stated as follow:

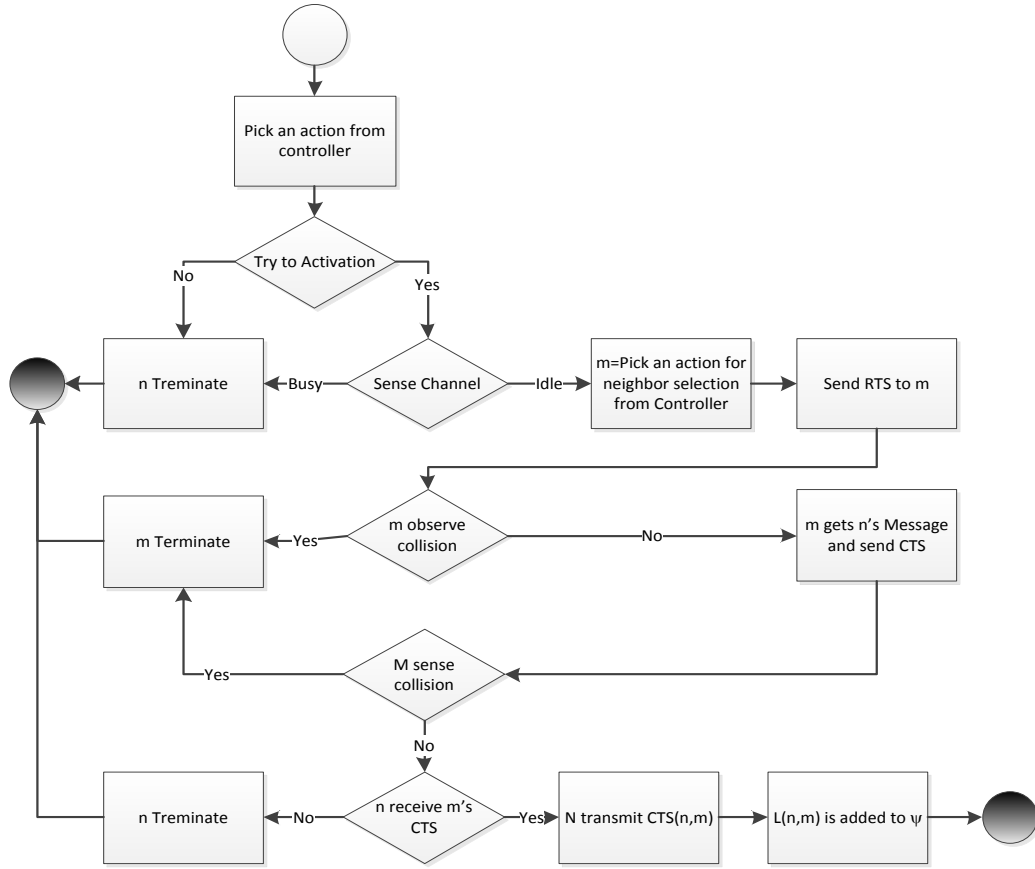


Figure 2 Pick algorithm steps for acquiring feasible schedule

Theorem 3. Consider the discrete-time system $X(t + 1) = f(X(t))$ Where X is a vector, f is a vector such that $f(0)=0$. Suppose there exist scalar function $V(x)$ continuous in x such that:

- (1) $V(X) > 0$ for $X \neq 0$.
- (2) $\Delta V(X) < 0$ for $X \neq 0$
- (3) $V(0) = 0$
- (4) $V(X) \rightarrow \infty$ as $\|X\| \rightarrow \infty$.

Then equilibrium state $X=0$ is asymptotically stable and $V(X)$ is a Lyapunov function.

Theorem 4. if the function $f(X)$ defined as: $\|f(X)\|_1 < \|X\|_1$ with $f(0) = 0$, for some set of $X \neq 0$ the system is asymptotically stable and one of its Lyapunov function is : $V(X) = \|X\|_1$

These two theorems and their proofs are given in [21].

Theorem 5. A multi-player game of automata using the proposed learning scheme in stationary environment reach the pure optimal strategy.

An Identical analysis for single automaton has been proposed in [22]. In this study easily shown that the 1st norm of expected value of all sub-optimal action converges to zero. For example for i th automaton if a_i is optimal action and $\bar{P}_i(t + 1)$ is a vector of expected value for other sub optimal action in steady state, we can show that:

$$\|\bar{P}(t + 1)\|_1 < \|\bar{P}(t)\|_1$$

We know that every automaton in the network choose an action independently base on its action probability vector, so we can extend this analysis to matrix of all probability vectors. So if we define $f(\bar{P}(t)) = \bar{P}(t + 1)$ and extending $P(t)$ to matrix of all probability vectors of every automata, we have proven that, the expected values of the

probabilities of the sub-optimal actions all converge to zero. This implies that the probability of the optimal action converges to 1, and according to Lyapunov function the stability of the optimal point is proven [23].

2.5 Compare Algorithm

The second part of our general scheduling algorithm, compares new schedule $\hat{S}(t)$ and old one $S(t)$ to reach new schedule $S(t + 1)$. At the end of every stage, the compare algorithm should compute improvement sign of current schedule and select new schedule as:

$$improvement = sign\left(\sum_{l \in \hat{S}(t)} Q_l(t) - \sum_{l' \in S(t)} Q_{l'}(t)\right) \quad (8)$$

If $improvement = 1$ then $S(t + 1) = \hat{S}(t)$ else $S(t + 1) = S(t)$

For computing(8), algorithm have to traverse two graph associated with $\hat{S}(t)$ and $S(t)$. We use the conflict graph associated with these two schedules which capture the data about interfering links. Some studies have shown that the use of inappropriate conflict graph based interference models, could lead to large performance degradation in wireless networks[24]. In its current design, the model fails to capture the cumulative effect of interference. Simultaneous activation of multiple links can occasion enough cumulated interference to disrupt a transmission even though none of these links alone is harmful for the transmission.

In the physical interference model, conflicts are not represented as binary which simultaneous transmission occur in two interfering links. Suppose node x wants to transmit to node y . The signal power gain of x 's transmission is calculated as received at y . The transmission is successful if $SNR_{xy} > SNR_{thresh}$, where SNR_{xy} is the signal to noise ratio at y of the transmission received from x . The total noise at y is the total of the ambient noise (N_{amb}) and the interference due to the power gain of other transmitters in the network. Based on this model, a link l_{xy} exists between x and y in the connectivity graph if and only if $G_{xy}/N_{amb} > SNR_{thresh}$ (i.e. SNR exceeds the minimum threshold at least in the presence of ambient noise only). Because conflicts are not binary, interference in the physical model gradually increases as more neighboring nodes transmit and becomes unacceptable when the noise level reaches a threshold. According to this observation, in [25] presented physical interference based conflict graph such that modeled by the weighted conflict graph, where the weight of a directed edge between two vertices indicates the fraction of the permissible noise at the receiving node. So the weight of a directed edge from vertices l_{zt} to vertices l_{xy} represented by $w_{zt \rightarrow xy}$, indicate what fraction of the permissible noise at node y from link l_{xy} to still be operable, in contributed by activity on link l_{zt} . In other form:

$$w_{zt \rightarrow xy} = \frac{G_{zy}}{G_{xy} / SNR_{thresh} - N_{amb}} \quad (9)$$

Where G_{zy} , G_{xy} are signal power gain of two transmitter z , x on receiving node y and the denominator is the maximum permissible interference noise at node y that allow successful reception from node x 's transmission.

Definition 6: Suppose $G = (N, L)$ is the network graph such that $N = \{n_1, n_2, \dots, n_{|N|}\}$, $L = \{l_{n_i, n_j} : n_i, n_j \in N\}$ the $G^c = (N^c, L^c)$ is the conflict graph for graph G such that: $N^c = L$, $L^c = \{(l_{zt}, l_{xy}) : l_{zt}, l_{xy} \in L, w_{zt \rightarrow xy} \geq 1\}$.

To demonstrate the definition of physical interference based conflict graph we consider the grid network depicted in

Figure 3. According to our model, link l_x will be connected to link l_y as a directed edge, if based on (9) $w_{l_x \rightarrow l_y} \geq 1$. A sub-conflict graph for 4x4 grid network that shows interfering links related to 112 shown in the Figure 3 (b) based on defined sub-graph of total conflict graph. Notice that the interference which extracted from the conflict graph represent all physical situations that results interference in receiving nodes. This observation will not restrictive, because the pick algorithm ensure that link allocation for current scheduling is feasible so no interference

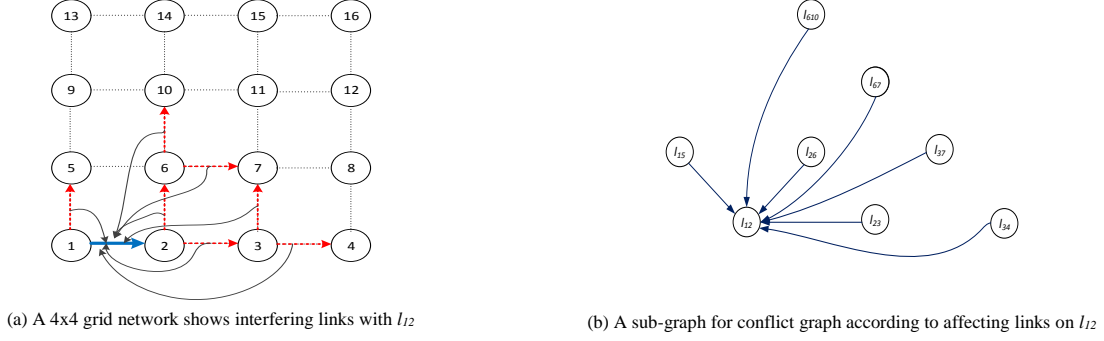


Figure 3: Conflict graph based on physical power gain model

occurs in sending nodes. Notice that to compute(8), it is sufficient to consider only a conflict graph associated with $S(t)$ and $\hat{S}(t)$ instead of the whole network graph.

Definition 7: The conflict graph $G_{S,\hat{S}}^c = (\hat{N}^c, \hat{L}^c)$ is a sub-graph of $G^c = (N^c, L^c)$ such that $\hat{N}^c \subset (S \cup \hat{S})$ and $\hat{L}^c = \{(L_S, L_{\hat{S}}) : l_S \in S(t), l_{\hat{S}} \in \hat{S}(t), w_{l_S \rightarrow l_{\hat{S}}} \geq 1\}$.

The conflict graph $G_{S,\hat{S}}^c$ can be generated as follows. Every link $l \in (S \cap \hat{S})$ easily can be removed from $G_{S,\hat{S}}^c$, because according to (8) have not any effect on calculating improvement factor. Each link $l \in (S \cup \hat{S}) - (S \cap \hat{S})$ denoted a node in conflict graph $G_{S,\hat{S}}^c$. If the fraction of power gain in receiving node of a link l on maximum permissible noise at that node in contributed by activity on link \hat{l} is larger than unity, we establish a link between nodes corresponding to l and \hat{l} . Since both S, \hat{S} are feasible schedule according to pick algorithm, no two links in the same schedule can interfere with each other in conflict graph $G_{S,\hat{S}}^c$. So all edges in $G_{S,\hat{S}}^c$ are between two nodes respectively from S, \hat{S} . Every links in G can easily compute its own queue length according to number of packets in two end points of the link. We consider that every node has unique identifier and nodes know the range of these identifiers.

Considering the structure of the conflict graph is considered, the resulting graph may be, is not connected. So the graph $G_{S,\hat{S}}^c$ is composed of connected sub-graphs and the compare algorithm can be executed independently in every connected sub-graph. Our compare algorithm has two main procedures that are executed consecutively: “Path Establishment” and “Weights Flooding”. The *Path Establishment* procedure construct a spanning tree in every connected sub-graph of $G_{S,\hat{S}}^c$. The *Weight Flooding* procedure broadcast computed improvement factor in spanning tree to inform all nodes about comparing result.

Although the compare algorithm operates on conflict graph $G_{S,\hat{S}}^c$ but it can be easily transformed to operate on network graph G by informing two end point of each link in $S \cup \hat{S}$ after each operation on $G_{S,\hat{S}}^c$. Every disconnected sub-graph of the conflict graph $G_{S,\hat{S}}^c$ can execute its procedures concurrently with others, because their resulting schedule is guaranteed to be feasible even their result of compare algorithm are different.

2.5.1. Spanning Tree Establishment Procedure

In this procedure a spanning tree $T_{S,\hat{S}}^c$ is constructed in every connected sub-graph. Our spanning tree construction algorithm is based on distributed depth first search (DDFS) algorithm developed by Awerbuch [26]. This algorithm requires a single initiator node, but according to nature of every connected sub-graph in our conflict graph model, developing the algorithm based on single initiator may not possible in distributed manner. We eliminate this restriction by some modification of the DDFS algorithm by considering minimum node ID initiator.

The DDFS algorithm simply operates as follow. First the initiator node send a DISCOVER message to itself. Every node that receive a DISCOVER message, it set the sender as its parent then informs all its neighbors by VISITED message. Every mode that receive VISITED message, learn that the sender joined to tree, so remove it from unvisited neighbors and send back an ACK message to the sender. Whenever the ACK message have been collected from all neighbors, it sends a RETURN message to itself. Upon receipt of RETURN message, it means

that the search is resumed from that node and if there exist an unvisited neighbor, the DISCOVER message is sent to it, otherwise the RETURN message is delivered to its parent.

The simple modification on this algorithm eliminates single initiator restriction. We assumed ID of every node in conflict graph corresponds to a scheduled link l_{ij} such as an active link between two nodes n_i and n_j as $ID(l_{ij}) = (\min(i, j), \max(i, j))$. According to link's ID comparison, we use specific ordering definition as follow

$ID(l_{ij}) > ID(l_{st})$ iff $i < s$ or $(i = s \text{ and } j < t)$.

We assume every node knows its neighbor's ID. So every node that have minimum ID among its neighbors, acts as an initiator. This can be lead to multiple spanning trees constructed by multiple initiators. We have to allow only one of these to complete and kill all of the others. We let just initiator with minimum ID complete the tree construction. This can be done by including initiator ID in DISCOVER message. So every time a node receive DISCOVER message with lower initiator ID than it currently has, it discards all previously information and join to the new tree. This algorithm formally represented in Algorithm 2.

Algorithm 2: SPANNING TREE CONSTRUCTION procedure

- In every node $x \in G_{S,\mathcal{S}}^c$ do
- a. On initializing do
 1. $N =$ the set of all neighbors nodeID
 2. $Parent =$ the parent of this node in resulting spanning tree
 3. $Flag =$ the set of binary flag kept for all neighbors and set to true after VISITED message is sent to corresponding neighbor and set to false after receiving ACK from that node
 4. $Unvisited =$ the set of all unvisited neighbors
 5. If $ID(l_{ij}) < \min \left(ID \left(Neighbors(l_{ij}) \right) \right)$ send DISCOVER(l_{ij}) to l_{ij}
 - b. On receiving DISCOVER(y) do
 1. if $y < Parent$ then call REJOIN(y)
 2. $Parent = y$
 3. For every node $z \in N \setminus \{y\}$ send VISITED(x) to z
 4. $Flag(z) = \text{true}$
 - c. On receiving VISITED (y) do
 1. Remove y from $Unvisited$
 2. Send ACK(x) to y
 - d. On receiving ACK(y) do
 1. $Flag(y) = \text{False}$
 2. If for every $y \in N$, $Flag(y) = \text{false}$ then send RETURN(x) to itself (i.e. x)
 - e. On receiving RETURN(y) do
 1. If there exist $y \in Unvisited$ then send DISCOVER(x) to y and remove y from $Unvisited$
 2. Else send RETURN(x) to $Parent$
 - f. On REJOIN(y)
 1. $Parent = y$
 2. $Unvisited = N$
 3. send RETURN(x) to x
-

Proposition 1. The Spanning Tree Establishment procedure construct a spanning tree $T_{S,\mathcal{S}}^c$ of a given connected sub-graph in $O(N^3)$.

Proof: To prove that the result of the algorithm 5 is spanning tree, we show that, all of nodes belong to the constructed sub-graph, and that sub-graph does not contain any cycle.

Because of the correctness of flooding a DISCOVER message to every unvisited neighbor in RETURN receiving subroutine, easily concluded that every node receive DISCOVER message and instantly join to tree, so the constructed graph is spanning sub-graph. Since every node, sets only one node as its parent whenever it receive DISCOVER message, there can be no cycles and the resulting graph is spanning tree.

According to the algorithm, except for the initiator, every node sends *VISITED* and *ACK* messages to the neighbors, and consequently after gathering all acknowledges from its neighbors, it sends *DISCOVER* message to unvisited neighbors, and finally it send back a return message to its parent. So $3*D + 1$ messages are delivered at every node in single initiator mode. Because of every node can be an initiator, the worst case for the communication cost of the algorithm is $O(|\hat{N}|, D)$ for each node. To compute the time complexity, note that each node joining operation till to return to its parent can take at most $O(D)$ times, and at most every joining operation can be repeated for every initiator, we need $O(|\hat{N}|, D)$ time for the operation to complete.

2.5.2. Weight Flooding Procedure

In second part of the compare algorithm, we use the constructed spanning tree which is the output of the previous procedure for comparing total weights of two consecutive schedules according to improvement sign of new schedule based on(8). The weight flooding procedure initiates from the leaf nodes, such that the weight of nodes disseminate to its parents. Every node n_i knows which schedule it belongs to. So whenever a node n_i receives all of the weights from its children, it computes the difference between the overall weights of schedules $S(t)$ and $\hat{S}(t)$ of incoming weights, and updates its weight consequently. Afterward it sends the computed weight to its parent. Upon the weight differences updated in the root, the result is total improvement sign of new schedule that should be flooded in the tree. For decreasing of communication cost of flooding, root only attempt to flood where the improvement sign is positive and will be silent where it is negative. Once all the nodes in spanning tree receive the improvement sign, all of them switch to better schedule.

Proposition 2. *The Weight Flooding procedure switch to better schedule in $O(N^3)$.*

Proof: It is clear that when difference of weights is received in the root, and after updating the root's weight, the computed result value indicate the choice of better schedule. So if the root broadcast this value throughout of spanning tree, all of the node will be able to decide between two consecutive schedules.

The depth of the spanning tree would be take most $O(\hat{N})$ and receiving weights from all children and broadcasting improvement sign to children can be at most $O(\hat{D})$. We know $\hat{N} < N^2$ and $\hat{D} < N$, so the whole procedure terminates at $O(N^3)$.

4 Experimental Results

In this section, we evaluate the performance of our learning automata based scheduling scheme through detailed numerical studies. A grid wireless multi hop mesh network topology generated using the following method. We first specify a square region with the area of 2500×2500 meter-square that has the width [0, 2500] on the x axis and the height [0, 2500] on the y axis. Then we generate 6×6 number of nodes and placed in the same distance from each other. We use Rayleigh fading for signal fading model and free space path loss model for the loss in signal strength. All of mesh routers are fixed and only client nodes can be mobile. Default radio frequency, radio reception sensitivity and radio reception threshold are subsequently assumed 2.4GHz, -91dBm and -81dBm and also for simplicity ambient noise was fixed to -50dBm. Sensing and receiving modes are determined based on physical situations of radio signal propagation.

In the first scenario We assume that four concurrent flows such that F1 from node (1,6) to (6,2), F2 from (1,5) to (6,6), F3 from (1,1) to (1,3) and F4 from (4,6) to (6,2) . First we study the evolution of the average length of the queues and throughput for each flow. The simulation results, as shown in the **Error! Reference source not found.** and **Error! Reference source not found.** indicates the total number of packets and throughput for each flow tends to a constant value is steady state. The first result of our observation is the throughputs of the flows are converged to completely different values according to separation of the source and destination. For example the longest hop flow F2 achieves the lowest throughput versus F3 that reaches to highest throughput with only 2 hops length.

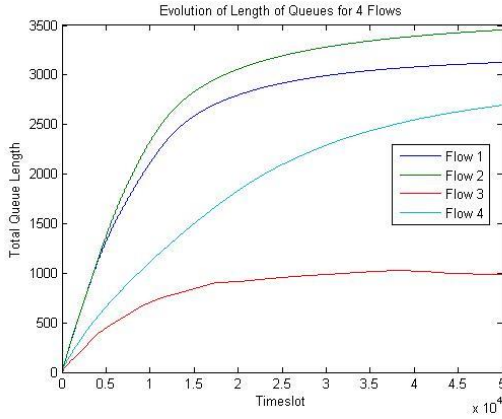


Figure 4: Evolution of queued backlog for the flows

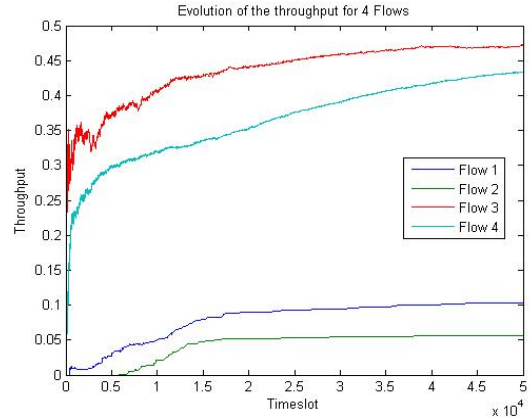


Figure 5: Evolution of throughput for the flows

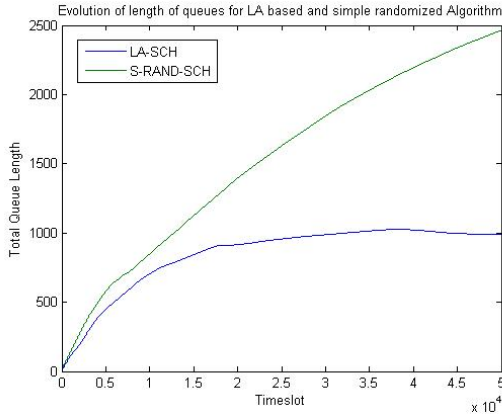


Figure 6: Performance of learning automata based algorithm on queues length evolution

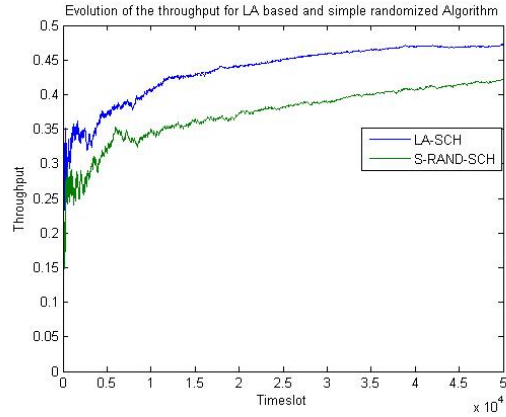


Figure 7: Performance of learning automata based scheduling in throughput evolution

In the second scenario we compare the performance of simplistic random scheduling algorithm and learning automata based scheme that gradually learn truly decision on neighbor selection. The performance of our algorithm are illustrated in **Error! Reference source not found.** It shows the total backlog changes quickly converge to steady state value compared to classic randomized pick algorithm. Although the total backlog value in steady state significantly lower than randomized pick scheme, suggest that in comparison with randomized pick, greater number of packets to the destination is reached. This is because of blind randomly choosing of neighbors in classic pick algorithm. **Error! Reference source not found.** shows the similar result is obtained when we study throughput among two discussed schemes. Clearly it shows that throughput for shortest and longest hop flows of the proposed method is significantly higher than classical pick algorithm.

Investigating the interference property can offer good insight into the operation of our algorithm. Fig. 4 clearly indicates evolution of interference according to network connectivity across the time slots. Obviously by increasing of network connectivity, potential interference will be increased, so strongly connected network has high potential interference. To measure the quality of the solution in term of channel interference, we track the behavior of the system by connectivity growing beside of interference increasing. Increasing in proportional connectivity vs. interference shows that algorithm effectively decreases interference and increases network topology connectivity proportionally.

5 Conclusions

We have developed enhanced pick and compare based scheduling algorithm for physical interference model in a wireless network that are throughput-optimal. A more realistic collision and interference conflicting graph based on physical model presented in this paper. Most of the throughput-optimal scheduling strategies leads to an NP-hard problem should be solved in a centralized algorithm in every time slot. We proposed learning automata based distributed pick and compare algorithm that can achieve 100% of the available capacity of the multi-hop wireless network in a polynomial message and time complexity.

References

1. Tassiulas, L. and A. Ephremides, *Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks*. Automatic Control, IEEE Transactions on, 1992. **37**(12): p. 1936-1948.
2. Tassiulas, L. *Linear complexity algorithms for maximum throughput in radio networks and input queued switches*. in *INFOCOM '98. Seventeenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*. 1998.
3. Xiaojun, L. and N.B. Shroff, *The impact of imperfect scheduling on cross-Layer congestion control in wireless networks*. Networking, IEEE/ACM Transactions on, 2006. **14**(2): p. 302-315.
4. Eryilmaz, A., A. Ozdaglar, and E. Modiano. *Polynomial Complexity Algorithms for Full Utilization of Multi-Hop Wireless Networks*. in *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*. 2007.
5. Yi, Y., G. De Veciana, and S. Shakkottai. *On optimal MAC scheduling with physical interference*. in *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*. 2007. IEEE.
6. Behzad, A. and I. Rubin, *Optimum integrated link scheduling and power control for multihop wireless networks*. Vehicular Technology, IEEE Transactions on, 2007. **56**(1): p. 194-205.
7. Negi, R. and A. Rajeswaran, *Physical layer effect on mac performance in ad-hoc wireless networks*. Proc. of Communication, Internet and Information Technoogy, 2003.
8. Zhou, Y., et al., *Throughput Optimizing Localized Link Scheduling for Multihop Wireless Networks Under Physical Interference Model*. Parallel and Distributed Systems, IEEE Transactions on, 2014. **25**(10): p. 2708-2720.
9. Wan, P.-J., X. Xu, and O. Frieder. *Shortest link scheduling with power control under physical interference model*. in *Mobile Ad-hoc and Sensor Networks (MSN), 2010 Sixth International Conference on*. 2010. IEEE.
10. Xu, X. and M. Song. *Stable wireless link scheduling subject to physical interference with power control*. in *Computer Communication and Networks (ICCCN), 2014 23rd International Conference on*. 2014. IEEE.
11. Iyer, A., C. Rosenberg, and A. Karnik, *What is the right model for wireless channel interference?* Wireless Communications, IEEE Transactions on, 2009. **8**(5): p. 2662-2671.
12. Joo, C., X. Lin, and N.B. Shroff, *Understanding the capacity region of the greedy maximal scheduling algorithm in multihop wireless networks*. IEEE/ACM Transactions on Networking (TON), 2009. **17**(4): p. 1132-1145.
13. Gupta, P. and P.R. Kumar, *The capacity of wireless networks*. Information Theory, IEEE Transactions on, 2000. **46**(2): p. 388-404.
14. O'Neill, D., et al. *Wireless NUM: Rate and reliability tradeoffs in random environment*. in *Wireless Communications and Networking Conference, 2009. WCNC 2009. IEEE*. 2009. IEEE.
15. Foschini, G. and J. Salz, *Digital communications over fading radio channels*. Bell Syst. Tech. J, 1983. **62**(2): p. 429-456.

16. Thathachar, M.A. and P.S. Sastry, *Networks of learning automata: Techniques for online stochastic optimization*. 2003: Springer.
17. Beheshtifard, Z. and M.R. Meybodi, *Online Channel Assignment in Multi-Radio Wireless Mesh Networks Using Learning Automata*. International Journal of Soft Computing and Software Engineering [JSCSE]: p. 0-0.
18. Joshi, T., et al., *SARA: stochastic automata rate adaptation for IEEE 802.11 networks*. Parallel and Distributed Systems, IEEE Transactions on, 2008. **19**(11): p. 1579-1590.
19. Torkestani, J.A. and M.R. Meybodi, *Mobility-based multicast routing algorithm for wireless mobile Ad-hoc networks: A learning automata approach*. Computer Communications, 2010. **33**(6): p. 721-735.
20. Xing, Y. and R. Chandramouli, *Stochastic learning solution for distributed discrete power control game in wireless data networks*. Networking, IEEE/ACM Transactions on, 2008. **16**(4): p. 932-944.
21. Kalman, R. and J. Bertram, *Control System Analysis and Design Via the "Second Method" of Lyapunov: II—Discrete-Time Systems*. Journal of Fluids Engineering, 1960. **82**(2): p. 394-400.
22. Ünsal, C., *Intelligent navigation of autonomous vehicles in an automated highway system: Learning methods and interacting vehicles approach*. 1997, Virginia Polytechnic Institute and State University.
23. Khalil, H.K. and J. Grizzle, *Nonlinear systems*. Vol. 3. 1996: Prentice hall New Jersey.
24. Brar, G., D.M. Blough, and P. Santi. *Computationally efficient scheduling with the physical interference model for throughput improvement in wireless mesh networks*. in *Proceedings of the 12th annual international conference on Mobile computing and networking*. 2006. ACM.
25. Jain, K., et al., *Impact of interference on multi-hop wireless network performance*. Wireless networks, 2005. **11**(4): p. 471-487.
26. Awerbuch, B. *Optimal distributed algorithms for minimum weight spanning tree, counting, leader election, and related problems*. in *Proceedings of the nineteenth annual ACM symposium on Theory of computing*. 1987. ACM.