Ad Hoc Networks 45 (2016) 65–79

Contents lists available at ScienceDirect

Ad Hoc Networks

journal homepage: www.elsevier.com/locate/adhoc

Maximal throughput scheduling based on the physical interference model using learning automata

Ziaeddin Beheshtifard[a,*], Mohammad Reza Meybodi[b]

[a] Department of Computer Engineering and Information Technology, Islamic Azad University, Qazvin Branch, Qazvin, Iran
[b] Department of Computer Engineering, Amirkabir University of Technology, Tehran, Iran

ARTICLE INFO

Article history:
Received 17 September 2015
Revised 25 February 2016
Accepted 26 February 2016
Available online 3 March 2016

Keywords:
Wireless network
Maximal scheduling
Physical interference
Learning automata

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. This paper improves the general randomized scheduling method by using learning automata based framework that allows throughput optimal scheduling algorithms to be developed in a distributed fashion. A distributed scheduling algorithm that operates on more realistic conflict graph was proposed based on the physical interference model. This model uses a combination of a distributed learning automata based on the pick algorithm and an algorithm that compares successive scheduling solutions. Comparison was made by creating spanning tree on the conflict graph of the two consecutive schedules. Briefly, a distributed scheduling schemes was proposed, that: (i) is throughput optimal, (ii) intelligently choose links for new schedule, and (iii) message and time complexity is in $O(n^3)$.

1. Introduction

Medium access control (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 generated by fully stochastic network dynamics. In addition, scheduling is the main part of joint scheduling, routing and congestion control policies in network utility maximization frameworks and the efficient algorithms 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 [1] in constraint queuing systems. They studied the throughput of scheduling policy, which is characterized by stability throughput region Λ, 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. Though their method reaches all the achievable throughput regions, the nature of the proposed method is based on the centralized computations that may have some issue in realistic applications. Based on simple pick-and-compare algorithms comparison, they proposed a low complexity randomized algorithm that greedily reach optimal throughput, but was still executed on centralized fashion [2].The approach presented in the study by [2] is as follows. In each time slot, a candidate solution is randomly obtained for the maximum weighted matching problem. The maximum weights are replaced, if the sum of the weights of the selected links is higher than the value of the current solution. The use of this approach guarantees the achievement of 100% throughput under certain conditions of how the matching is obtained.

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

In a recent study by Eryilmaz et al. [4] inspired by pick-and-compare algorithm, they developed a scheduling problem in the two-hop interference model. They proposed a general framework to be developed in two-hop interference that allows some distributed algorithms with polynomial communication and computation complexity model in wireless networks. Another work was also presented based on the physical interference model that used a randomized algorithm for link activation based on transmission successfulness and two level of priority consideration. They proved their work in the multi-hop wireless networks context, guarantees full throughput efficiency in a distributed manner [5].

* Corresponding author. Tel.: +98 912 186 3244.
E-mail addresses: beheshti@qiau.ac.ir, zia.beheshti@gmail.com (Z. Beheshtifard), mmeybodi@aut.ac.ir (M.R. Meybodi).

http://dx.doi.org/10.1016/j.adhoc.2016.02.020
1570-8705/

Recently, there have been several efforts towards the analysis and design of wireless multi-hop networks under more general interference models than the graph-based interference model. In the study by [6], mixed-integer linear programming formulation was developed for minimizing the schedule length over a time division multiple access (TDMA) wireless multi-hop networks, based on the joint MAC scheduling and power control under the physical interference model. Our work is different from the proposed distributed algorithm by [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. Similar work by [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 [8–11] have addressed some challenges related to scheduling problem under the physical interference model, but their solutions only guarantee sub-optimal approximation of the maximum achievable throughput. The work presented in [10] develops interference-aware scheduling protocols under the arbitrary physical interference model, such that, it considers different transmission power control settings, that is, uniform and monotone power controls. The later work develops a greedy maximal scheduling as Longest-Queue-First (LQF) policy under the SINR interference model. This paper employed the ρ-local pooling technique, showed that LQF scheduling achieves zero throughput in the worst case and provided a sufficient condition for LQF scheduling to achieve maximum throughput in the special case of network topology.

In this paper, an adaptive learning automata was developed based on the distributed randomized scheduling algorithm that can fully utilize the throughput region of a multi-hop wireless network under the realistic Signal to Interference and Noise Ratio (SINR) model. We introduce a new conflict graph based on the physical model, such that it can be used in the construction of the spanning tree between two successive scheduling graphs. Also, we develop a new version of 'Compare' algorithm that can operate in a distributed fashion under the physical interference model to compare two consecutive scheduling. To do this, we used our physical conflict graph that is presented in the later. In Section 2, the system model from three different perspectives, including the interference model, network model and stochastic automata game model of the network was presented. In Section 3, collection of distributed algorithms based on randomized pick-and-compare method was proposed that could practically be developed in more realistic wireless network. Section 4 provides some simulation results and finally, the results of this paper were summarized in Section 5.

2. Models

2.1. Interference model

According to intrinsic difficulty in the modeling of wireless network physical layer, in contrast with wired link networks, wireless links capacity are dependent on one another, so they suffer from mutual interference required to model the wireless channel interference accurately. Several studies within the literature have considered the simplified graph based on the interference model. In a comparative study presented by [12], they investigated two of such model, the interference range model which uses fixed distance ranges, and the physical SINR model which specifies the 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 a different physical layer model will lead to a different result. Specifically, the interference range model misleadingly predicts the throughput as the function of the transmitted 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 $O(1/N^{3/2})$ [12]. These conflicting results revealed the importance of choosing more realistic physical layer model.

In terms of wireless resource allocation, the interference model is considerably important in wireless link scheduling algorithms. Many studies consider simple models of interference, such that the 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 simultaneously be activated. 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 becomes NP-Hard under more general interference model in wireless networks [13]. 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 [13]. Under these types of interference models conflicting links were predetermined by conflict graph, but in realistic wireless networks, interference on links was determined by global additive noise rather than distance metrics of transmission range.

The interference model can be formulated as follows. Consider a network of n wireless nodes, $n_1, n_2, \ldots, n_N$. A message from a transmitter n_s can successfully be 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 on 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 [14].

Some of the factors that have effect on the 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 the coding error and R_l is the rate of wireless link l [15]. Physical level factors are the second type of factors that the realistic rate of wireless links depends on. These factors constitute a probabilistic model for the channel state of physical wireless links [16]. 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 [16]:

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

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_{amb} is the ambient noise. The probability distribution of G is unknown to the network. The link rates can also vary, because the channel varies randomly, thereby resulting in congestion and queuing delay at the link buffers. The notations are listed in Table 1.

2.2. Network model

Considering a wireless mesh network where $N = \{1,2,\ldots,n\}$ 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

Table 1
Summary of notations.

Notation	Description
P_r	Signal strength of the message at the receiver n_r
I_r	Sum of all interferences at n_r
N_{amb}	Ambient noise
A	Path-loss exponent
BER	Bit error rate
θ_1	Coding error
R_l	Rate of wireless link l
G	Gain matrix
S	Vector of transmitter powers
K	Scaling parameter for the received power
Φ	Coding gain
L	Set of directed links
$A_l(t)$	Amount of data that arrives at link l on slot t
$\overrightarrow{S(t)}$	Scheduled links vector
$Q_l(t)$	Amount of data that backlogged at link l on slot t
Λ	Throughput region
$P_i(k)$	Action selection probability vector for ith node
λ	Learning parameter
$\beta(k)$	Received environmental response at cycle k
$a(k)$	Set of actions chosen by the automata at cycle k
$w_{zt\to xy}$	Edge weight in conflict graph
$G^c_{S,\hat{S}}$	Conflict graph of two successive scheduled links $s, \hat{s}$

also different power levels, it is even possible that two nodes i and j belonging to N are linked in only one-way, that is, (i, j) belongs to L, but (j, i) is not. Each link corresponds to a pair of transmitter node and receiver. It is assumed that all radio devices use half-duplex transmission where each radio has only one transceiver. Time was assumed to be divided into slots of unit length, and was synchronized between nodes, denoted by t, and the duration is adequately chosen such that one fixed size packet would be transmittable in a unit time slot.

Let $Q_l(t)$ represent the current amount of packets in transmitter node that should be served to receiver node of link l, which is also called the queue backlog. 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 transmitted to its destination is removed from the network. $A_l(t)$ is defined as the amount of data that arrives at transmitter node of l on slot t that must eventually be delivered to receiver node of l.

Definition 1. A schedule $\overrightarrow{S(t)} = (S_l(t) \in \{0, 1\} : l = 1, 2, \ldots, |L|)$, $S_l(t)$, 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.

Because of the 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 can be model with SINR parameters.

A scheduling algorithm, in this paper, selects $\overrightarrow{S(t)}$ based on the queue-sizes, $\vec{Q}(t'), t' \le t$, such that the dynamics of the system can be described as evolution of queues size, that may be written as:

$$Q_l(t+1) = [Q_l(t) - S_l(t)]^+ + A_l(t) \tag{2}$$

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

$$\lim_{t\to\infty} \sup_t (E[Q_l(t)]) < \infty$$

Definition 3. Throughput region Λ represents the set of all feasible transmission scheduling vectors such that the network is stable under the policy for any arrival process.

$$\Lambda = \left\{ \overrightarrow{\delta} : \overrightarrow{\delta} = \sum_{S_i \in \overrightarrow{S}} \alpha_i S_i, \quad 0 \le \alpha_i \le 1, \quad \sum_{i=1}^{\left|\overrightarrow{S}\right|} \alpha_i = 1 \right\} \tag{3}$$

2.3. Stochastic automata game

Here, an overview of the concept of driving automata games was presented by [17]. Abstractly, learning automata (LA) are adaptive decision making unit situated in a probabilistic environment. LA is widely used as a useful artificial intelligent tool 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 the action. A corresponding stochastic feedback is sent to the automaton based on the next action chosen. As this process progresses, the automaton learns how to choose the optimal action for the unknown probability distribution of the environment asymptotically. An important property of the learning automaton is its ability to improve its performance with time, while in 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 [18], congestion control [19], routing [20] and power control [21].

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 update is performed, the LA selects the action to be performed 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 stated by following the update equation:

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

where $p_k \in \Re^N$, is called the state vector and $\xi_k \in \Re^N$, is the noise vector and b is the step size or learning parameter. A general reinforcement scheme has the form of Eqs. (5) and (6):

$$\begin{aligned} p_i(n+1) = {} & p_i(n) - (1-\beta(n))g_i(p(n)) \\ & - \beta(n)h_i(p(n)) \quad if\, \alpha(n) \ne \alpha_i \end{aligned} \tag{5}$$

$$\begin{aligned} p_i(n+1) = {} & p_i(n) - (1-\beta(n))\sum_{j\ne i} g_i(p(n)) \\ & - \beta(n)\sum_{j\ne i} h_i(p(n)) \quad if\, \alpha(n) = \alpha_i \end{aligned} \tag{6}$$

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. According to functions g and h, different schemes of reinforcement can be used. The simplest and most commonly used of them is reward-inaction ($L_{R\text{-}I}$). The $L_{R\text{-}I}$ algorithm updates the action probabilities described as follows:

$$\begin{aligned} p_i(k+1) &= p_i(k) + \lambda\beta(k)(1 - p_i(k)) \\ p_j(k+1) &= p_j(k) - \lambda\beta(k)p_i(k) \quad \forall j \ne i \end{aligned}$$

The above algorithm can be rewritten in vector notation as:

$$P(k+1) = P(k) + \lambda\beta(k)(e_i - P(k)) \tag{7}$$

Here, e_i is the unit vector with ith component unity where the index i corresponds to the action selected at k and λ is the learning (or step-size) parameter satisfying $0 < \lambda < 1$. As compared to $L_{R\text{-}P}$, 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

This section establishes the scheduling algorithm for obtaining the 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-compares approach introduced by Tassiulas [2] presented as gravitational search algorithm (GSA) in the algorithm depicted in Fig. 1x in this study. This is based on the idea that finding the max-weight schedule is 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.

The following theorem was stated according to the general property of the mentioned algorithm.

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

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

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

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

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

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

Proof. The detailed proof is given by [2].

In Section 1, the characteristics of physical interference model and general randomized scheduling algorithm that satisfy important stability property of queuing network were described. This section establishes the use of 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 use learning automata based game theoretic approach to exploit contention pattern on its neighborhoods. Request to Send/Clear to Send (RTS/CTS) followed 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.

The objective of this section is to develop two generic *Pick* and *Compare* algorithm that consecutively operate in repeated time slots. In the pick step, a stochastic learning automata based on the algorithm attains a feasible schedule in distributed fashion. The algorithm gradually reaches the 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 the step was used in making decision about whether an old schedule should be hold or not.

Pick and *Compare* algorithms operate consecutively in each time slot. Link scheduling for data transmission is updated at the beginning of each time slot. According to the same shared wireless medium and to prevent data and control message collision, the time slot is divided into two separated intervals, called *Control Messaging Period* and *Data Messaging period* (Fig. 2). Both pick and compare algorithms required to operate the exact coincidence in time and node should be synchronized in time. Time synchronization can be relaxed by using a buffer to accommodate propagation delay.

3.1. Pick algorithm

This section describes the pick algorithm that gradually learns contention patterns in the neighborhood of each node. Every node in this network uses a controller equipped by two independent learning automata. The first automaton supports decision for nodes to start activation for link scheduling, and the second one supports decision for neighbor selection if the first decision was positive for participation in link activation. Fig. 3 represents the logical orders of steps required for activation of a link. After pick algorithm, two possible conditions could occur. The first condition is when node withdraws participation in link activation (that is, endpoint W) where the automaton should be penalized. In the second case, the automaton's operation is postponed to that of the 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 attempt to link activation. Another event is that more than one neighbor of node are attempting to establish a connection. In this case, the receiving node does not get the 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 sides of the link withdraw link activation. Finally, if the receiving node senses another transmission during CTS sending phase, it withdraws. This ensures that two interfering node does not become the 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.

Proposition 1. The pick algorithm takes at most two message transmissions per node to terminate.

Proof. According to the algorithm depicted in Fig. 3, each node that participate as link activator, at most sends one RTS to selected neighbor and finally sends a CTS to it if the activator node receives a CTS from selected neighbor. Another nodes just sends one CTS message to activator node.

This section analyzes the stability of learning method in the game of multi-automata. A play $a(t) = (a_1(t), \ldots, a_N(t))$ 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 ith automata. Also $\beta(t) = (\beta_1(t), \ldots, \beta_N(t))$ is the payoff vector, such that $\beta^i(t)$ is the payoff to ith automata. Let the action set of ith automata be denoted by A_i with $|A_i| = r_i$, such that r_i is the number of neighbors of the node i. In defining the function $F^i : \{a_1, \ldots, a_N\} \to [0, 1], 1 \leq i \leq N$

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

Algorithm 1: GENERIC SCHEDULING ALGORITHM (GSA)

1. t = 0
2. **Do**
3. $\vec{S}(t) = \text{pick}()$
4. $[\vec{S}(t=1), \varepsilon] = \text{compare}(\vec{S}(t), \vec{\hat{S}}(t))$
5. $t = t+1$
6. **while**($\varepsilon < threshold$)

Fig. 1. Generic Scheduling Algorithm (GSA).

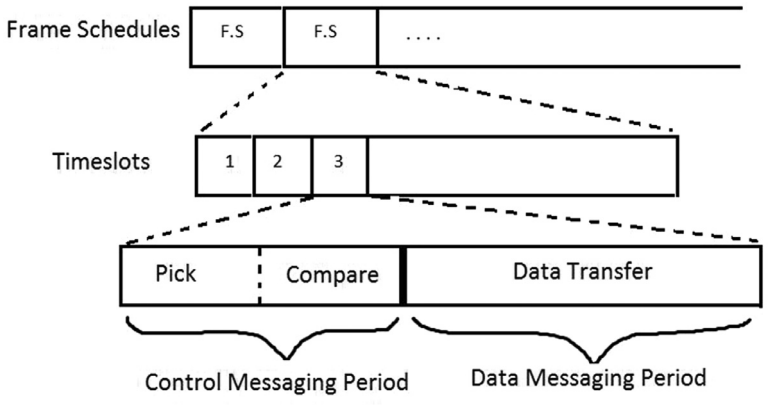


Fig. 2. Timeslot configuration for data and control messaging.

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

$F^i(a^*) \ge F^i(a)$ for all

$a = (a_1^*, \ldots, a_{i-1}^*, a_i, a_{i+1}^*, \ldots, a_N^*), \ a_i \ne a_i^*$

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

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

Theorem 3. *Considering 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 exists a scalar function,* $V(x)$ *is continuous in* x *such that:*

(1) $V(X) > 0$ for $X \ne 0$.
(2) $\Delta V(X) < 0$ for $X \ne 0$
(3) $V(0) = 0$
(4) $V(X) \to \infty$ *as* $X \to \infty$.

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

Theorem 4. *If the function* $f(X)$ *is defined as:* $f(X)_1 < X_1$ with $f(0) = 0$, *for some set of* $X \ne 0$, *the system is asymptotically stable and one of its Lyapunov function is:* $V(X) = X_1$.

These two theorems and their proofs have been given in [22].

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 [23]. This study easily shows that the 1st norm of the expected value of all sub-optimal action converges to zero. For example, for ith automaton if a_1 is an optimal action and $\overline{P_i}(t+1)$ is a vector of the expected value for other sub optimal action in steady state, it can be shown that:

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

It is known that every automaton in the network choose an action independently base on its action probability vector, so this analysis can be extended to the matrix of all probability vectors. So if we define $f(\bar{P}(t)) = \bar{P}(t+1)$ and extend $P(t)$ to the 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 [24].

3.2. 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 the current schedule and select new schedule as:

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

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

For computing Eq. (8), algorithm have to traverse two graph associated with $\hat{S}(t)$ and $S(t)$. Conflict graph associated with these two schedules were used to capture data concerning the 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 [25]. In its current design, the model fails to capture the cumulative effect of interference. Simultaneous activation of multiple links can occasionally cumulate enough interference to disrupt a transmission even though none of these links alone is harmful for the transmission.

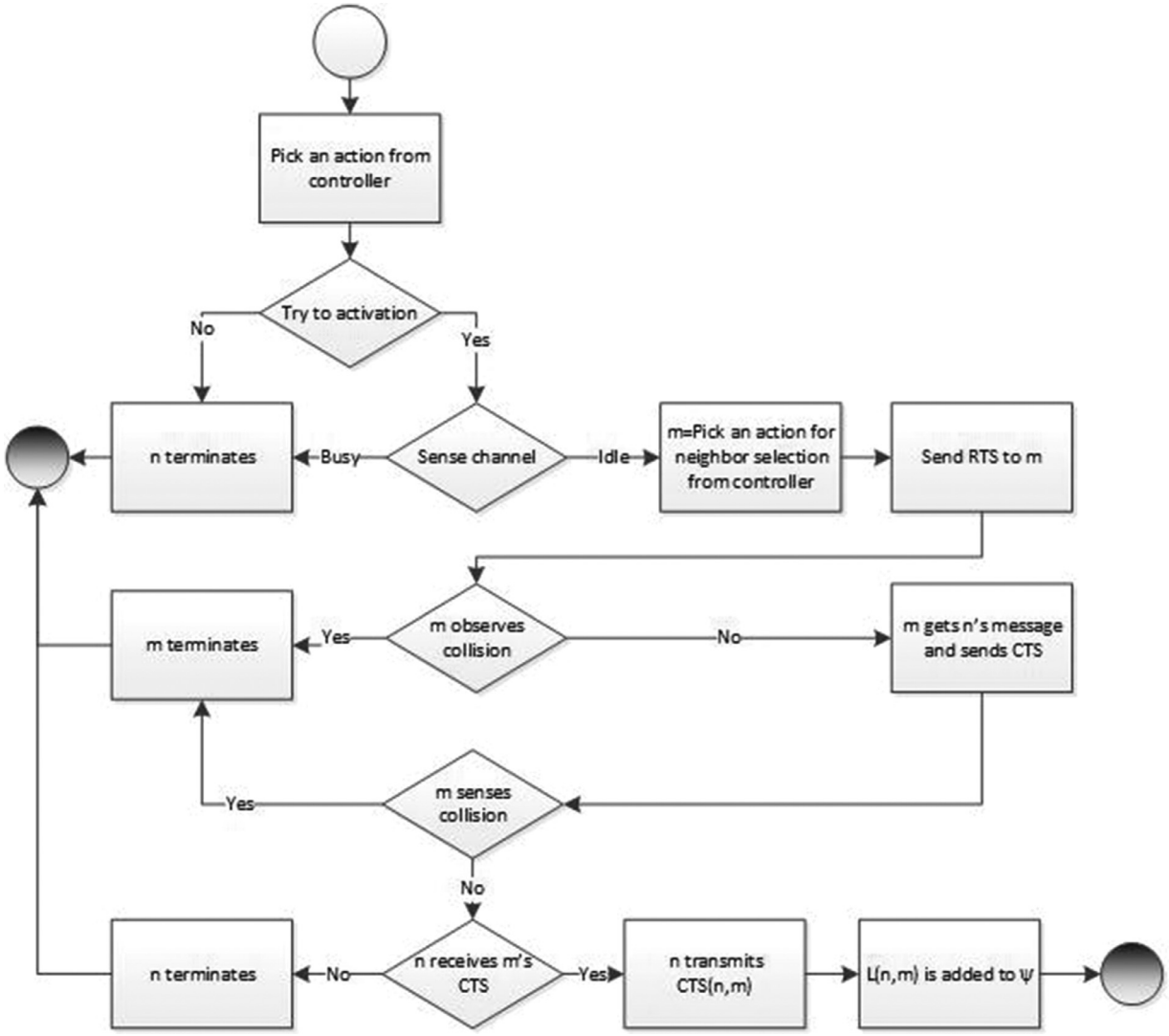


Fig. 3. Pick algorithm steps for acquiring feasible schedule.

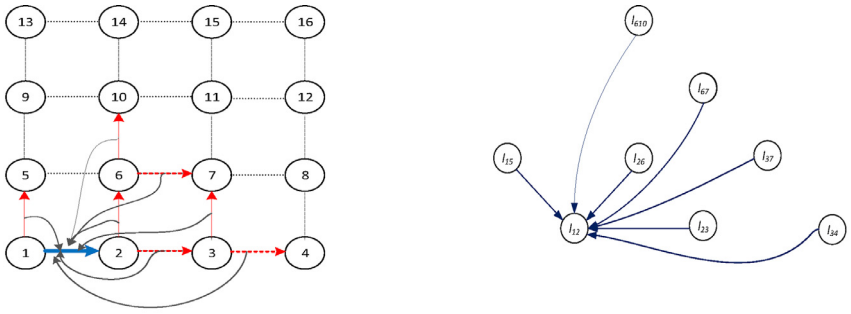
In the physical interference model, conflicts are not represented as binary when simultaneous transmissions occur in two interfering links. Suppose node x wants to transmit to node y. The signal power gain of x's transmission is calculated as received at y. The transmission is successful if $SNR_{xy} > SNR_{thresh}$, 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 ambient noise (N_{amb}) and the interference owing to the power gain of the 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 $G_{xy}/N_{amb} > SNR_{thresh}$ (that is, SNR exceeds the minimum threshold in the presence of ambient noise only). Interference in the physical model gradually increases as more neighboring nodes transmit and become unacceptable when the noise level reaches a threshold, because the conflicts are not binary. According to this observation, [26] presented physical interference based on conflict graph, such that they are 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 $w_{zt\to xy}$, indicates the fraction of the permissible noise at node y from link l_{xy} to still be operable, in contribution to the activity on link l_{zt}. In other form:

$$w_{zt\to xy} = \frac{G_{zy}}{G_{xy}/SNR_{thresh} - N_{amb}} \tag{9}$$

where G_{zy}, G_{xy} are the signal power gain of the two transmitters 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 $N = \{n_1, n_2, \ldots, n_{|N|}\}$, $L = \{l_{n_i,n_j} : n_i, n_j \in N\}$ then $G^c = (N^c, L^c)$ is the conflict graph for graph G such that:$N^c = L$, $L^c = \{(l_{zt}, l_{xy}) : l_{zt}, l_{xy} \in L, w_{zt\to xy} \geq 1\}$.

To demonstrate the definition of the physical interference based on conflict graph, the grid network was considered as shown in Fig. 4(a). According to our model, link lx will be connected to link ly as a directed edge,if based on Eq. (9) $w_{l_x\to l_y} \geq 1$. A sub-conflict graph for 4×4 grid network that shows interfering links in relation to l_{12} shown in the Fig. 4(b) based on the defined sub-graph of the total conflict graph. It was noticed that the interference extracted from the conflict graph represents all the physical situations that results in interference in receiving nodes. This observation will not be restrictive, because the pick algorithm ensures that link allocation for the current scheduling is feasible, so no interference occurs in the sending nodes. When computing Eq. (8), it is sufficient to consider only a conflict graph associated with $S(t)$ and $\hat{S}(t)$ instead of the whole network graph.



(a) A 4x4 grid network shows interfering links with l_{12} (b) A sub-graph for conflict graph according to affecting links on l_{12}

Fig. 4. Conflict graph based on the physical power gain model. (a) A 4×4 grid network shows interfering links with l_{12}. (b) A sub-graph for conflict graph according to affecting links on l_{12}.

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

The conflict graph $G^c_{S,\hat{S}}$ can be generated as follows. Every link $l \in (S \cap \hat{S})$ easily can be removed from $G^c_{S,\hat{S}}$, because according to Eq. (8), there is no effect in calculating the improvement factor. Each link $l \in (S \cup \hat{S}) - (S \cap \hat{S})$ denoted a node in conflict graph $G^c_{S,\hat{S}}$. If the fraction of the power gained in receiving node of a link l on maximum permissible noise at node contributed by activity on link $\hat{l}$ is larger than unity. This study establishes 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 $G^c_{S,\hat{S}}$. So all the edges in $G^c_{S,\hat{S}}$ are between two nodes, respectively from S, $\widehat{S}$. Every links in G can easily compute its own queue length according to the number of packets in two end points of the link. It is considered that every node has unique identifier and nodes know the range of these identifiers.

When the structure of the conflict graph is considered, the resulting graph may not be connected. So the graph $G^c_{S,\hat{S}}$ 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 constructs a spanning tree in every connected sub-graph of $G^c_{S,\hat{S}}$. 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 $G^c_{S,\hat{S}}$, but it can easily be transformed to operate on network graph G by informing two end point of each link in $S \cup \hat{S}$ after each operation on $G^c_{S,\hat{S}}$. Every disconnected sub-graph of the conflict graph $G^c_{S,\hat{S}}$ can execute its procedures concurrently with others, because their resulting schedule is guaranteed to be feasible even when the result of the compare algorithm is different.

3.2.1. Spanning tree establishment procedure

In this procedure, a spanning tree $T^c_{S,\hat{S}}$ is constructed in every connected sub-graph. Our spanning tree construction algorithm is based on the distributed depth first search (DDFS) algorithm developed by Awerbuch [27]. This algorithm requires a single initiator node, but according to the nature of every connected sub-graph in our conflict graph model, developing the algorithm based on single initiator may not be possible in distributed manner. This restriction was eliminated by some modification on the DDFS algorithm by considering the minimum node ID initiator.

The DDFS algorithm simply operates in the following manner. First, the initiator node sends a DISCOVER message to itself. For every node that receive a DISCOVER message, it set the sender as its parent, and then informs all its neighbors by VISITED message. Every node that received VISITED message learn that the sender joined to the tree, thus remove it from unvisited neighbors and send back an ACK message to the sender. Whenever the ACK message has 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. It is assumed that the 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, specific ordering definition was used as follow:

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

It is assumed that every node knows it neighbor's ID. So every node that have minimum ID among its neighbors, acts as an initiator. This can lead to multiple spanning trees constructed by multiple initiators. Only one of these is allowed to complete and kill all of the others. Only initiator with minimum ID is allowed to complete the tree construction. This can be done by including initiator ID in DISCOVER message. So every time a node receives DISCOVER message with lower initiator ID than it currently has, it discards all previous information and join to the new tree. This algorithm is formally represented in Algorithm depicted in Fig. 5.

Proposition 2. The spanning tree establishment procedure constructs a spanning tree $T^c_{S,\hat{S}}$ of a given connected sub-graph in $O(N^3)$.

Proof. To prove that the result of algorithm 2 in Fig. 5 is a spanning tree, it is shown that, all nodes belong to the constructed sub-graph, and that the sub-graph does not contain any cycle.

It was easily concluded that every node receives *DISCOVER* message and instantly join the tree, so the constructed graph is spanning sub-graph, because of the correctness of flooding a *DISCOVER* message to every unvisited neighbor in *RETURN* by receiving sub-routine. Since every node sets only one node as its parent node whenever it receives *DISCOVER* message, there can be no cycles and the resulting graph is a 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 acknowledgement from its neighbors, it sends *DISCOVER* message to unvisited neighbors, and finally it

Algorithm 2: SPANNING TREE CONSTRUCTION procedure

1. **Do** in every node $x \in G^c_{S,\hat{S}}$
2. On initializing **do**
3. N = *the set of all neighbors nodeID*
4. *Parent* = *the parent of this node in resulting spanning tree*
5. *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*
6. *Unvisited* = *the set of all unvisited neighbors*
7. **If** $ID(l_{ij}) < \min\left(ID\left(Neighbors(l_{ij})\right)\right)$ send $DISCOVER(l_{ij})$ to l_{ij}
8. On receiving *DISCOVER(y)* **do**
9. **if** $y <$ *Parent* then call *REJOIN(y)*
10. *Parent* = y
11. For every node $z \in N\backslash\{y\}$ send *VISITED*(x) to z
12. *Flag(z)=true*
13. On receiving *VISITED (y)* **do**
14. Remove y from *Unvisited*
15. Send *ACK(x)* to y
16. On receiving *ACK(y)* **do**
17. *Flag(y)* = *False*
18. **If** for every $y \in N$, *Flag(y)= false* then send *RETURN(x)* to itself (i.e. x)
19. On receiving *RETURN(y)* **do**
20. **If** there exist $y \in Unvisited$ then send *DISCOVER(x)* to y and remove y from *Unvisited*
21. **Else** send *RETURN(x)* to *Parent*
22. On *REJOIN(y)* **do**
23. *Parent* =y
24. *Unvisited* = N
25. send *RETURN(x)* to x

Fig. 5. Spanning tree construction procedure.

sends it back as a return message to its parent. So $3*D+1$ messages are delivered at every node in single initiator node. Since 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 it returns to its parent can take at most $O(D)$ times, and at most every joining operation can be repeated for every initiator, which needed $O(|\hat{N}|, D)$ time for the operation to complete.

3.2.2. Weight flooding procedure

In the second part of the compare algorithm, the constructed spanning tree was used, which is the output of the previous procedure for comparing total weights of two consecutive schedules according to the improvement sign of new schedule based on Eq. (8). The weight flooding procedure initiates from the leaf nodes, such that the weight of the 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 the total improvement sign of new schedule that should be flooded in the tree. For decreasing of communication cost of flooding, root only attempts to flood where the improvement sign is positive and will be silent where it is negative. Once all the nodes in the spanning tree receive the improvement sign, all of them switch to better schedule.

Proposition 3. The weight flooding procedure switch to better schedule in $O(N^3)$.

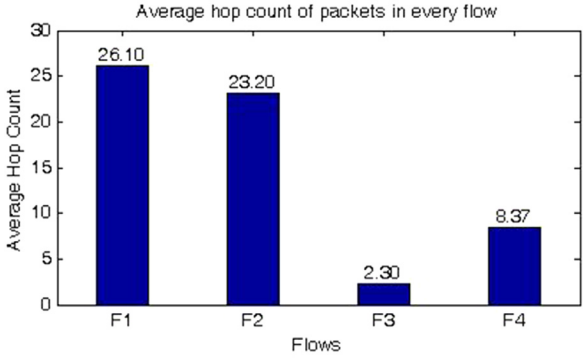


Fig. 6. The average of hop count of packets for every flow.

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 indicates the choice of better schedule. So if the root broadcast this value throughout the spanning tree, all of the nodes would be able to decide between two consecutive schedules.

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

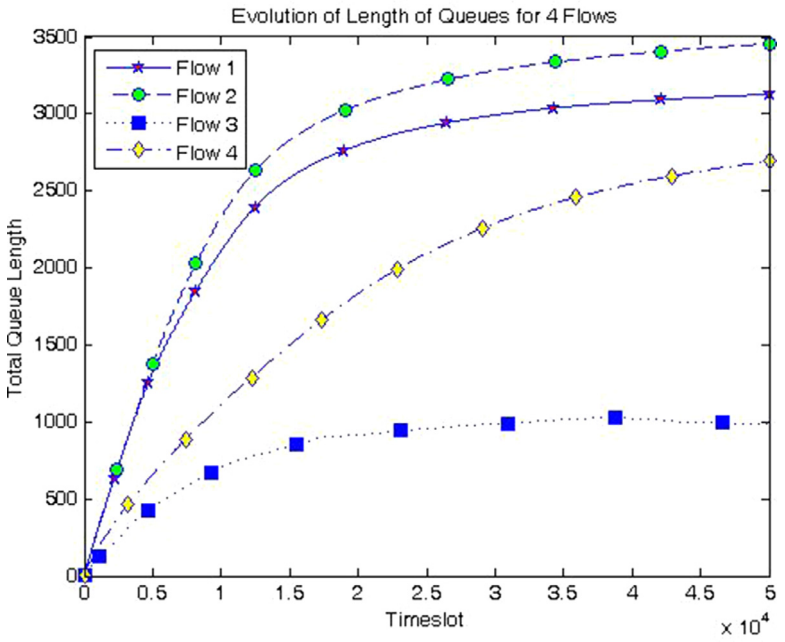


Fig. 7. Evolution of queued backlog for the flows.

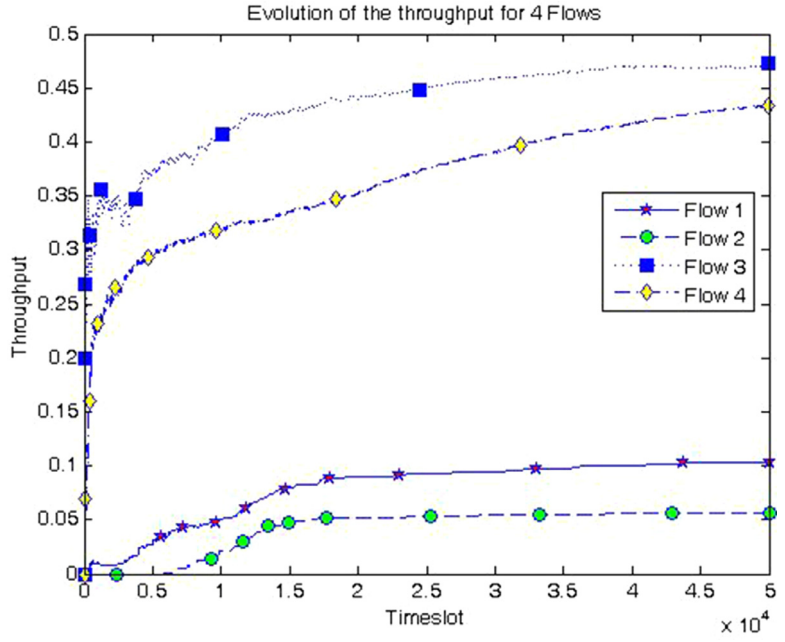


Fig. 8. Evolution of throughput for the flows.

4. Simulation results

In this section, the performance of our learning automata was evaluated based on the scheduling scheme through detailed numerical studies. The simulation platform used is JiST/SWANS [28]. It includes various components to support networking, routing, MAC protocols, radio transmission especially SINR models, noise models, node mobility models, signal propagation and fading models. All simulations were run on a desktop PC with Intel® Core™ i7 CPU running at 3.07 GHz with 16 GB of memory.

A grid wireless multi-hop mesh network topology was generated using the following method. A square region with the area of $2500 \times 2500\,m^2$ that has the width [0, 2500] on the x axis and the height [0, 2500] on the y axis was first specified. Then 6×6 number of nodes were generated and placed in the same distance from each other. Rayleigh fading was used for signal fading model and free space path loss model was used for the loss in signal strength. All mesh routers were fixed and only client nodes can be mobile. Default radio frequency, radio reception sensitivity and radio reception threshold was subsequently assumed to be 2.4 GHz, −91 dBm and −81 dBm and also

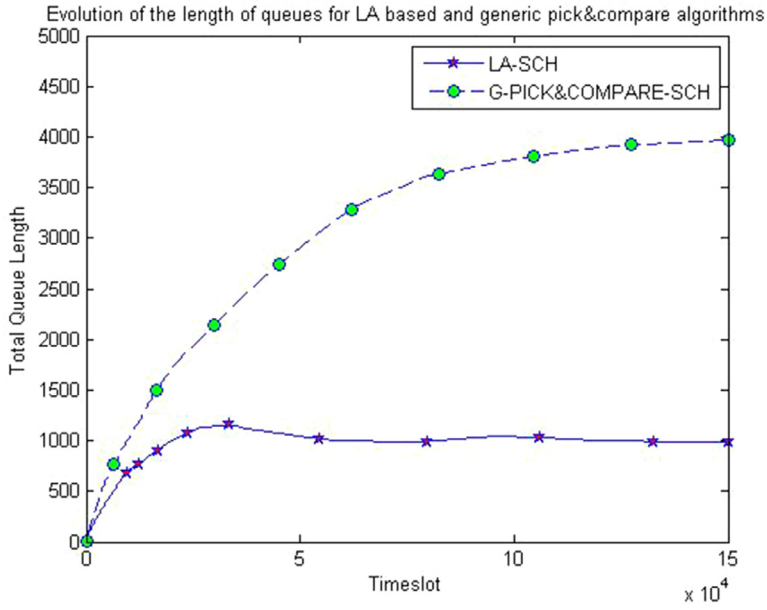


Fig. 9. Performance of learning automata based algorithm on queues length evolution.

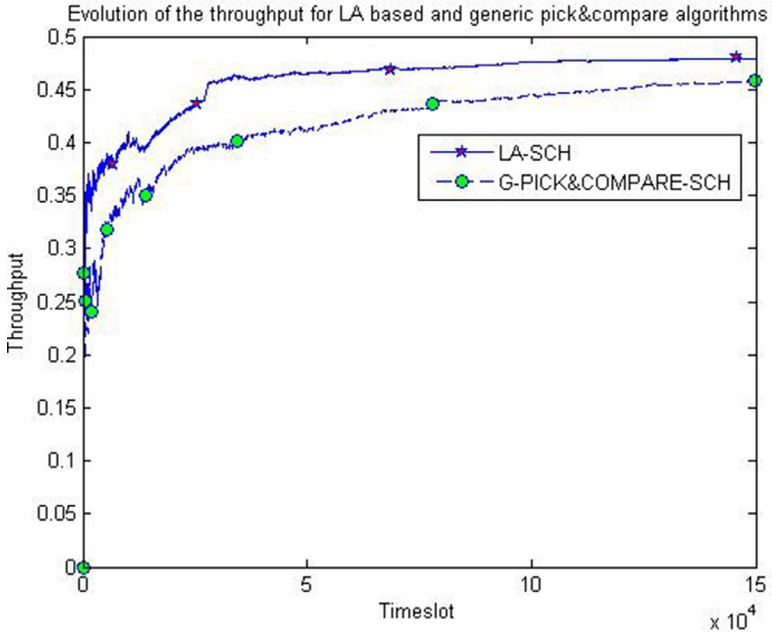


Fig. 10. Performance of learning automata based scheduling in throughput evolution.

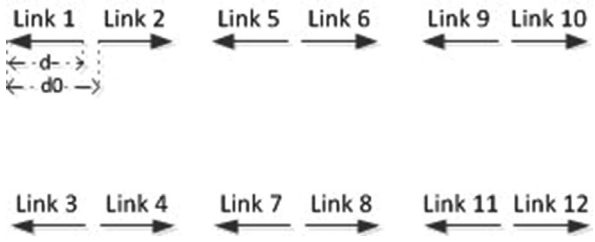


Fig. 11. Network of 12 links.

the simplicity ambient noise was fixed to −50 dBm. Sensing and receiving modes were determined based on the physical situations of the radio signal propagation.

In the first scenario, it was assumed 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, the average of hop count, the evolution of the average length of the queues and throughput for each flow was studied. Fig. 6 shows the average number of hops for packets of every flow. According to physical interference model, the number of hops mainly depends on transmitter power level and total additive interference in destination. The comparison of hop-count between two flows F2 and F4, it is observed that F2 in its path crosses the F1 and F4. So due to the common node located in the F2 and F4 as well as between F2 and F1, congestion and physical interference would be high in nodes that carrying packets of F2. According to learning automata based neighbor selection, F2 gradually learns to choose an alternative path from the source to destination. The simulation results,

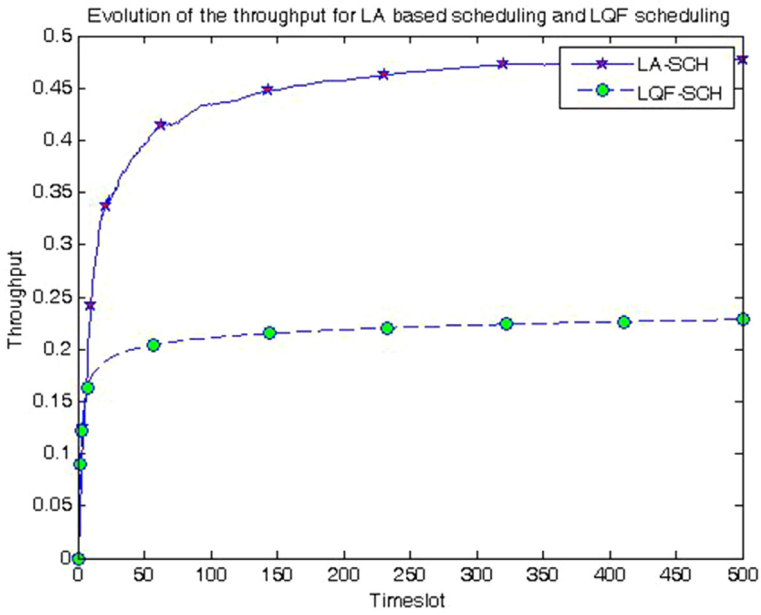


Fig. 12. Evolution of the throughput for LA based scheduling and LQF scheduling.

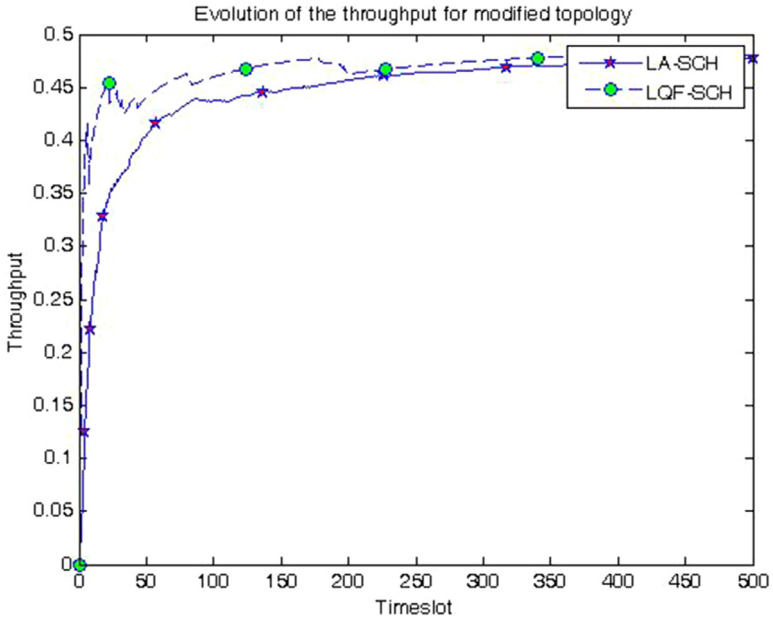


Fig. 13. Evolution of the throughput for modified topology.

as shown in Figs. 7 and 8 indicate the total number of packets and throughput for each flow tends to a constant value in steady state. The next result of our observation is that the throughputs of the flows converged to completely different values according to the separation of the source and destination. For example, the longest hop flow F2 achieves the lowest throughput versus F3 that reaches to the highest throughput with only 2.3 hops in average length.

In the second scenario, the performance of generic pick-and-compare scheduling algorithm and learning automata based scheme that gradually learn truly decision on neighbor selection were compared. The performance of our algorithm is illustrated in Fig. 9. It shows that the total backlog changes quickly and converges to the steady state value as compared to the generic randomized pick algorithm. Although the total backlog value in the steady state is significantly lower than the randomized pick scheme, suggesting that in comparison with randomized pick, greater number of packets to the destination is reached. This is because of blind random choosing of neighbors in classic pick algorithm. Fig. 10 shows that similar result was obtained when throughput was studied among two discussed schemes. It clearly shows that the throughput for the shortest and longest hop flows of the proposed method is significantly higher than the classical pick algorithm.

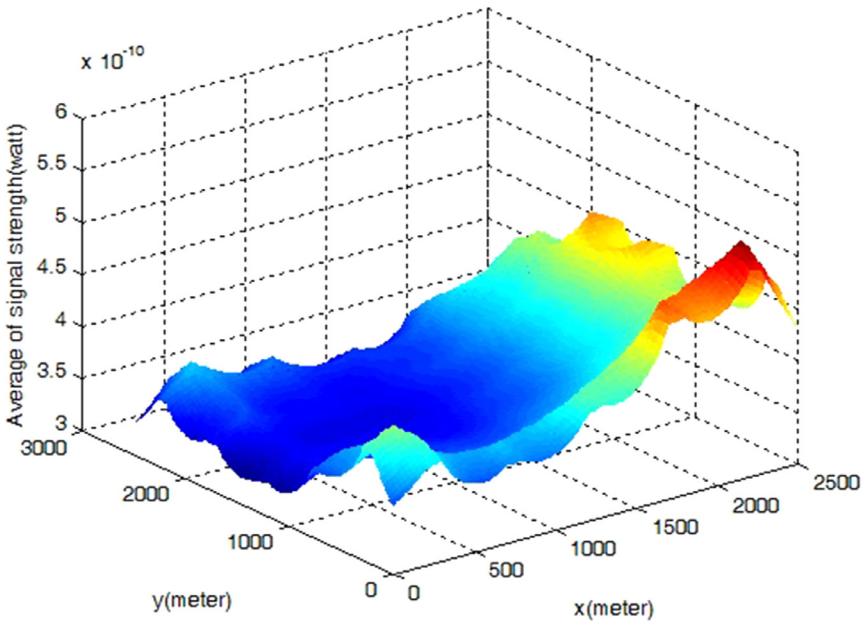


Fig. 14. Average of signal strength sensed in every point of network.

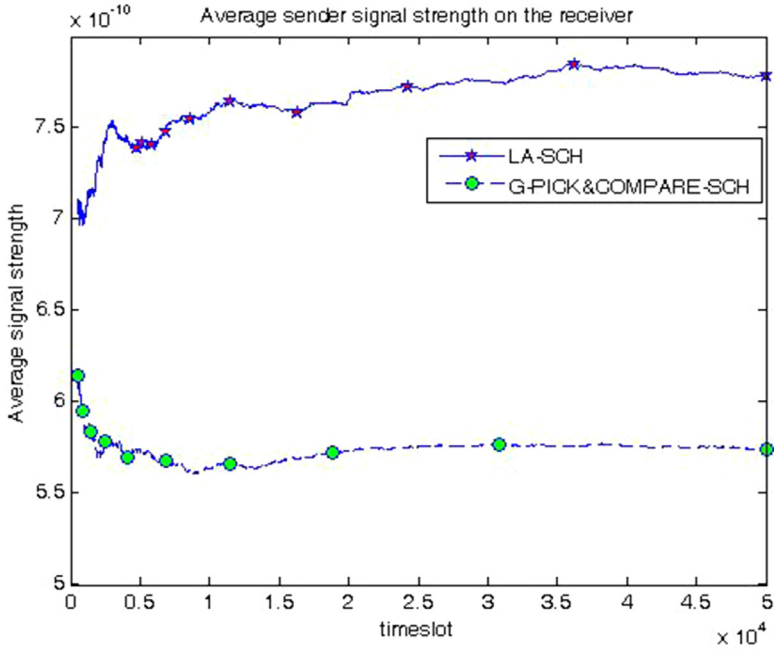


Fig. 15. Average of senders signal strength that sensed on the receiver nodes.

We also compare our algorithm with the LQF algorithm presented in [11]. We use the same topology presented in the mentioned study, consider a simple network with 12 links whose link lengths are all equal to $d = 80$ m as shown in Fig. 11 The links are placed in this network such that if we activate two links $2k+1$ and $2k+2$ for $0 \le k \le 5$ then the SINR of each link is equal to $\beta = 5$ (i.e., this means $d_0 = d\beta^{1/\alpha}$ where d_0 is shown in this figure). In this case, LQF find maximal schedule set with a size of 2 where our algorithm finds six schedulable links. Fig. 12 shows that LQF scheduling achieves smaller throughput than our scheme in this setting.

In the next case, we use modified network topology in Fig. 10 slightly by changing the distance d_0 in this figure to $d_0 - \varepsilon$ for some small $\varepsilon > 0$. With this small change, any two links $2k+1$ and $2k+2$ for $0 \le k \le 5$ do not form a feasible schedule anymore. Instead all maximal schedules in this modified network have two "large" maximal schedules: S1 activates 6 odd links (1, 3, 5, 7, 9, and 11) and S2 activates 6 even links (2, 4, 6, 8, 10, and 12). We show the performance of the two scheduling schemes again under this modified network topology in Fig. 13. The results show that the two scheduling schemes achieve the same throughput performance.

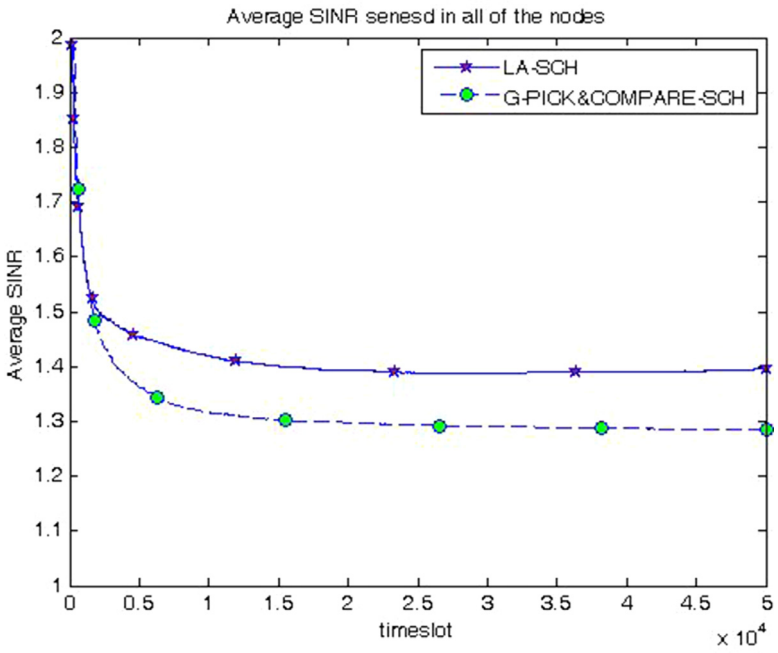


Fig. 16. Average of SINR sensed in all of the nodes.

This can be interpreted as follows. Since all maximal schedules include the same number of active links, they both achieve 100% throughput.

Finally, we investigate the evolution of the physical interference values during the execution of the algorithms. Fig. 14 shows the average of additive physical interference values which is sensed in every point of the network area. The result, as shown in the figure indicates physical interference values are very high near the receiving nodes. Next we study signal to interference plus noise ratio (SINR) comparatively in our learning automata based scheduling algorithms and generic pick and compare scheduling. Fig. 15 shows average senders signal strength that is sensed in receiving nodes is improved gradually over time-slot, whereas in LA based scheduling this amount reaches to 20% higher than generic pick and compare algorithm at the steady-state. Average of total SINR that is sensed in all of the nodes is the last result of our study which is shown in Fig. 16. In both LA based scheduling and generic pick and compare scheme, despite the increase of SINR in receiving nodes, more activated links across the time slot, will lead to more additive interference and results low SINR in intermediate nodes.

5. Conclusions

This paper presents a class of simple distributed scheduling algorithm in wireless network. Enhanced pick-and compare based scheduling algorithm have been developed for physical interference model in a wireless network that can achieve any fraction of throughput region. According to unknown characteristics of nodes contention, those equipped by learning automata can predict node participation in scheduling according to the environmental response of the networks for last behavior of network components. In compare algorithm, a more realistic collision and interference conflicting graph was presented in this paper based on physical model. Most of the throughput-optimal scheduling strategies lead to an NP-hard problem that should be solved in a centralized algorithm in every time slot. 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 was proposed.

References

[1] L. Tassiulas, A. Ephremides, Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks, IEEE Trans. Autom. Control 37 (12) (1992) 1936–1948.
[2] L. Tassiulas, Linear complexity algorithms for maximum throughput in radio networks and input queued switches, in: Proceedings of the Seventeenth Annual Joint Conference of the IEEE Computer and Communications Societies, INFOCOM '98, IEEE, 1998.
[3] L. Xiaojun, N.B. Shroff, The impact of imperfect scheduling on cross-Layer congestion control in wireless networks, IEEE/ACM Trans. Netw. 14 (2) (2006) 302–315.
[4] A. Eryilmaz, A. Ozdaglar, E. Modiano, Polynomial complexity algorithms for full utilization of multi-hop wireless networks, in: Proceedings of the 26th IEEE International Conference on Computer Communications, INFOCOM, 2007.
[5] Y. Yi, G. De Veciana, S. Shakkottai, On optimal MAC scheduling with physical interference, in: Proceedings of the 26th IEEE International Conference on Computer Communications. INFOCOM, IEEE, 2007.
[6] A. Behzad, I. Rubin, Optimum integrated link scheduling and power control for multihop wireless networks, IEEE Trans. Vehicular Technol. 56 (1) (2007) 194–205.
[7] R. Negi, A. Rajeswaran, Physical layer effect on mac performance in ad-hoc wireless networks, in: Proceedings of the Communication, Internet and Information Technology, 2003.
[8] Y. Zhou, et al., Throughput optimizing localized link scheduling for multihop wireless networks under physical interference model, IEEE Trans. Parallel Distrib. Syst. 25 (10) (2014) 2708–2720.
[9] P.-J. Wan, X. Xu, O. Frieder, Shortest link scheduling with power control under physical interference model, in: Proceedings of the 2010 Sixth International Conference on Mobile Ad-Hoc and Sensor Networks (MSN), IEEE, 2010.
[10] X. Xu, M. Song, Stable wireless link scheduling subject to physical interference with power control, in: Proceedings of the 23rd International Conference on Computer Communication and Networks (ICCCN), IEEE, 2014.
[11] L.B. Le, et al., Longest-queue-first scheduling under SINR interference model, in: Proceedings of the Eleventh ACM International Symposium on Mobile ad hoc Networking and Computing, ACM, 2010.
[12] A. Iyer, C. Rosenberg, A. Karnik, What is the right model for wireless channel interference? IEEE Trans. Wirel. Commun. 8 (5) (2009) 2662–2671.
[13] C. Joo, X. Lin, N.B. Shroff, Understanding the capacity region of the greedy maximal scheduling algorithm in multihop wireless networks, IEEE/ACM Trans.Netw. (TON) 17 (4) (2009) 1132–1145.
[14] P. Gupta, P.R. Kumar, The capacity of wireless networks, IEEE Trans. Inf. Theory 46 (2) (2000) 388–404.
[15] D. O'Neill, et al., Wireless NUM: Rate and reliability tradeoffs in random environment, in: Proceedings of the IEEE Wireless Communications and Networking Conference WCNC, IEEE, 2009.
[16] G. Foschini, J. Salz, Digital communications over fading radio channels, Bell Syst. Tech. J 62 (2) (1983) 429–456.
[17] M.A. Thathachar, P.S. Sastry, Networks of Learning Automata: Techniques for Online Stochastic Optimization, Springer, 2003.

[18] B. Ziaeddin, M. Mohammad Reza, Online channel assignment in multi-radio wireless mesh networks using learning automata, Int. J. Soft Comput. Softw. Eng. 3 (3) (2013) 726–732.
[19] T. Joshi, et al., SARA: stochastic automata rate adaptation for IEEE 802.11 networks, IEEE Trans. Parallel Distrib. Syst. 19 (11) (2008) 1579–1590.
[20] J.A. Torkestani, M.R. Meybodi, Mobility-based multicast routing algorithm for wireless mobile ad-hoc networks: A learning automata approach, Comput. Commun. 33 (6) (2010) 721–735.
[21] Y. Xing, R. Chandramouli, Stochastic learning solution for distributed discrete power control game in wireless data networks, IEEE/ACM Trans. Netw. 16 (4) (2008) 932–944.
[22] R. Kalman, J. Bertram, Control system analysis and design via the "second method" of Lyapunov: II—Discrete-time systems, J. Fluids Eng. 82 (2) (1960) 394–400.
[23] C. Ünsal, Intelligent Navigation of Autonomous Vehicles in an Automated Highway System: Learning Methods and Interacting Vehicles Approach, Virginia Polytechnic Institute and State University, 1997.
[24] H.K. Khalil, J. Grizzle, Nonlinear Systems, Vol. 3, Prentice hall, New Jersey, 1996.
[25] G. Brar, D.M. Blough, 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, ACM, 2006.
[26] K. Jain, et al., Impact of interference on multi-hop wireless network performance, Wirel. Netw. 11 (4) (2005) 471–487.
[27] B. Awerbuch, 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, ACM, 1987.
[28] R. Barr, Z.J. Haas, R. Van Renesse, Scalable wireless ad hoc network simulation, Handbook on Theoretical and Algorithmic Aspects of Sensor, Ad hoc Wireless, and Peer-to-Peer Networks, Auerbach Publications, 2005, pp. 297–311.

Ziaeddin Beheshtifard received his B.S. and M.S. degrees in Computer Engineering from the University of Tehran, Tehran, Iran in 1996 and Tarbiat Modares University, Tehran, Iran in 1999, respectively. He is currently pursuing his Ph.D. degree in Computer Engineering at the Azad University of Qazvin Branch, Qazvin, Iran. He joined the faculty of Computer Engineering Department at South Tehran Azad University, Tehran, Iran, in 2008. His research interests include wireless networks, ad hoc networks, soft computing, learning systems and distributed algorithms.

Mohammad Reza Meybodi received his B.S. and M.S. degrees in Economics from Shahid Beheshti University in Iran during 1973 and 1977, respectively. He also received his M.S. and Ph.D. degrees from Oklahoma University, USA during 1980 and 1983, respectively in Computer Science. Presently, he is a Full Professor in Computer Engineering Department, Amirkabir University of Technology, Tehran, Iran. Prior to the present position, he worked from 1983 to 1985 as an Assistant Professor at Western Michigan University, and from 1985 to 1991 as an Associate Professor at Ohio University, USA. His research interests include wireless networks, fault tolerant systems, learning systems, parallel algorithms, soft computing and software development.