An Energy-Efficient Learning Automata Based Coordination Topology Maintenance Algorithm for Prolonging Lifetime of Ad Hoc Wireless Networks #287

Shekufeh.Shafeie
Department of Technology and Computer Engineering, Young Researchers Club,
Islamic Azad University,Arak Branch
Arak, Iran
sh.shafeie@gmail.com

M. R. Meybodi
Department of Computer Engineering and Information Technology
Amirkabir University of Technology
Tehran, Iran
mmeybodi@aut.ac.ir

Abstract — **One of the basic restrictions in Ad Hoc Wireless Networks is energy supply and because of that proposing of power saving protocols that do the normal tasks of network without significantly diminishing the quality of services of the network and consequently, prolonging the lifetime of network has high importance. So, in this paper a distributed power saving technique for multi-hop ad hoc wireless networks based on learning automata has been proposed that all nodes in the network that are equipped with learning automata don't need to be synchronized with each other. Learning automata abilities such as low computational load, usability in distributed environments with ambiguous information, and adaptability to changes via low environmental feedbacks, causes to better fitness with local techniques in ad hoc wireless networks.**

The proposed protocol, SpanLA, consists of two phases; coordinator announcement and coordinator withdrawal. In SpanLA with a randomized algorithm the option of making local decisions on whether a node going to sleep or to join a forwarding backbone as coordinator is given to learning automata of each node. Unlike the basis protocol of this proposed protocol (span) each node after a random backoff delay uses its own learning automata and if it is needed to be a coordinator according to current conditions, selects the correspondent action and with the help of learning automata, the SpanLA's made decisions more than span protocol prevents from redundant nodes to be coordinator. Coordinator withdrawal is also runs locally, in a distributed manner and periodically after a node has been a coordinator for some period of time and whenever would be needed. To ensure fairness, SpanLA uses each node's allocated learning automata which are based on the status of nodes during the activity of the network.

So, using learning automata with passing of time causes to decreasing in energy consumption and improving network lifetime in SpanLA protocol with an 802.11 network in power saving mode in comparison to other similar protocols such as Span and without any topology management protocols. Simulation results with a practical energy model also show that above result.

Keywords- ***Adaptive protocol, Learning Automata, Topology maintenance, Ad hoc Wireless Networks, Energy Efficient, Scheduling.***

I. Introduction

Minimizing energy consumption is an important challenge in mobile networking. Significant progress has been made on low-power hardware design for mobile devices that the wireless network interface is often a device's single largest consumer of power. Since the network interface may often be idle, this power could be saved by turning the radio off when not in use [4]. In practice, however, this approach is not straightforward: a node must arrange to turn its radio on not just to receive packets addressed to it, but also to participate in any higher-level routing and control protocols. The requirement of cooperation between power saving and routing protocols is particularly acute in the case of multi-hop ad hoc wireless networks, when nodes must forward packets for each other. Coordination of power saving with routing in ad hoc wireless networks with the help of learning automata is the subject of this paper.

A good power-saving coordination technique for wireless ad-hoc networks ought to have the following characteristics. It should allow as many nodes as possible to turn their radio receivers off most of the time, since even an idle receive circuit can consume almost as much energy as an active transmitter. On the other hand, it should forward packets between any source and destination with minimally more delay than if all nodes were awake.

Investigations on Learning Automata (L.A) and Ad Hoc Wireless networks characteristics have shown that Learning Automata, with consideration to properties such as low computing overload, the ability of using in distributed environments with ambiguous information, and the adaptability with environmental changes, is a very suitable model for using in sensor networks [2].

The algorithm presented in this paper, SpanLA, is fully distributed and in it the nodes do not need to be fully synchronized with each other and fulfill the above requirements. Each node in the network running SpanLA makes periodic, local decisions with the aid of its allocated learning automata, on whether to sleep or stay awake as coordinator and participate in the forwarding backbone topology. Selected actions of non-coordinator nodes' learning automata and dedicating reward or penalty to them based on environmental conditions cause to preserving capacity. The proposed algorithm consists of two phases: coordinator announcement and coordinator withdrawal. To keep the number of redundant coordinators low and to rotate being coordinator role amongst all nodes, each node delays coordinator announcement and also coordinator withdrawal with a random delay that takes two factors into account: the amount of remaining battery energy, and the number of pairs of neighbors it can connect together. This combination along with taken decisions of learning automata's nodes ensure with high probability, a capacity preserving connected backbone at any point in time, where nodes tend to consume energy at about the same rate. SpanLA does all these using only local information with learning that learning that with passing of time, cause to better nodes would be selected as coordinators, consequently scaling well with the number of nodes. Our simulation results show that system lifetime with SpanLA for a range of node densities, is more than span with better forwarding capacity than to it and without span, for a range of node densities, without much reduction in overall forwarding capacity.

The rest of the paper is organized as follows. In section 2 learning automata is briefly reviewed. Section 3 gives an overview on some power-saving proposed methods in ad hoc networks. The proposed learning automata based protocol is described in section 4. Section 5 presents performance evaluation. Finally, Section 6 concludes.

II. Learning Automata

Learning automata (L.A) is an abstract model [7,8,9,16] which randomly selects one action out of its finite set of actions and performs it on a random environment. Environment then evaluates the selected action and responses to the automata with a reinforcement signal. Based on selected action, and received signal, the automata updates its internal state and selects its next action. Figure 1 depicts the relationship between an automata and its environment.

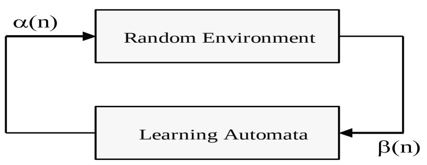

Figure 1. Relationship between learning automata and its environment.

Environment can be defined by the triple $E=\{\alpha,\beta,c\}$ where $\alpha=\{\alpha_1,\alpha_2,\ldots,\alpha_r\}$ represents a finite input set, $\beta=\{\beta_1,\beta_2,\ldots,\beta_r\}$ represents the output set, and $c=\{c_1,c_2,\ldots,c_r\}$ is a set of penalty probabilities, where each element c_i of c corresponds to one input of action α_i. An environment in which β can take only binary values 0 or 1 is referred to as P-model environment. A further generalization of the environment allows finite output sets with more than two elements that take values in the interval [0, 1]. Such an environment is referred to as Q-model. Finally, when the output of the environment is a continuous random variable which assumes values in the interval [0, 1]; it is referred to as an S-model. Learning automata are classified into fixed-structure stochastic, and variable-structure stochastic. In the following, we consider only variable-structure automata.

A variable-structure automaton is defined by the quadruple $\{\alpha,\beta,p,T\}$ in which $\alpha=\{\alpha_1,\alpha_2,\cdots,\alpha_r\}$ represents the action set of the automata, $\beta=\{\beta_1,\beta_2,\cdots,\beta_r\}$ represents the input set, $p=\{p_1,p_2,\cdots,p_r\}$ represents the action probability set, and finally $p(n+1)=T[\alpha(n),\beta(n),p(n)]$ represents the learning algorithm. This automaton operates as follows. Based on the action probability set p, automaton randomly selects an action α_i, and performs it on the environment. After receiving the environment's reinforcement signal, automaton updates its action probability set based on equations (1) for favorable responses, and equations (2) for unfavorable ones.

$$\begin{aligned} &p_i(n+1)=p_i(n)+a.(1-p_i(n)) \\ &p_j(n+1)=p_j(n)-a.p_j(n) \qquad \forall j \quad j\neq i \end{aligned} \tag{1}$$

$$\begin{aligned} &p_i(n+1)=(1-b).p_i(n) \\ &p_j(n+1)=\frac{b}{r-1}+(1-b)p_j(n) \qquad \forall j \quad j\neq i \end{aligned} \tag{2}$$

III. RELATED WORK

Two of proposed protocols that has similar goals to those of our proposed protocol, SpanLA and its base protocol, Span [4], are GAF [5,15] and CEC [18,19]. In GAF schema of Xu et al. that in it, nodes use geographic location information to divide the world into fixed square grids. The size of each grid stays constant, regardless of node density. Nodes within a grid switch between sleeping and listening, with the guarantee that one node in each grid stays up to route packets. Span also builds on its observation that when a region of a shared-channel wireless network has a sufficient density of nodes, only a small number of them need be on at any time to forward traffic for active connections and is a distributed, randomized algorithm where nodes make local decisions on whether to sleep, or to join a forwarding backbone as a coordinator. Each node bases its decision on an estimate of how many of its neighbors will benefit from it being awake, and the amount of energy available to it. A randomized algorithm is given in it where coordinators rotate with time, demonstrating how localized node decisions lead to a connected, capacity-preserving global topology. Span differs from GAF in two areas. First, unlike GAF, Span does not require that nodes know their geographic positions. Instead, Span uses broadcast messages to discover and react to changes in the network topology. Second, Span integrates with 802.11 power saving mode nicely: non-coordinator nodes can still receive packets when operating in power saving mode. In spite of the fact that, Span rotates the role of being coordinator between all nodes in the network; but with adding learning method better decisions for future assigned role of nodes in the network would be made, that causes to energy efficiency without considerable diminishing the quality service of the network.

IV. PROPOSED PROTOCOL (SPANLA)

SpanLA with the help of learning elects “coordinators” adaptively from all nodes in the network. SpanLA coordinators stay awake continuously and perform multi-hop packet routing within the ad hoc network, while other nodes are remained in power-saving mode and periodically check if they should wake up and become a coordinator.

SpanLA rotates the decision for being coordinator to ensure that all nodes are given the same chance for providing global connectivity roughly equally. SpanLA is proactive: each node periodically broadcasts HELLO messages that contain the node’s status (the selected action of its allocated L.A), its current coordinators, and its current neighbors. The HELLO messages, and consequently the states maintained by SpanLA, are similar to its base protocol Span [4] and also to those of proactive ad hoc routing protocols (e.g., geographic routing [10,11,12] or DSDV[17]). Each node maintains only a small amount of additional state—its coordinators and coordinators of neighbors—in addition to a list of neighbors normally found in the routing table.

As shown in Figure 2, SpanLA like its base protocol Span [4] runs above the link and MAC layers and interacts with the routing protocol. This structuring allows SpanLA to take advantage of power-saving features of the link layer protocol, while still being able to affect the routing process. For example, non-coordinator nodes can periodically turn on their radios and listen (e.g. 802.11 power-saving mode [1]) for their packets. SpanLA leverages a feature of modern power-saving MAC layers, where if a node has been asleep for a while, packets destined for it are not lost but are buffered at an upstream neighbor. When the node awakens, it can retrieve these packets from the buffering node, typically a coordinator. SpanLA requires a modification to the route lookup process at each node-at any time, only those entries in a node’s routing table that correspond to currently active coordinators can be used as valid next-hops (unless the next hop is destination itself).

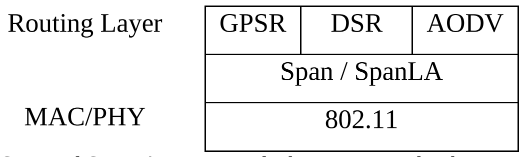

Figure 2. Span and SpanLA are protocols that operate under the routing layer and above the MAC and physical layers. The routing layer uses the information which is provided by Span and SpanLA. Span and SpanLA leverage any power saving features of underlying MAC layer.

A SpanLA node switches state from time to time between being a coordinator and being a non-coordinator based on its learning automata decision. A node includes its current state in its HELLO messages. SpanLA consists of two phases: coordinator announcement and coordinator withdrawal which with the help of learning try to make better fairness between nodes from point of energy consumption in comparison with similar protocols especially its base protocol, Span. The following subsections describe these two phases and influence of learning on its performance.

Table 0. Hello packet for SpanLA protocol.

Source ID
Node position
Is coordinator
Is tentative
Coordinator
Neighbor list

A. Coordinator Announcement

At the first, and because the source and destination nodes of each flow need to constantly send and receive packets, they do not operate in power saving mode. Thus, they automatically become coordinator. After that, learning algorithm performs for the purpose of selecting coordinators between all remaining non-coordinator nodes. In this proposed protocol, each node in the network is equipped with one learning automata and has two actions which are as follows: selecting as a coordinator (α1), and selecting as a non-coordinator (α2). Initially, supposed that, the selection probability of both these actions is equal to each other. In other words, is according to the following relation (3) where, *m* is the number of learning automata actions which is two here:

$$\forall i \quad i \leq m \quad P_i = \frac{1}{m} \qquad (3)$$

α1: (being Coordinator) = 0
α2: (being Non-Coordinator) = 1

Selecting coordinator algorithm triggers locally, and whenever requiring to selection of coordinators amongst middle nodes would be needed. Each non-coordinator node at its first stage of learning randomly selects one of its learning automaton's actions and then based on that selection and according to the required parameters either reward or penalty is given to that selected action as follows:

Each node periodically broadcasts Hello messages that contain the node's status (i.e., whether or not the node is a coordinator; which is the same as selected action of node's allocated learning automaton), and also its current coordinators, and its current neighbors. Based on these Hello messages, each node makes a list of its neighbors and its coordinators, and for each of its neighbors also does the same as for itself. These Hello messages, and consequently the states maintained by SpanLA like Span, are similar to those of proactive ad hoc routing protocols (e.g., geographic routing [10,11,12] or DSDV [17]). Each node maintains only a small amount of additional state—its coordinators and coordinators of neighbors— in addition to a list of neighbors normally found in the routing table.

After that, each node forms reinforcement signal (B_i) based on the number of additional pairs of nodes among its neighbors that would be connected if this node were to become a coordinator to forward packets, which is called check-announce function and its output is called C.

Details of forming the reinforcement signal (B_i) is as follows:

If node's learning automaton has selected to be a coordinator, then if return value of check-announce function or in other words C becomes larger than 0, it means that the node's learning automaton action is proper and being coordinator of this node causes to connect non-iterative additional pairs of nodes which were disconnected in this stage and were required to be coordinator. So, this action would be awarded reward; else penalty would be given to it. Then action probabilities change according to relation 1 or 2 depends on having reward or penalty.

If node's learning automaton has selected to be non-coordinator, then if return value of check-announce function or in other words C becomes lower than or equal to 0, it means that node's learning automaton has selected a proper action, since this node was a redundant node and reward should be dedicated to this action of node's automaton; else if check-announce function returns a value higher than 0, then selected action would be penalized.

$$B_{i=} \begin{cases} 0: & \text{if } (\alpha{=}0) \text{ and } (c>0) \text{ or if } (\alpha{=}1) \text{ and } (c<=0) \\ & \text{Favorable Response (Reward)} \\ 1: & \text{if } (\alpha{=}0) \text{ and } (c<=0) \text{ or if } (\alpha{=}1) \text{ and } (c>0) \end{cases}$$

: Unfavorable Response (Penalty)

Also, if a non-coordinator node has received a large number of packets after 5 times to route in the recent past, besides announcing itself as a coordinator, the probability of being coordinator action in this node would be increased with dedicating reward to it; since the large number of received packets remind that node is located in a populated radio range that sleeping of it causes to connectedness and high loss rate.

In network lifetime and for making better fairness between nodes in terms of energy consumption, a non-coordinator node periodically, determines if it should become a coordinator or not, and for making decision applies learning. Selection algorithm based on learning automata locally and whenever selection of coordinators amongst middle nodes for preserving connectivity is required, would be performed.

Announcement contention for being coordinator may occur too, where multiple nodes discover the lack of a coordinator at the same time, and all decide to become a coordinator and perform coordinator announcement phase. Contention would be resolved by delaying coordinator announcement with a randomized back off delay as follows.

Selection coordinator algorithm of a node i, before entering into learning phase for selecting one of learning automata's actions with calculating the number of additional pairs of nodes among the neighbors that would be connected if i were to become a coordinator or the same connect pair (C_i), and also the number of neighbors for node i or (N_i), calculates back off delay according to equation (4), somehow before entering into Selection coordinator phase waits for this achieved delay time. After passing of this delay period of time, node applies its allocated learning automata for making decision of being coordinator.

$$delay = \left(\left(1 - \frac{E_r}{E_m}\right) + \left(1 - \frac{C_i}{\binom{N_i}{2}}\right) + R \right) \times N_i \times T \qquad (4)$$

In the above equation (4) for calculating delay, N_i is the number of neighbors for node i and C_i is the number of additional pairs of nodes among the neighbors that would be connected if i were to become a coordinator and would forward packets. Clearly, 0<= Ci <= com (Ni ,2) and $\frac{C_i}{\binom{N_i}{2}}$ is the *utility* of node i. $\left(1 - \frac{C_i}{\binom{N_i}{2}}\right)$ part in the above equation is called U and its value is assigned based on the following conditions:

If ((Ni =1) or (Ci<0)) then U=1

If ((Ni< >1) or (Ci >0)) then U= $\left(1 - \frac{C_i}{\binom{N_i}{2}}\right)$

And finally,
if (U < 0) then U = 0
if (U > 1) then U = 1

T is a round-trip delay for a small packet over the wireless link.

R uniformly at random is achieved from the interval [0, 1]. E_r denotes the amount of energy (in Joules) that still remains at a node, and E_m is the maximum amount of energy available at the same node.

At the end of the delay time, node applies its selection algorithm based on learning automata for determining if becomes a coordinator or not required to be selected as a coordinator; and based on recent received Hello messages and assumed previous parameters such as before mentioned check-announce function, evaluates its learning automaton selected actions with the aid of automaton's environment which is the same as surroundings nodes in the network and gives reward or penalty to them, and finally if it is necessary announces its being coordinator with sending Hello messages.

Learning algorithm in selecting coordinator phase with the help of learning automata takes along until the end of network lifetime, and whenever a non coordinator node becomes active locally for making decision, will be applied and causes to the fact that a node always makes the best decision in this phase, in regarding to its neighbors. In other words, a node selects the action with higher probability and through this way with conducting lower efficient nodes to sleep (from points of energy and the number of pair connector nodes), optimally causes to decrease energy consumption.

B. *Coordinator Withdrawal*

Each coordinator periodically checks if it should withdraw as coordinator or not. For making true decision for this purpose which is withdrawal of being coordinator, the same allocated learning automaton to each node in the phase of selecting coordinator would be applied which has two before actions, and based on the achieved probability for each of its actions during the network activity that has been done till now, the best action would be selected, and be compared with required conditions for topology maintenance, and then reinforcement signals would be formed. Selected action is compared with return value of a function which is called check_withdraw, which its task is to controll if every pair of neighbor nodes can reach each other via some other neighbors, even if those neighbors are not currently coordinators. If such pairs would be found for a node, function returns True; else it returns False.

Generally, forming reinforcement signal (B_i) and giving reward or penalty to selected actions of one node's allocated automaton in this phase, is as follows:

Initially, one node selects its best action among available actions, and then its status would be investigated from points of energy, being active and alive and also its return value of *check_withdraw* function.

If node be inactive or return value of its *check_withdraw* function would be False, then if selected action of the node would be Non-Coordinator, this action would be awarded reward with reward coefficient (a) according to relation (1); but if selected action of the node would be Coordinator, this action would be penalized with a penalty coefficient (b) according to relation (2).

Also, for rotating the coordinators among all nodes fairly, a *check_withdraw* function is defined that runs after a node has been a coordinator for some period of time, and has two return values which are True and False. Its description is as follows: if every pair of one node neighbors can reach each other via one or two other neighbors, even if those neighbors are not currently coordinators, it marks itself as a tentative coordinator in order to give its neighbors a chance to become coordinators and returns True. A tentative coordinator can still be used to forward packets. A coordinator stays tentative for W_T amount of time, where W_T is the maximum value in equation (4) that is equal to $W_T = 3*Ni*T$. also here again reinforcement signal (B_i) is formed after a node has been a coordinator for some period of time based on *check_withdraw* function output. After that reward or penalty is given to learning automata's selected actions. However, the coordinator announcement algorithm treats a tentative coordinator as a non-coordinator.

For situations in which nodes are active, alive and have energy:

If *check_withdraw* function returns True then:

1- If selected action of a node would be being non-coordinator, its learning automaton action would be awarded reward with coefficient a_{low}, as *check_withdraw* function also marks this node as a redundant node according to relation (1).

2- If selected action of a node would be being coordinator, its learning automaton action would be penalize with b_{low} coefficient according to relation (2), because *check_withdraw* function expresses that pair of neighbor nodes can connect to each other via one or other more nodes too.

If a coordinator has not withdrawn after W_T, it clears its tentative bit. To prevent an unlucky low energy node from draining all of its energy once it becomes a coordinator, the amount of time a node stays as a coordinator before turning on its tentative bit is proportional to the amount of energy it has(E_r).

But, if *check_withdraw* function returns False, it means that there is not any other node for connecting pair of neighbor nodes to each other excepting the node which is currently under evaluation; so:

1- If selected action of a node would be being non-coordinator, its learning automaton action would be penalize with b_{low} coefficient, because *check_withdraw* function also marks this node as a redundant node that it is not necessary to stay alive this time.

2- If selected action of a node would be being coordinator, its learning automaton action would be awarded reward with a_{low} coefficient, because *check_withdraw* function also shows that this node is not a redundant node and must stay as a coordinator.

a_{low}, b_{low} coefficient are supposed to be constant in experiments and orderly equal to 0.3 and 0.1. The constituted reinforcement signal (B_i) by the environment for learning automata is defined as follows:

$$B_i = \begin{cases} 0\text{: if (!active() and(check\_withdraw(false))) and if }(\alpha=1) \text{ : }\textbf{Favorable Response (Reward)} \\ 1\text{: if (!active() and(check\_withdraw(false))) and if }(\alpha=0) \text{ : }\textbf{Unfavorable Response (Penalty)} \\ 1\text{: if (active()) and(check\_withdraw(true)) and if }(\alpha=0)\text{ ، redundant node: }\textbf{Unfavorable Response (Penalty)} \\ 0\text{: if (active()) and(check\_withdraw(true)) and if }(\alpha=1)\text{ ، redundant node: }\textbf{Favorable Response (Reward)} \\ 1\text{: if (active()) and(check\_withdraw(false)) and if }(\alpha=1)\text{ ، non-redundant node: }\textbf{Unfavorable Response (Penalty)} \\ 0\text{: if (active()) and(check\_withdraw(false)) and if }(\alpha=0)\text{، non-redundant node: }\textbf{Favorable Response (Reward)} \end{cases}$$

V. EXPERIMENTAL RESULTS

The proposed protocol, SpanLA, has been implemented in the ns-2 network simulator environment; the same simulator like its base protocol, Span, and uses geographic forwarding algorithm, the 802.11 power saving mode (with the improvements have been made in [4]), and the energy model that has been used for Span protocol too.

As mentioned in the Span protocol implementation too; although, the geographic forwarding algorithm has been chosen because of its simplicity; SpanLA as the same like its base protocol, can be used with other routing protocols as well. SpanLA's Coordinator announcement algorithm requires each node to advertise its coordinators, its neighbors, and whether its allocated learning automaton has selected to be a coordinator or not. To reduce protocol overhead, SpanLA Hello information has been piggybacked onto the broadcast updates which are requiring by geographic forwarding. See Table 0.

SpanLA determines when to turn a node's radio on or off with the help of its learning allocated algorithm, but depends on the low level MAC layer to support power saving functions, such as buffering packets for sleeping nodes. SpanLA has been implemented on top of the 802.11 MAC and physical layers with ad hoc power saving support [1]. To measure the effective of SpanLA, we simulated it with geographic forwarding, on several static and mobile topologies. Simulation results show that SpanLA not only performs well by extending network lifetime, even in comparison with Span protocol; it also out-performs unmodified 802.11 power saving network in handling heavy load, per-packet delivery latency, and network lifetime like Span.

We simulated SpanLA with other comparable protocols such as Span in the ns-2[14] network simulator using the CMU wireless extensions [13]. Geographic forwarding algorithm, as described in [4], route packets from source to destination. SpanLA like Span runs on top of 802.11 MAC layer with power saving support and modifications described in [4].

We have compared the performance of SpanLA against Span which is completely specified in [4] and above mentioned descriptions, both unmodified 802.11 MAC in power saving mode and unmodified 802.11 MAC not in power saving mode from points of consumed energy, capacity preservation and network lifetime. For convenience, we will refer to them as Span, 802.11 PSM, and 802.11.

To evaluate SpanLA with other comparable protocols such as Span in different node densities, we have simulated 120-node networks in square region of different sizes. Nodes in our simulation use radios with a 2 Mbps bandwidth and 250 meters nominal radio range. Twenty nodes send and receive traffic. Each of these nodes send a CBR flow to another node, and each CBR flow sends 128 byte packets. Each sender sends three packets per second, for a total of 60 Kbps of traffic.

To ensure that the packets of each CBR flow go through multi hops before reaching the destination node, 10 source and destination nodes are placed, uniformly at random, on each of two 50 meter-wide, full-height strips located at the left and right of the simulated region. A source must send packets to a destination node on the other strip. The initial positions of the remaining 100 nodes are chosen uniformly at random in the entire simulated region. Thus, the square root of the simulated region and the number of hops needed by each packet are approximately proportional.

Source and destination nodes never move. In mobile experiments, the motion of the remaining 100 nodes follows the *random waypoint* model [3]: initially, each node chooses a destination uniformly at random in the simulated region, choose a speed uniformly at random between 0 and 20 m/s, and moves there with the chosen speed. The node then pauses for an adjustable period of time before repeating the same process. The degree of mobility is reflected in the pause time. By default, we used a pause time of 60 seconds.

For simplicity, we did not use a location service in our simulations. Instead, a router obtains the location of destination node from the God module in ns. Since the location lookup is only required once per flow at the sender, we believe the overhead produced by the location service is not likely to change our results. Nevertheless, location services such as GLS [11] can be used with SpanLA.

The energy model used for simulation of SpanLA protocol is the same as its base protocol, Span, that has been completely specified in [4] and has been summarized the time-averaged results in Table 1 with Cabletron Roamabout 802.11 DS High Rate network interface card (NIC) operating at 2 Mbps in base station mode for different states of a node such as transmitting, receiving, idle and sleeping mode, and has been noted in [4] that these closely match the results obtained by Feeney and Nilsson [6] for similar 802.11 network interface cards in the ad hoc mode. "Rx" state measurement was obtained by putting the card into non-power saving mode, and measuring the power required to listen for a packet, decode it, and pass its contents up to the host. The "idle" state measurement was obtained in the same manner, but measuring only the power required to listen for a packet. In contrast, the "sleep" state measurement was obtained by putting the card into power saving mode, and measuring the average (lower, and near constant) power consumption during the part of the power saving cycle where the card was not listening for packets. The key point to note is the large difference between the power consumption of idle and sleeping modes.

Table 1: Power consumption of the Cabletron 802.11 network card in the "TX" (transmit), "RX" (receive), "Idle" and "Sleeping" modes.

Tx	Rx	Idle	Sleeping
1400 mW	1000 mW	830mW	130mW

Reward and penalty parameters (a,b) for allocated learning automata to each node in selecting coordinator phase is assumed constant and orderly equal to 0.1, 0.01. Also reward and penalty parameters for withdrawing coordinator phase are set to 0.3 and 0.1. All experimental results are average of five runs on different randomly-chosen scenarios. We define node density (as used in our graph axis labels) as the number of nodes that are not source or destinations per radio range, an area of $250^2 * \pi$ square meters.

A. *Experiment 1 Energy Consumption*

The goal of this experiment is considering the ability of proposed protocol SpanLA which is based on improved 802.11 power saving mode from point of energy conservation or energy consumption with other comparable protocols such as Span which is completely specified in [4], both unmodified 802.11 MAC in power saving mode and unmodified 802.11 MAC not in power saving mode. For convenience, we will refer to them as Span, 802.11 PSM, and 802.11. The potential for saving depends on node density, since the fraction of sleeping nodes depends on the number of nodes per radio coverage area. The energy savings also depends on a radio's power consumption in sleep mode and the amount of time that sleeping nodes must turn on their receivers to listen for 802.11 beacons and Span or SpanLA HELLO messages.

Figure 3 shows the fraction of energy remaining at each node after 300 seconds of simulation. Each node has an initial energy of 300 J. As it shows; Span and SpanLA provide a considerable amount of energy savings over 802.11, while 802.11 PSM does not provide any energy saving at all. This is because geographic forwarding, Span and SpanLA need to send broadcast messages, while with 802.11 PSM, each time a node receives a broadcast advertisement, it must stay up for the entire beacon period. This prevents non-coordinators from going back to sleep. When the node density is low, the number of broadcast messages in a radio range decreases, and 802.11 PSM yields a small amount of energy savings. As density increases, a smaller fraction of the nodes are elected coordinators. Consequently, expect that energy savings to increase. In practice, however, energy savings do not increase as much; and also performed calculations in [4] certified above results and expressed that gain in energy savings is a sub-linear function of node density.

Of course, SpanLA protocol until the time 300 second of simulation; which is approximately initial phase of network lifetime, saves less energy than its base protocol, Span. Because of learning algorithm does not progress so much until this time and does not achieve certainty and optimum decision, but with passing of more time from network lifetime and with the help of learning automata it takes better decisions, somehow compensates consumed energy of initial phase of network lifetime and achieves to more energy savings than Span; which causes to prolonging network lifetime in future and shows the effect of applying learning algorithm.

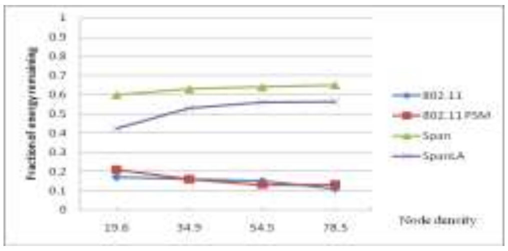

Figure 3.Fraction of energy remaining after 300 seconds of simulation for proposed protocol SpanLA, Span, 802.11 PSM and802.11. Span and SpanLA provides significant amount of savings over 8011 PSM and 802.11.

Figure 4 which compares, energy remaining for two protocols Span and SpanLA after 300, 600, 850 seconds of simulation in a 1000 meter * 1000 meter area, where each radio has an isotropic circular range with a 250 meter radius or the same density equal to 19.6, expresses above results too.

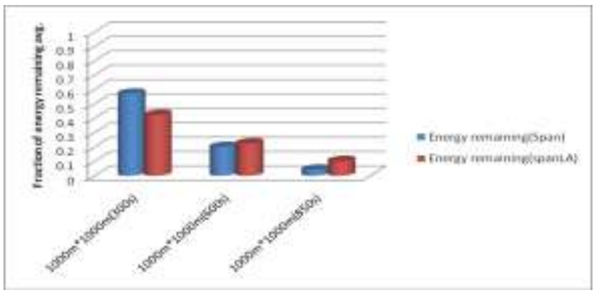

Figure 4.Comparison fraction of energy remaining for proposed protocol SpanLA, Span after 300,600,850 seconds of simulation.

B. *Experiment 2 Network Lifetime*

The goal of this experiment is investigation of network lifetime with proposed protocol and effect of using learning automata on this important parameter in ad-hoc sensor networks and comparison the achieved lifetime of appliance proposed protocol against other depicted protocols in [4] such as Span protocol. Achieved results are from a single mobile experiment on a 500m*500m area. The 20 source and destination nodes start with 2000 Joules of energy, and the remaining 100 forwarding nodes start with 300 Joules of energy. As before said; these simulations involve mobility.

Figures 5 and 6 show the achieved network lifetime from apply of SpanLA protocol as a management topology protocol against other comparable protocols such as Span, and also without any definite topology management protocol and just with 802.11 PSM and 802.11.

Two definitions are assumed for network lifetime based on [4]. First, as the time it takes for aggregate delivery rate to drop below 90%. Second, based on the number of forwarding nodes which are still alive. As figure 5 depicts; without Span, and SpanLA, nodes critical to multi-hop routing die around the same time, 335 seconds into the simulation. But, with Span the first node failure occurs 615 seconds into the simulation. The packet delivery rate does not drop below 90% until 851 seconds into the simulation, when 12 out of the 100 forwarding nodes are still alive. As nodes start to die, the node density decreases. Consequently, the drain rate also increases. Span protocol based on both above definitions for lifetime; significantly provides longer lifetime in comparison with not using topology management protocol modes like just with 802.11 PSM and 802.11.

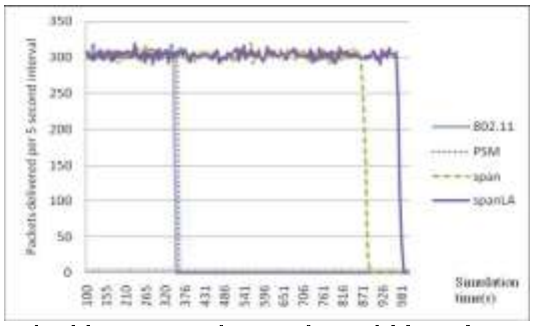

Figure 5.Packet delivery rate as a function of network lifetime for proposed protocol SpanLA, Span, 802.11 PSM and 802.11.

With applying of SpanLA protocol; the packet delivery rate does not drop below 90% until 975 seconds into the simulation, when 30 out of the 100 forwarding nodes are still alive. As both figures 5 and 6 show SpanLA protocol could prolong network lifetime based on two above definitions for lifetime; in comparison to other modes such as applying Span protocol.

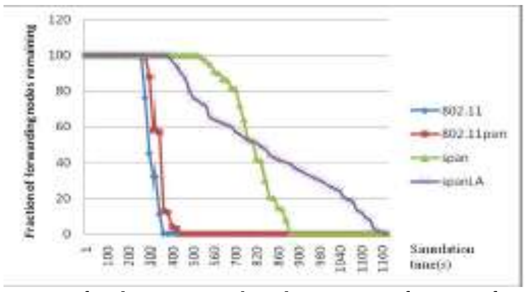

Figure 6.Fraction of nodes remaining based on time as a function of network lifetime for proposed protocol SpanLA, Span, 802.11 PSM and 802.11.

VI. CONCLUSION

In this paper, SpanLA protocol with an 802.11 network in power saving mode, which is based on learning automata has been introduced, that with passing of time causes to decrease in energy consumption and improve the system lifetime. Simulation results with a practical energy model also show that the above result for SpanLA protocol in comparison to other similar protocols, such as Span, and without any topology management protocols; and what is important and can be proved is that this result is such an achievable one with improvements in capacity and connectivity and communication latency.

ACKNOWLEDGMENT

We thank Benjie Chen for making the SPAN simulation code available on the web and M.Esnaashari at soft computing laboratory of Amirkabir University for his valuable suggestions.

REFERENCES

[1] *Wireless LAN Medium Access Control and Physical Layer Specifications*, Aug.1999. IEEE 802.11 Standards (IEEE Computer Society LAN MAN Standards Committee).

[2] M. Haleem and R. Chandramouli, "Adaptive downlink scheduling and rate selection: a cross layer design", *Special issue on Mobile Computing and Networking, IEEE Journal on Selected Areas in Communications,* vol. 23, no.6, June 2005.

[3] Broch, J., Maltz, D.Johnson,D., Hu,Y., And Jetcheva, J. "*A Performance Comparison of Multi-hop Wireless Ad Hoc Networki Routing Protocols."* In Proceeding of the fifth Annual ACM/IEEE International Conference on Mobile Computing and Networking(MobiCom)(Dallas,TX,Aug. 1998).

[4] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris, "*Span: An Energy-Efficient Coordination Algorithm for Topology Maintenance in Ad Hoc Wireless Networks*," In Proceedings of ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom 2001), Rome, Italy, July 16-21,2001, pages 70–84.

[5] Xu Y., Heidemann J. S. and Estrin D., *"Geography-informed energy conservation for ad hoc routing"*, in: Proceedings of the Seventh Annual International Conference on Mobile Computing and Networking (MobiCom), pp. 70-84, 2001.

[6] Feeney,L. AND Nilsson, M. "*Investigating the Energy Consumption of a Wireless Network Interface in Ad Hoc Networking Environment*." In Proceedings of IEEE INFOCOM(Anchorage, AK,2001).

[7] Narendra K. S. and Thathachar M. A. L, *"Learning Automata: An Introduction"*, Englewood Cliffs, NJ: Prentice Hall, 1989.

[8] Beigy H. and Meybodi M. R., *"A Mathematical Framework for Cellular Learning Automata"*, Advances on Complex Systems, Vol. 7, No. 3, pp. 1-25, 2004.

[9] Thathachar M. A. L. and Sastry P. S., *"Varieties of Learning Automata: An Overview"*, IEEE Transaction on Systems, Man, and Cybernetics-Part B: Cybernetics, Vol. 32, No. 6, PP. 711-722, 2002.

[10] FINN, G.G. , "*Routing and Addressing problems in Large Metropolitan-Scale Internetworks*.", ISI/R-R-57-180, ISI, Mar.1987.

[11] Li, J., Jannotti, J., Couto, D.D. , Karger, D., And Morris, R. *"A Scalable Location Service for Geographic Ad-Hoc Routing.",* In Proceedings of the Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking(Mobicom)(Aug. 2000).

[12] KARP, B., AND KUNG, H.T. GPSR: "*Greedy Perimeter Stateless Routing for Wireless Networks."*, In Proceedings of the Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking(Mobicom)(Aug. 2000).

[13] CMU Monarch Extensions to ns. *http://www.monarch.cs.cmu.edu/*.

[14] The Network Simulator - ns-2. *http://www.isi.edu/nsnam/ns/*.

[15] Xu Y., Heidemann J. S. and Estrin D., *"Geography-informed energy conservation for ad hoc routing"*, in: Proceedings of the Seventh Annual International Conference on Mobile Computing and Networking (MobiCom), pp. 70-84, 2001.

[16] K. Najim and A. S. Poznyak, "*Learning Automata: Theory and Application*", Tarrytown, NY: Elsevier Science Ltd., 1994.

[17] PERKINS, C.,"Highly Dynamic Destination –Sequenced Distanced Vector Routing (DSDV) for Mobile Computers.", in proceedings of the ACM SIGCOMM (London, U.K, 1994).

[18] Jie Wu, Handbook on Theoretical and Algorithmic Aspects of Sensor," Ad Hoc Wireless,and Peer-to-Peer Networks", © 2006 by Taylor & Francis Group, LLC

[19] Xu Y, "Adaptive Energy Conservation Protocols forWireless Ad Hoc Routing", Ph.D. thesis, University of Sourthern California (USC), 2002.