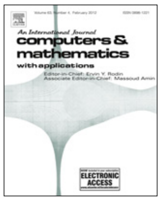
Computers and Mathematics with Applications ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at SciVerse ScienceDirect

Computers and Mathematics with Applications

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



Sleep-based topology control in the Ad Hoc networks by using fitness aware learning automata

Mina Shirali[a,*], Nasrin Shirali[b], Mohamad Reza Meybodi[c]

[a] Islamic Azad University, Zarandieh Branch, Department of Computer, Markazi, Iran
[b] Islamic Azad University, Tehran North Branch, Department of Computer, Tehran, Iran
[c] Department of Computer, Amirkabir University of Technology, Tehran, Iran

ARTICLE INFO

Article history:
Received 9 September 2011
Received in revised form 13 January 2012
Accepted 2 February 2012

Keywords:
Topology control
Learning automata
Sleep mode
Ad Hoc

ABSTRACT

Learning automata have been found to be useful in the systems with incomplete knowledge. Therefore, it can be used as a tool to solve problems of Ad Hoc networks, where nodes are mobile and operate within a dynamic environment, which entails possibly unknown and time varying characteristics. In this paper, after a short review on the related works, learning automata and CEC algorithm, which is a sleep based topology control algorithm, a modified version (called MCEC) is proposed. In addition, a probabilistic algorithm is recommended to make decision about whether or not a node has to sleep. Furthermore, a distributed algorithm is recommended; in order to improve the proposed probabilistic algorithm, using learning automata. Finally, nominated algorithms have been simulated in both of the stationary and non-stationary networks. In conclusion, as the simulation results show, the proposed algorithms outperform corresponding topology control algorithms and reveal the effectiveness of using learning automata.

1. Introduction

The most important challenge in the Ad Hoc networks is limited battery of each node. Therefore, many topology control algorithms have been proposed to address this problem. Topology control (TC) maintains a topology with certain properties (e.g. connectivity), while reducing energy consumption and/or increasing network capacity [1]. The importance of TC lies in the fact that it critically affects the performance of a system in several ways. For example, as it is shown in [2], TC improves network's spatial reuse. Hence its traffic carrying capacity increases. Topology control can be done by setting the redundant nodes in the sleep mode in such a way that the network remains connected while minimizing energy consumption. Topology control reduces energy usage of communication and consequently impacts on the battery life, which is a critical resource in many mobile applications. In addition, topology control impacts on the medium contention and collisions can be decreased as much as possible by putting redundant nodes to the sleeping mode or adjusting the transmission power of nodes.

Cost-effective topology control is critical in Ad Hoc networks. While much research has been carried out in this aspect using various methods, not enough attention has been given on utilizing modern heuristics like learning automata. In this paper we have assumed that, first each node is capable of adjusting its transmission range, and second there is a constant bounded power stretch factor. After describing CEC, a modified version is proposed. This modified version, called MCEC,

* Correspondence to: unit 4, Plot 6, golestan Alley no. 10, golestan St, marzdaran Blvd, shahrak jhandarmeri 14637-77984, Tehran, Iran. Tel.: +98 2144225020; +98 9362097924 (Mob.).
E-mail addresses: mina_shirali@yahoo.com, mina.shirali1983@gmail.com (M. Shirali).

0898-1221/$ – see front matter © 2012 Elsevier Ltd. All rights reserved.
doi:10.1016/j.camwa.2012.02.014

provides better results by solving the problem of CEC. In addition, two probabilistic approaches are proposed to decide about putting a node to the sleep mode. Also, a distributed algorithm is proposed that improves these probabilistic approaches and implies fitness of using learning automata.

The rest of this paper is organized as follows: in Section 2, we introduce some related works. Section 3, describes the distributed learning automata. Section 4, describes how to compute the fitness of an action in the learning automata. Section 5, describes CEC, its problem and the proposed modification. Section 6, represents our proposed probabilistic topology control approaches to determine whether a node has to remain active or go to the sleep mode. Section 7, represents a learning automata based topology control algorithm that improves the proposed probabilistic algorithm. Section 8, gives the performance evaluation of the proposed algorithms compared to the corresponding topology control algorithms, and finally Section 9 concludes the paper.

2. Related works

An important problem of MANETs is limited battery of nodes. Low power consuming techniques are generally acquired by adjusting the transmission range or setting the redundant nodes in the sleep mode [3]. In a CDS-based topology control technique, a backbone – comprising a set of highly connected nodes – is formed, which allows communication between any arbitrary pair of nodes in the network. In [4] a novel topology construction protocol based on the idea of polygons, called poly, is proposed and it is shown that formation of a polygon in the network provides a reliable and energy-efficient topology. They compared the performance of Poly with three prominent CDS-based topology construction protocols, namely CDS-Rule K [5], Energy-efficient CDS (EECDS) [6] and A3 [7]. Their simulation results demonstrate that poly performs consistently better in terms of message overhead and other selected metrics and achieves better connectivity under highly dynamic network topologies. Hu et al. proposed a distributed cross-layer traffic-aware (TAP) algorithm [8], where nodes get dynamic traffic characteristics as well as active neighbors within two hops periodically and then make decision on whether to sleep or not based on both the traffic pattern and local connectivity.

Geographic adaptive fidelity (GAF) and cluster-based energy conservation (CEC) [9] are two most well known sleep based topology control algorithms. GAF identifies redundant nodes by their physical location and a conservative estimate of radio range. GAF is deployed in the virtual grid topology, where in each cell of this grid, one node remains active to keep network connected. CEC directly observes radio connectivity to determine redundancy. Consequently, it is more aggressive at identifying duplication and more robust to radio fading.

As mentioned before, some topology control algorithms try to address this problem by transmission power control. Many works are done on power-based topology control in the Ad Hoc networks. Following this, we introduce a few of the most well-known algorithms. In LMST [10], each node builds a local minimum spanning tree to include its one hop neighbors, and selects neighbors in the computed MST as logical neighbors. A fully distributed, asynchronous, and localized protocol, called K-Neigh [11], is proposed based on determining the number of neighbors, in which each node try to keep a node degree over its neighbors to achieve full connectivity with a high probability. Also, LINT [12], absolute distance-based (ABD) [13] and predictive distance-based (PRD) [13] algorithms attempt to maintain the logical number of neighbors between predefined values; $K_{\min}$ and $K_{\max}$.

However many studies are done on topology control using Meta heuristics, to the best of our knowledge, none of them are done on the sleep based area. Many works are done on power based topology control using game theory [14–21]. ToCMA [22] uses a combination of local search and genetic algorithm to solve the minimum energy network connectivity and provides better results in comparison with MST.

In Ad Hoc networks, nodes can be mobile and operate within a dynamic environment, which entails possibly unknown and time varying characteristics. On the other hand, learning automata is beneficial particularly in the systems with incomplete knowledge. Therefore, it can be utilized as a tool to solve problems of these networks. An important problem of these networks is limited battery of nodes.

El-Osery and Baird [23] have proposed a transmission power control algorithm based on the learning automata, in which the penalty or reward is assigned according to the number of collisions. If the collision level is low a reward is given, otherwise the action is penalized. But this algorithm did not consider a lower bound for transmission power. That is, when collision occurs, transmission power decreases without considering the actual power level needed to keep a node degree over neighbors.

Learning automata have been also used to solve other problems of the wireless networks, such as medium access control, data rate adaptation, data aggregation, and multicast routing. For example, a data aggregation scheme is proposed [24], in which each node is equipped with a learning automata to learn optimum paths that have maximum aggregation ratio, collectively. Akbari-Torkestani and Meybodi [25] have proposed a MAC protocol to address the burst traffic problem of clustered wireless Ad Hoc networks, in which three learning automata based algorithms are proposed for cluster formation, code assignment, and slot assignment. In addition, they have used learning automata to solve the routing problem, in the Ad Hoc wireless networks [26,27]. In their recommended algorithm a stochastic graph is built which represents the virtual multicast backbone of the network and contains routes with higher lifetime. Then a distributed learning automata based algorithm is applied to this graph to solve the multicast routing problem. The virtual backbone graph is also exploited in this algorithm to solve the broadcast storm problem that is inherent in broadcast based routing via global flooding [27].

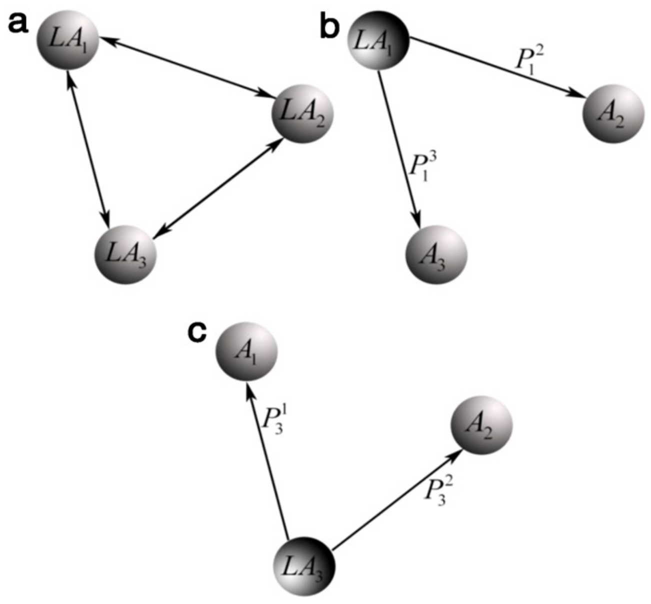


Fig. 1. A distributed learning automata with three learning automata.

3. Distributed learning automata

A distributed learning automata (DLA) is a network of learning automata (LA) which collectively cooperate to solve a particular problem [25–29]. Formally, a DLA can be defined as a graph DLA $= (V, E)$, where $V = \{\mathrm{LA}_1, \mathrm{LA}_2, \ldots, \mathrm{LA}_n\}$ is the set of LA and $E \subset V \times V$ is the set of edges in the graph. Number of actions for a particular LA in the DLA, is equal to the number of LA's that are connected to that LA. Therefore, maximum number of actions for a particular LA_k $(k = 1, 2, \ldots, m)$ is equal to the number of LA's (i.e. $|V|$). Each LA_k keeps a probability vector p_k to determine the probability of selecting an action. For example, p_k^j represents the probability of selecting action j by LA_k.

Selection of an action by LA in the network activates one LA corresponding to that action. In other words, transition on the edge of the graph DLA(k, j), occurs when LA_k is activated and selects its action (j) by the probability of p_k. After that, LA_j is activated for the next iteration. One DLA with three LA is shown in the Fig. 1. The activated automata LA_1 with its actions, A_2 and A_3 is shown in the Fig. 1(b). Then, as shown in the Fig. 1(c), LA_1 selects A_3 with probability p_1^3 and LA_3 is activated, consequently.

4. CEC

CEC is a neighbor, cluster and sleep based topology control algorithm. In some applications, where geographic location information is not available, GAF fails. Therefore CEC was proposed to solve this problem. As it is shown in [9], CEC outperforms GAF. This algorithm consists of three phases; cluster-head selection, gateway selection, and duty cycle determination. Assuming that network is synchronous, it works in the following manner:

1. Each node broadcasts a discovery message along with its node ID and estimated lifetime.
2. After awhile the node that has the longest life time among its neighbors, declares itself as a cluster-head and broadcasts this information.
3. Each node that is not a cluster-head and has received cluster-head messages from more than one cluster-head, declares itself as a gateway node and broadcasts this information.
4. All nodes except cluster-heads and gateway nodes go to the sleep mode to save more energy.
5. After re-clustering interval (RCI), in the next re-clustering, entire clustering process is reiterated. The RCI is a fraction of cluster's life time (LT_C).

Note that for gateway selection from multiple gateways, those ones that have the longest lifetime or cover more cluster-heads are assigned a higher priority and other gateway nodes can also go to sleep mode to save more energy. In CEC, each node that has the longest life-time among its neighbors announces itself as the cluster-head. This may lead to the chain problem. For example, consider a scenario like Fig. 2. As you can see in this figure each node is marked with (a) or (b), where 'a' and 'b' imply the name and lifetime of a node, respectively. In this scenario, node 'u' declares itself as a cluster-head.

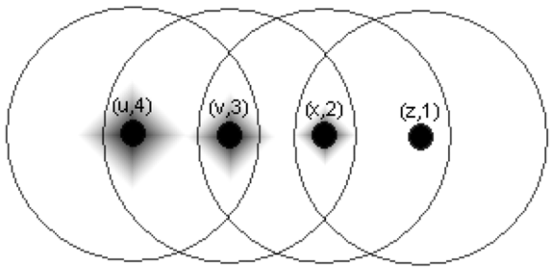


Fig. 2. The chain problem.

Therefore, nodes '*x*' and '*z*' do not have any cluster-head to rely on and do not sleep consequently. On the other hand, since node '*v*' has a stronger neighbor, it can go to the sleep mode. However when node '*v*' sleeps, network becomes disconnected.

5. Modified CEC

To solve the chain problem of CEC a modified version of CEC (MCEC) is proposed. In MCEC instead of accepting imposed cluster-head, each node determines its cluster-head itself. The following description clarifies inner working stages of MCEC:

1. Each node '*u*' broadcasts a discovery message along with its node ID and estimated lifetime.
2. After awhile each node that has the longest life time among its neighbors, broadcasts a cluster-head message.
3. Each node that is not a cluster-head and has received cluster-head messages from more than one cluster head, declares itself as a gateway node and broadcasts this information.
4. Each node that is not cluster-head and has not received any cluster head message, selects the strongest node among its neighbors as the cluster head and sends a cluster-head message to this selected neighbor.
5. Each node that has received a cluster-head message with its ID, declares itself as the cluster-head and remains active.
6. Except cluster-head and gateway nodes, other nodes are powered off to conserve more energy.
7. After re-clustering interval (RCI), entire clustering process is reiterated.

6. Probabilistic sleep-based topology control

In probabilistic sleep-based topology control (PSToC) algorithm each node estimates the likelihood of its sleep mode in order to adapt with environmental changes. This algorithm works in the following manner:

1. Each node broadcasts a discovery message along with its node ID and estimated lifetime.
2. Each node estimates its sleeping probability according to the manner described later.
3. Each node that is not a cluster-head and has received at least one cluster-head message goes to sleep mode according to its estimated sleeping probability.
4. After re-clustering interval (RCI), entire clustering process is reiterated.

To estimate the sleeping probability of a node, distance and number of neighbors are used as input parameters. It is obvious that sleeping probability of a particular node '*u*' has a positive relation with the number of its one hop neighbors that have longer life time (i.e. stronger neighbors) and a negative relation with the mean distance from these stronger neighbors. In other words, sleeping probability decreases as the mean distance between node and its stronger neighbor increases.

On the other hand, receiving power decreases as distance increases. Therefore, location information necessity can be omitted by having knowledge about receive and transmit powers. In PSToC each node uses the number of its stronger neighbors and receiving powers as input parameters to estimate its sleeping probability. For this purpose, it computes W_n and W_r as Eqs. (1) and (2):

$$W_n = \frac{N_d}{N_d^{\max}} \tag{1}$$

$$W_r = \frac{R_d}{R_d^{\max}} \tag{2}$$

where, W_n is the weight of stronger neighbors, W_r is the obtained weight for the receiving powers from stronger neighbors, N_d is the number of stronger neighbors, $N_d^{\max}$ is maximum value of N_d, R_d is the receiving power fraction and $R_d^{\max}$ is the maximum value of R_d, through d steps. R_d is computed as Eq. (3):

$$R_d = \frac{\sum_{i=1}^{N_d} \frac{rx_{i,d}}{tx_{i,d}}}{N_d} \tag{3}$$

where, the hello message from "ith stronger neighbor" is sent with $tx_{i,d}$ and received with $rx_{i,d}$ watt. Finally, sleeping probability is computed according to Eq. (4):

$$P_{\text{sleep}} = W_n \times W_r. \tag{4}$$

In PSToC, a node can sleep, if $random(P_{\text{sleep}}) > random(1 - P_{\text{sleep}})$. In this inequality *random* implies a function that generates a random value between zero and given input parameter. Alternatively, in the Mid-PSToC, a node sleeps if obtained value of P_{sleep} satisfies the following inequality, where $P_{\text{sleep},l}$ is the sleeping probability in the l'th step:

$$P_{\text{sleep}} > \frac{\sum_{l=1}^{d} P_{\text{sleep},l}}{d}. \tag{5}$$

Each node that decides to go to the sleep mode can sleep for $\frac{t}{c \times v}$ s. Where, t is its transmission range, v implies the velocity and c is a positive real constant.

7. Proposed algorithm based on the learning automata

In this section a leaning automata based algorithm (called LASToC) is proposed to determine whether a node has to sleep or not. At first, it is necessary to know how to compute fitness of an action. f_{dik} implies k'th fitness factor of an action in the i'th node and d'th step and it is computed through Eq. (6):

$$f_{dik} = \frac{\prod_{j=1}^{m} (\rho_{ij} \Phi(u_{dij}) + \overline{\rho_{ij}}) - \prod_{j=1}^{m} \overline{\rho_{ij}}}{1 - \prod_{j=1}^{m} \overline{\rho_{ij}}} \quad \forall 0 \le \rho_{ij},\ \Psi, \Phi \le 1 \tag{6}$$
$$0 \le \rho_{ij} \le 1 \Rightarrow \overline{\rho_{ij}} = 1 - \rho_{ij}$$

where, m is the number of utilization metrics, u_{dij} is j'th metric in the i'th node and d'th step. Also, p_{ij}, implies the effect constant of u_{dij}. If u_{dij} has to decrease, we normalize it as Eq. (7). Otherwise, it is normalized as Eq. (8).

$$\phi(u_{dij}) \begin{cases} \dfrac{u_{dij}^{\max} - u_{dij}}{u_{dij}^{\max} - u_{dij}^{\min}} & u_{dij}^{\min} \le u_{dij} \le u_{dij}^{\max} \\ 0 & u_{dij} > u_{dij}^{\max} \\ 1 & u_{dij} < u_{dij}^{\min} \end{cases} \tag{7}$$

$$\phi(u_{dij}) \begin{cases} \dfrac{u_{dij} - u_{dij}^{\min}}{u_{dij}^{\max} - u_{dij}^{\min}} & u_{dij}^{\min} \le u_{dij} \le u_{dij}^{\max} \\ 1 & u_{dij} > u_{dij}^{\max} \\ 0 & u_{dij} < u_{dij}^{\min}. \end{cases} \tag{8}$$

In this equation, $u_{dij}^{\min}$ and $u_{dij}^{\max}$ is the minimum and maximum threshold of u_{dij}, respectively. In this research, these thresholds are updated dynamically as Eq. (9):

$$u_{dij}^{\max} = \max\{u_{dij}^{\max}, u_{eij}\}, \quad 0 \le e \le d. \tag{9}$$

That is, if the recent value of u_{dij} is greater than $u_{dij}^{\max}$, then $u_{dij}^{\max}$ must be updated as u_{dij}. The $u_{dij}^{\max}$ is not the actual value of maximum threshold, but provides an overestimation.

Alternatively, we could choose it as a large constant. But as one can imagine, choosing a large maximum threshold compared to u_{dij} leads to a significant decrease in the convergence speed. In addition, $\Phi\ (u_{dij})$ becomes a small value and rounding errors reduces system performance. As mentioned before, p_{ij} implies the effect constant of u_{dij} and it is by default equal to one. In this paper two utilization metrics are considered, which are computed according to the sleeping probability (P_{sleep}) and remained power. To obtain a better performance, we can use fitness factor in the computation of reward (α) or penalty (β) parameters as shown in the Eqs. (10) and (11), where ω and λ are positive real numbers.

$$\alpha = \omega_\alpha + \lambda_\alpha f_{dik} \tag{10}$$
$$\beta = \omega_\beta + \lambda_\beta (1 - f_{dik}). \tag{11}$$

Table 1
Simulation summary of sleep-based topology control.

	Received (bit)	Received (bit/mW)	Delay (s)	Collision (packet)	Dropped (bit)	Throughput (bit)
CEC	10 225 664	3.932948	34 089.65	1124	2 366 400	120 718 052
MCEC	12 163 072	4.304252	35 943.09	1449	2 784 640	130 678 652
Mid-PSToC	12 221 440	3.998782	42 629.87	1205	2 770 552	148 914 976
PSToC	12 934 144	4.166964	33 639.12	1407	2 837 048	157 542 544
LASToC	13 435 904	4.616204	32 733.44	1097	2 484 544	163 812 256
LASToC B	12 863 488	4.384823	34 523.68	1110	2 160 344	149 393 856
LASToC S	12 573 696	4.305211	40 262.78	1333	2 770 544	152 116 896
LASToC B S	12 537 856	4.230858	28 991.22	1331	2 162 584	151 071 704

After computing fitness, now we can continue describing LASToC. The proposed algorithm is implemented with both of the standard and distributed models of learning automata. As mentioned before, the standard model of learning automata is a DLA in which the automata set has only one learning automata. LASToC works in the following manner in the i'th node:

```
1. Initialize:
        1.1. $d = 0$
        1.2. Initialize probability vector $p_{1ij}$, for each LA$_j$.
        1.3. Activate an automata randomly (LA$_a$).
2. $d = d + 1$.
3. Activated automata (LA$_a$) selects an action according to the $p_{dia}$.
4. The node goes to the sleep mode, according to the selected action.
5. Neighbor discovery phase is performed and a set of reachable neighbors is computed.
6. LA$_a$ computes fitness of the selected action and updates its action probability vector, according to the reward–
   penalty model of the stochastic learning automata [11, 12].
7. A learning automata is activated, according to the selected action.
8. If probability vector converges, process ceases. Otherwise, the process is reiterated.
```

Note that, in LASToC, the action set of learning automata has two members and each one of them mentions one of the sleep or active modes. After selecting an action, fitness is computed according to the selected action. That is, if the selected action was placing node in the sleep mode, utilization metrics should be normalized as Eq. (8) to compute sleeping fitness. Otherwise, these metrics should be decreased and normalized as Eq. (7) to compute the fitness of putting a node in the active mode.

8. Simulation

In this section, several simulation results are presented to demonstrate effectiveness of the proposed algorithms. OPNET is used for simulation, which is known as a powerful network simulator. Also, all the predefined settings are used for IEEE 802.11b standard model. One of the basic features of a mobile Ad Hoc network (MANET) is that, nodes move according to a mobility model. For MANETs the random way-point model (RWP) [30] is, by far, the most popular one. Speed (m/s) and pause time (s) parameters of all nodes are set as uniform (0, 10) and constant (1), respectively. All nodes are placed in a 500×500 m area, non-uniformly. Due to the massive amount of data to be processed, strictly a simulator issue, the number of nodes is limited to what is needed to proof the concept. In this research, simulation has run for 7200 s. In this paper, main factors used for comparison are:

- Received bits per watt
- Received bits.

However, other parameters like collision, dropped bits, etc...., are also summarized in the Table 1. Note that, network density is an important issue that has to be considered in the stationary networks. It is assumed that each node has at least one neighbor at first. Comparing with CEC, MCEC provides more appropriate connectivity. However, in the dense networks where each node has more neighbors, there is a stronger neighbor with a higher probability in the middle of the chain. In other words, forming a chain like Fig. 2 seems less possible as network density increases. Therefore, in the dense networks, the chain problem of CEC occurs less, and MCEC converges to the CEC, consequently (Figs. 3 and 4). The proposed algorithms are implemented in both stationary and non-stationary networks. MCEC provides stronger connections in comparison with CEC. Therefore, it acquires better results in non-stationary networks, where connections are weak (Figs. 5 and 6).

A probabilistic approach is also proposed in this paper, in which each node decides to go to the sleep mode when it finds itself as a redundant node, while those nodes that are far from their cluster-head remain active to relay packets. As it is shown in the following figures, PSToC and Mid-PSToC outperform CEC. Further, a learning automata based algorithm is proposed, called LASToC, which provides more received data and more received bits per milliwatt (mW). It has also less delay and dropped data (Figs. 7 and 8). As the results show, it outperforms other algorithms in both stationary and non-stationary scenarios.

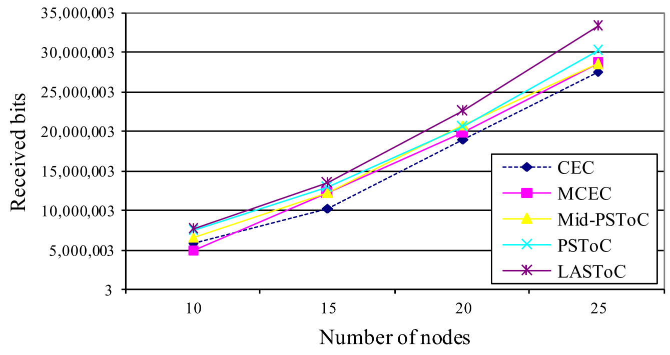


Fig. 3. Impact of network density on received bits.

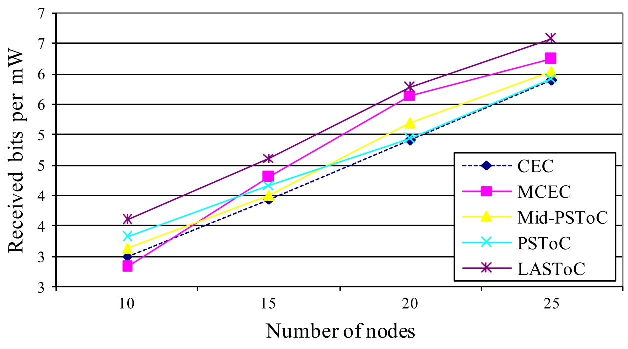


Fig. 4. Impact of network density on received bits per mW.

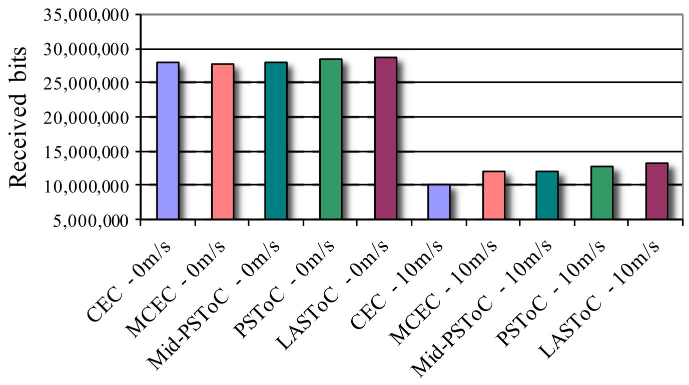


Fig. 5. Impact of mobility on received bits.

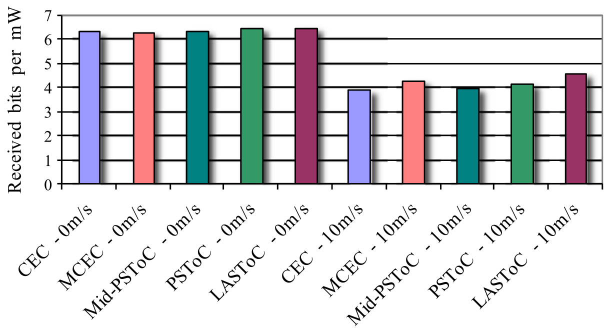


Fig. 6. Impact of mobility on received bits per mW.

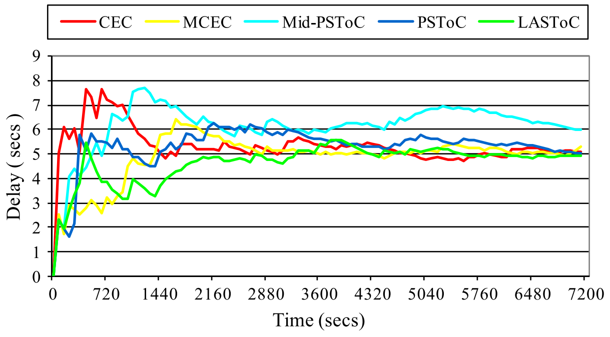


Fig. 7. Average delay in LASToC.

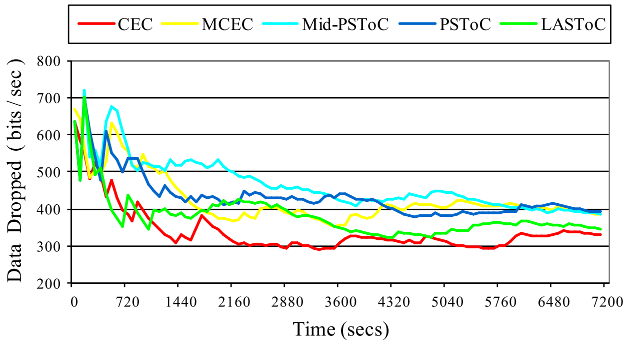


Fig. 8. Average amount of dropped data in LASToC.

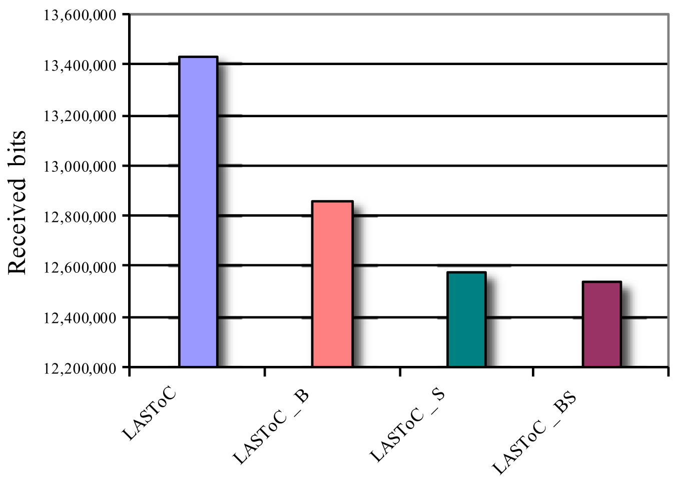


Fig. 9. Impact of using fitness aware reward–penalty and DLA on the received bits.

As mentioned before, LASToC is implemented with both of standard and distributed models of learning automata. Simulation results are shown in the Figs. 9 and 10, where 'S' implies omitting the use of DLA, and 'B' implies omitting the use of fitness aware reward–penalty. DLA considers a weighted graph of transitions between actions and provides more accurate decisions, consequently. Therefore, better results can be provided by using DLA. Also, by using fitness aware reward–penalty, a continuous and more accurate value for penalty and reward parameters of learning automata is provided and better results are obtained.

9. Conclusion

Topology control algorithms presented in this paper, try to increase received data while minimizing energy consumption. A modified version of CEC (called MCEC) is proposed in this paper, which resolves the chain problem of CEC and provides more appropriate connectivity. The chain problem of CEC occurs less in the dense networks, and MCEC converges to the CEC

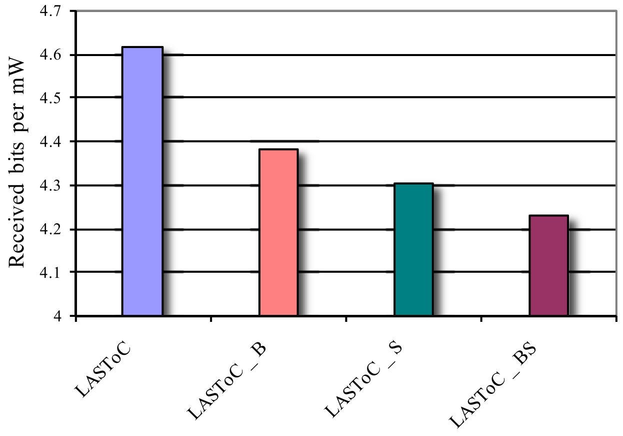


Fig. 10. Impact of using fitness aware reward–penalty and DLA on the received bits per mW.

as network density increases, consequently. On the other hand, MCEC provides better results when mobility is introduced in the network, due to the appropriate connectivity provided to solve the chain problem. A probabilistic approach is also proposed (PSToC and Mid-PSToC) to determine whether a node has to sleep or not. Comparing with CEC, the probabilistic approach has less overhead and a simpler implementation. In addition, by using learning automata we have improved the outcome. However LASToC uses the same idea used in the PSToC and Mid-PSToC, but it provides better results. This implies the effectiveness of using learning automata. We have implemented LASToC with both of the standard and distributed models of learning automata. As it is shown in the simulation results, distributed model of learning automata provides better results. Also, by using the fitness aware reward–penalty, the final results were further improved.

Integral simulation results for two hours are summarized in the Table 1. As mentioned before, 'S' implies omitting the use of DLA, and 'B' implies omitting the use of fitness aware reward–penalty process. "Dropped" column of the table represents the number of bits dropped due to their exceeded retry threshold and "Throughput" column represents the average number of bits received by receiver channel, in the MAC layer of network.

References

[1] P. Santi, Topology Control in Wireless Ad Hoc and Sensor Networks, John Wiley and Sons, 2005.
[2] P. Gupta, P.R. Kumar, The capacity of wireless networks, IEEE Transactions on Information Theory 46 (2000) 388–404.
[3] A. Jie-Jia, C. Jian, B. Guiran, W. Yingyou, S. Jingping, Multi-objective optimization for coverage control in wireless sensor network with adjustable sensing radius, Journal Computers & Mathematics with Applications 57 (2009) 1767–1775.
[4] H.K. Qureshi, S. Rizvi, M. Saleem, S.A. Khayam, V. Rakocevic, M. Rajarajan, Poly: a reliable and energy efficient topology control protocol for wireless sensor networks, Journal of Computer Communications 34 (2010) 1235–1242.
[5] J. Wu, M. Cardei, F. Dai, S. Yang, Extended dominating set and its applications in Ad Hoc networks using cooperative communication, IEEE Transactions on Parallel and Distributed Systems 17 (2006) 851–864.
[6] Z. Yuanyuan, X. Jia, H. Yanxiang, Energy efficient distributed connected dominating sets construction in wireless sensor networks, in: ACM Conference on Communications and Mobile Computing, 2006, pp. 797–802.
[7] P.M. Wightman, M.A. Labrador, A3: a topology construction algorithm for wireless sensor network, in: IEEE Conference on Global Communications, Globecom, 2008.
[8] P. Hu, P. Hong, J. Li, Z.Q. Qin, TAP: traffic-aware topology control in on-demand Ad Hoc networks, Journal of Computer Communications 29 (2006) 3877–3885.
[9] Y. Xu, S. Bien, Y. Mori, J. Heidemann, D. Estrin, Topology control protocols to conserve energy in wireless Ad Hoc networks, Technical Reports of Center for Embedded Network Sensing, Los Angeles, USA, 2003.
[10] N. Li, J.C. Hou, L. Sha, Design and analysis of an MST-based topology control algorithm, IEEE Transaction on Wireless Communications 4 (2005) 1195–1206.
[11] D.M. Blough, M. Leoncini, G. Resta, P. Santi, The K-Neigh protocol for symmetric topology control in Ad Hoc networks, in: ACM Conference on Mobile Ad Hoc Networking and Computing, MobiHoc, New York, USA, 2003, pp. 141–152.
[12] J.B.D. Cabrera, R. Ramanathan, C. Guitierrez, R. Mehra, Stable topology control for mobile Ad Hoc networks, IEEE Communications Letters 11 (7) (2007) 574–576.
[13] A.C.C. Yao, Mobility-aware topology control in mobile Ad Hoc networks, Journal of Computer Communications 31 (2008) 3521–3532.
[14] Y. Xing, R. Chandramouli, Stochastic learning solution for distributed discrete power control game in wireless data networks, IEEE/ACM Transactions on Networking (2008).
[15] S. Koskie, Z. Gajic, A nash game algorithm for SIR-based power control in 3G wireless CDMA networks, IEEE/ACM Transactions on Networking 13 (2005) 1017–1026.
[16] H. Park, M. van-der-Schaar, Multi-user multimedia resource management using nash bargaining solution, in: IEEE Conference on Acoustics, Speech and Signal Processing, 2007, vol. 2, pp. 717–720.
[17] C.G. Yang, J.D. Li, Z. Tian, Optimal power control for cognitive radio networks with coupled interference constraints: a cooperative game-theoretic perspective, IEEE Transactions on Vehicular Technology 59 (2010) 1696–1706.
[18] Y. Chen, G.G. Yu, Z. Zhang, H.H. Chen, P.L. Qiu, On cognitive radio networks with opportunistic power control strategies in fading channels, IEEE Transaction on Wireless Communications 7 (7) (2008) 2752–2761.
[19] P. Zhou, W. Yuan, W. Liu, W. Cheng, Joint power and rate control in cognitive radio networks: a game-theoretical approach, in: IEEE Conference on Communications, 2008, pp. 3296–3301.

[20] M. Rasti, A.R. Sharafat, B. Seyfe, Pareto-efficient and goal-driven power control in wireless networks: a game-theoretic approach with a novel pricing scheme, IEEE/ACM Transaction on Networks 17 (2009) 556–569.

[21] X. Wang, Q. Zhu, Power control for cognitive radio base on game theory, in: IEEE Conference on Wireless Communications, Networking and Mobile Computing (WICoM), 2007, pp. 1256–1259.

[22] A. Konstantinidis, K. Yang, H.H. Chen, Q. Zhang, Energy aware topology control for wireless sensor networks using memetic algorithms, Journal of Computer Communications 30 (2007) 2753–2764.

[23] A.I. El-Osery, D. Baird, A learning automata based power management for Ad Hoc networks, in: IEEE Conference on Power Systems, Man and Cyberntics, 2005, pp. 3569–3573.

[24] M. Esnaashari, M.R. Meybodi, Data aggregation in sensor networks using learning automata, Journal Wireless Networks 16 (2009) 687–699.

[25] J. Akbari-Torkestani, M.R. Meybodi, An efficient cluster-based CDMA/TDMA scheme for wireless mobile Ad Hoc networks: a learning automata approach, Journal of Network and Computer Applications 33 (2010) 477–490.

[26] J. Akbari-Torkestani, M.R. Meybodi, Mobility-based multicast routing algorithm for wireless mobile Ad Hoc networks: a learning automata approach, Journal of Computer Communications 33 (2010) 721–735.

[27] J. Akbari-Torkestani, M.R. Meybodi, An intelligent backbone formation algorithm for wireless Ad Hoc networks based on distributed learning automata, Journal of Computer Network 54 (2010) 826–843.

[28] R. Forsati, M.R. Meybodi, Effective web page recommendation algorithms based on distributed learning automata and weighted association rules, Journal Expert Systems with Applications 37 (2010) 1316–1330.

[29] K. Narendra, M.A.L. Thathachar, Learning Automata: An Introduction, Prentice Hall, Englewood Cliffs, 1989.

[30] C. Bettstetter, G. Resta, P. Santi, The node distribution of the random waypoint mobility model for wireless Ad Hoc networks, IEEE Transactions on Mobile Computing (2003) 257–269.