ISBN 978-1-84626-xxx-x
Proceedings of 2009 International Conference on Machine Learning and Computing
Perth, Australia, 10-12 July, 2009, pp. xxx-xxx

Cellular Learning Automata-based Channel Assignment Algorithms in Mobile Ad Hoc Network

Alireza Enami Eraghi[1], Javad Akbari Torkestani[2],
Mohammad Reza Meybodi[3] and Amir Hosein Fathy Navid[4]

[1]Computer Engineering Department, Islamic Azad University, Farahan Branch, Iran. alireza_enami@yahoo.com
[2] Department of Computer Engineering, Islamic Azad University, Science and Research Branch, Tehran, Iran. j-akbari@iau-arak.ac.ir
[3]Computer Engineering Department, Amirkabir University of Technology, Tehran, Iran. mmeybodi@aut.ac.ir
[4]Computer Engineering Department, Islamic Azad University, Hamedan Branch, Iran. amir.fathy.n@qazviniau.ac.ir

Abstract. The wireless mobile ad hoc network (MANET) architecture has received a lot of attention recently. This paper considers the access of multiple channels in a MANET with multi-hop communication behavior. We point out several interesting issues when using multiple channels. Our proposed algorithms enables hosts to utilize multiple channels by switching channels dynamically, thus increasing network throughput and decrease packet delay. In this paper, we first introduce the model of cellular learning automata in which learning automata are used to adjust the state transition probabilities of cellular automata. Then a cellular learning automata based channel assignment algorithm is proposed.

Keywords: Medium Access Control, Channel Assignment, Cellular Learning Automata.

1. Introduction

A mobile ad-hoc network (MANET) is formed by a cluster of mobile hosts without the infrastructure of base stations. The applications of MANETs appear in places where pre deployment of network infrastructure is difficult or unavailable (e.g., fleets in oceans, armies in march, natural disasters, battle fields, festival field grounds, and historic sites). A working group called MANET [l] has been formed by the Internet Engineering Task Force (IETF) to stimulate research in this direction.

A MAC (medium access control) protocol is to address how to resolve potential contention and collision on using the communication medium. Many MAC protocols have been proposed for wireless networks [2][3][4], which assume a common channel shared by mobile hosts. We call such protocols single-channel MAC protocols. A standard that has been widely accepted based on the single-channel model is the IEEE 802.11 [5]. One common problem with such protocols is that the network performance will degrade quickly as the number of mobile hosts increases, due to higher contention/collision.

One approach to relieving the contention/collision problem is to utilize multiple channels. With the advance of technology, empowering a mobile host to access multiple channels is already feasible. We thus define a multi-channel MAC protocol as one with such capability [6][7]. Using multiple channels has several advantages. First, while the maximum throughput of a single-channel MAC protocol will be limited by the bandwidth of the channel, the throughput may be increased immediately if a host is allowed to utilize multiple channels. Second, as shown in [8], using multiple channels will experience less normalized propagation delay per channel than its single-channel counterpart, where the normalized propagation delay is defined to be the ratio of the propagation time over the packet transmission time. Therefore, this reduces the probability of collisions. Third, since using a single channel is difficult to support quality of service (QoS), it is easier to do so by using multiple channels.

Here, we use “channel” upon a logical level. Physically, a channel can be a frequency band (under FDMA), or an orthogonal code (under CDMA). How to access multiple channels is thus technology-dependent.

A multi-channel MAC typically needs to address two issues: channel assignment (or code assignment) and medium access. The former is to decide which channels to be used by which hosts, while the later is to resolve the contention/collision problem when using a particular channel.

The rest of this paper is organized as follows. In section 2, a brief review of cellular learning automata and irregular cellular learning automata is given. Sections 3 and 4 presents the proposed algorithm and numerical example, respectively and section 5 concludes the paper.

The rest of this paper is organized as follows: in section 2, a brief review of irregular cellular learning automata is given. In section 3, we describe proposed algorithms based on irregular cellular learning automata and The performance of the proposed algorithm is evaluated through the simulation experiments in section 4. Section 5 is conclusion.

2. Irregular Cellular Learning Automata

An irregular cellular learning automata (ICLA) [9] is a CLA in which the restriction of rectangular grid structure in traditional CLA is removed. This generalization is expected because there are applications such as wireless sensor networks, immune network systems, graph related applications, etc. that cannot be adequately modeled with rectangular grids. An ICLA is defined as an undirected graph in which, each vertex represents a cell which is equipped with a learning automaton. The learning automaton residing in a particular cell determines its state (action) on the basis of its action probability vector. Like CLA, there is a rule that the ICLA operate under. The rule of the CLA and the actions selected by the neighboring LAs of any particular LA determine the reinforcement signal to the LA residing in a cell. The neighboring LAs of any particular LA constitute the local environment of that cell. The local environment of a cell is non stationary because the action probability vectors of the neighboring LAs vary during the evolution of the ICLA.

3. The Proposed Channel Assignment Algorithms

In this method, we use the concept of Irregular cellular learning automata presented in section 2 for mapping a MANET topology to a network of learning automata. To do so, each mobile host is associated with a cell, and equipped with a learning automaton. Two cells are neighbors, if their corresponding hosts are within the transmission range of each other. Each host has n channels to access, and so each automaton has n actions to choose the given channels. To implement the proposed algorithms the following data structures should be defined.

1- CL[](Channel List): It contains the list of the free channels maintained by each host , and is dynamically computed. Channel j is not being used by host i, if CL[i,j]=1 and otherwise if CL[i,j]=0.

2- CPL[](channel Probability List): CPL[i,j] represents the probability of choosing channel j by host i.

3.1. The First Proposed Algorithm (CLA-1)

In this algorithm, when a sender has a packet to send; the learning automaton associated with the receiver chooses one of its action. If the chosen action(channel) doesn’t interfere with the channels used in its neighboring cells, then the chosen channel is assigned to the cell and rewarded; otherwise it is penalized. When the cell fails to assign the chosen channel, it retries to choose another channel. This process of algorithm repeats until all channel are being tested or a channel is found for assigning to the call. When the cell fails to find any channel to assign, the incoming call is blocked. The description of the first proposed algorithm is shown algorithmically in Fig. 1.

3.2. The Second Proposed Algorithm (CLA-2)

In this algorithm, when a sender has a packet to send; it sends its CL and PCL to receiver. The learning automaton in receiver, chooses one of its free channel (enable action) of sender that is common in sender and receiver and then it assigns to the call and the chosen channel is rewarded. If the free channel set of sender is empty, then the incoming call will be blocked. When selected channel is assigned to the incoming call, the sender and receiver informs the neighboring cells about this assignment. When a cell finds out that a channel is assigned in a neighboring cell, it disables the action of its learning automaton corresponding to the assigned channel. When a call is terminated, its channel is released. Then the sender and receiver inform the neighboring cells about this release and action of the learning automata in the neighboring cells are enabled. The description of the second proposed algorithm is shown algorithmically in Fig. 2.

```
Procedure Channel_Assignment_CLA1(CL_receiver,PCL_receiver)
Input: CL_receiver and PCL_receiver
Output: candid_channel
Begin
    For i:=1 to n do
        If CLA has any enable action then
            Set candid_channel=CLA.SelectAction(CL_receiver);
            Broadcast candid_channel to neighborhood receiver;
            For all neighborhood receiver
                If candid_channel has no conflict then
                    Reward action candid_cannel of CLA;
                    Return candid_channel;
                Else
                    Penalize action candid_cannel of CLA;
                    Disable action candid_cannel of CLA;
                End if
            End for
        End if
    End for
    If no channel take reward then
        Blocking call;
        Send "no transfer" message to sender;
        Exit of algorithm;
    End if
End.

Algorithm CLA1
    1)Sender sends a RTS packet to receiver;
    2) Set candid_channel =
        Procedure Channel_Assignmet_CLA1(CL_receiver,PCL_receiver);
    3) Receiver sends a CTS(candid_channel) packet to sender;
    4) Sender and reseiver switch to candid_channel and sender
       tranfers DATA;
    5) Receiver sends a ACK packet to reciever;
```

Fig. 1: First proposed algorithm (CLA-1)

```
Procedure Channel_Assignment_CLA2 (CL_sender, PCL_sender,
          CL_receiver)
Input: CL_sender,PCL_sender,CL_receiver
Output: candid_channel
Begin
    For i:=1 to n do
        If CLA has any enable action then
            Set candid_channel=CLA.SelectAction(CL_sender);
            If sender[candid_channel]==1 and
               receiver[candid_channel]==1 then
                Reward action candid_cannel of CLA;
                Return candid_channel;
            Else
                Penalize action candid_cannel of CLA;
                Disable action candid_cannel of CLA;
            End if
        End if
    End for
    If no channel take reward then
        Blocking call;
        Send "no transfer" message to sender;
        Exit of algorithm;
    End if
End.

Algorithm CLA2
    1)Sender sends a RTS(CL[i,j],CPL[i,j]) packet to receiver;
    2) Set candid_channel = Procedure
       Channel_Assignmet_CLA2(CL_sender,PCL_sender,CL_receiver);
    3) Set action candid_cannel of CLA's receiver = 0;
    4) Receiver sends a CTS(candid_channel) packet to sender;
    5) Set action candid_cannel of CLA's sender = 0;
    6) sender and receiver send RES(cch) to therir neighborhood for
       thet they disable  action candid_channel of their CLA;
    7) Sender and reseiver switch to candid_channel and sender
       tranfers DATA;
    8) Receiver sends a ACK packet to reciever;
    9) Set candid_channel in sender and receiver = 1;
    10) Sender and reciever send REL(cch) for set  candid_channel in
        CLA of neighborhood's sender  and receiver = 1;
```

Fig. 2: Second proposed algorithm (CLA-2)

4. Simulation Results

In this section, we compare our proposed algorithms with 802.11DCF [5] which uses a single channel, MMAC [6] which is a typical multi-channel MAC protocol and TMMAC [7] which is proposed recently and. All of them is implemented in GloMoSim [10][11], a scalable discrete event simulator developed by UCLA.

The simulation settings are as follows: the bit rate is 2Mbps for each channel; the data packet size is 512 bytes; the channel switch delay is set to 80us; the beacon interval is set to 100ms; the network area size is 1000m*1000m; constant-bit rate(CBR) traffic is used in the application layer; the source and destination of each follow are randomly chosen; AODV protocol is used in the routing layer; the communication range is 250m; the carrier sense is 500m; the minimal ATIM window size in TMMAC is 8.57ms and the maximal is 31.43ms; the length of the ATIM window size in MMAC is 20ms. In cases that number of nodes and channels are constant; they are 50 and 5 respectively. We use the following two metrics to evaluate the performance of proposed algorithms: a) throughput, b) delay packet.

Fig. 3, shows throughput for different number of nodes. The number of nodes is 10, 50 and 100 nodes in figures 3(a), 3(b) and 3(c) respectively. Fig. 4, shows packet delay for different number of nodes. The number of nodes is 10, 50 and 100 nodes in figures 4(a), 4(b) and 4(c) respectively.

Fig. 5, shows throughput for different number of channels. The number of channels is 5 and 20 channels in figures 5(a) and 5(b) respectively. Fig. 6, shows packet delay for different number of channels. The number of channels is 5 and 20 channels in figures 6(a) and 6(b) respectively.

Fig. 7, shows blocking probability of calls for different two proposed algorithms. The results of simulations are compared with the results obtained for IEEE809.11DCF, MMAC and TMMAC.

Fig. 8, shows the blocking probability for different algorithms for a typical run.

Fig. 9, shows the number of messages transmitted among the cell and its neighboring cells.

Fig. 10, shows evaluation of the interference for different channel. First and second proposed algorithms are in figure 10(a) and 10(b) respectively.

5. Conclusion

In this paper, cellular learning automata is introduced. The cellular learning automata is a model in which learning automata are used to adjust the state transition probabilities of cellular automata, then as application of cellular learning automata, a channel assignment algorithm based on cellular learning automata is given. Our proposed algorithms is compared with IEEE802.11DCF, MMAC and TMMAC simulation results show that proposed algorithms successfully exploits multiple channels to improve network throughput and delay packet. Second proposed algorithm is better or at least comparable to first proposed algorithm in most cases.

6. References

[1] IETF MANET Working Group. http://www.ietf.org/html.charters/manet-charter.html.

[2] V. Bhargavan, A. Demers, S. Shenker and L. Zhang, “MACAW-A Media Access Protocol for Wireless LANs”, in: Proceedings of the ACM SIGCOMM, pp. 212-225, 1994.

[3] C.L. Fullmer and G.L.Aceves, “Floor Acquisition Multiple Access (FAMA) for Packet-Radio Networks”, in: Proc. ACM SIGCOMM, Cambridge MA, 1995.

[4] P. Karn, “MACA-a New Channel Access Method for Packet Radio”, in: Proceedings of the ARRL/CRRL Amateur Radio 9th Computer Networking Conference September 22, 1990.

[5] IEEE Standards Department, “Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications”, IEEE standard 802.11, 1997.

[6] J. So and N. Vaidya, “Multi-Channel MAC for Ad Hoc Networks: Handling Multi-Channel Hidden Terminals Using a Single Transceiver”, ACM MobiHoc, pp. 222–233, 2004.

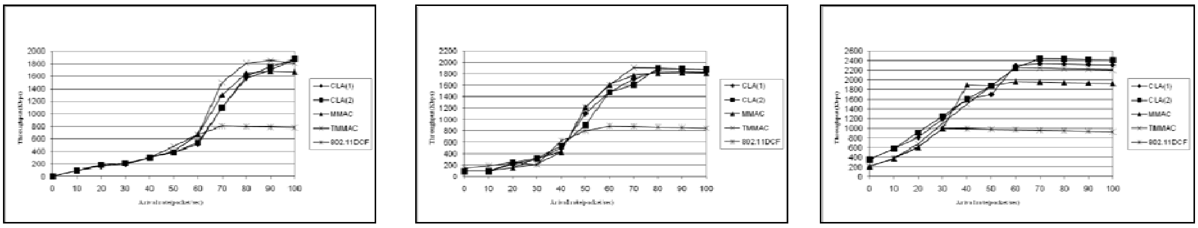
[7] J. Zhang, G. Zhou, C. huang, S.H. Son and J.H. Stankovic,” TMMAC: An Energy Efficient Multi-Channel MAC Protocol for Ad Hoc Networks”, This paper appears in: Communications ICC 2007. IEEE International Conference, pp. 3554-3561, 2007.

[8] A. Nasipuri, J. Zhuang and S.R. Das, “A Multichannel CSMA MAC Protocol for Multi-Hop Wireless Networks”, in: IEEE WCNC, 1999.

[9] M. Asnaashari and M.R. Meybodi, "Irregular Cellular Learning Automata and Its Application to Clustering in Sensor Networks", Proceedings of 15th Conference on Electrical Engineering (15th ICEE), Volume on Communication, Telecommunication Research Center, Tehran, Iran, May 15-17, 2007.

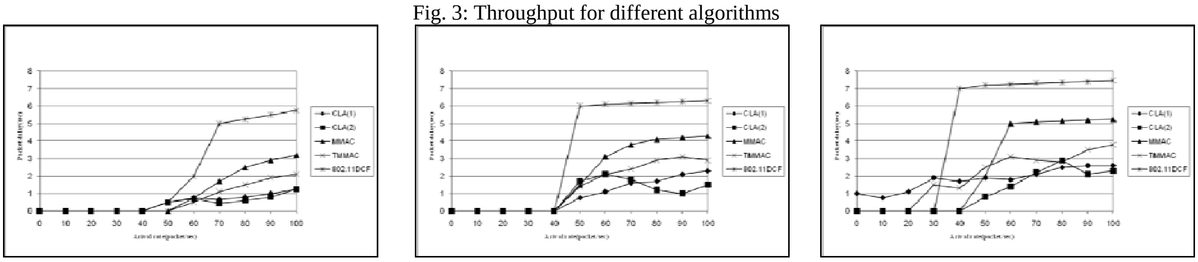
[10] X. Zeng, R. Bagrodia, and M. Gerla, “GloMoSim: A Library for Parallel Simulation of Large-Scale Wireless Networks”, in: PADS, 1998.

[11] http://pcl.cs.ucla.edu/projects/domains/glomosim.html.



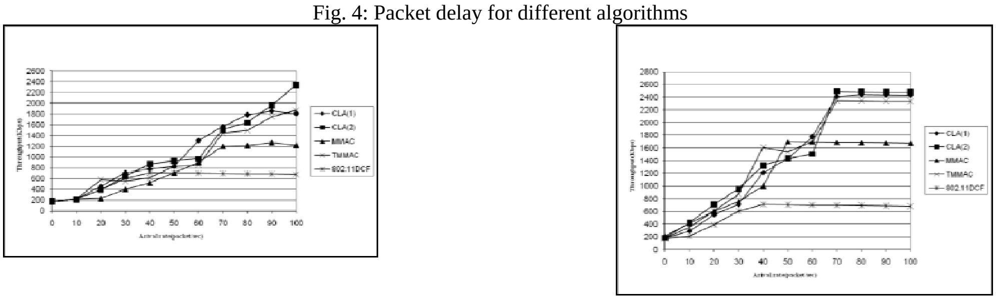
(a) n=10 (b) n=50 (c) n=100

Fig. 3: Throughput for different algorithms



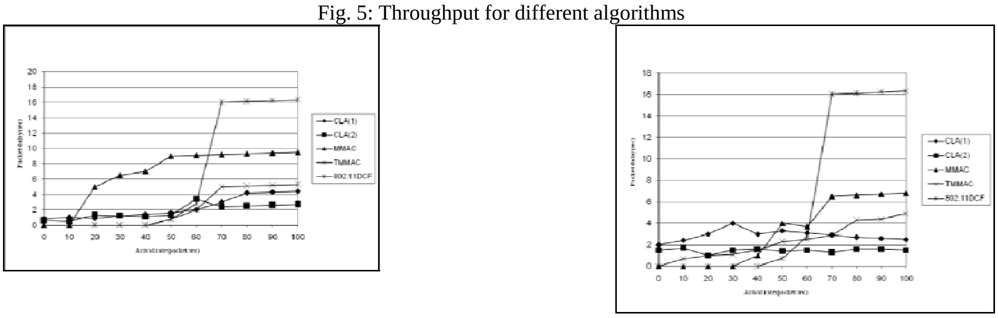
(a) n=10 (b) n=50 (c) n=100

Fig. 4: Packet delay for different algorithms



(a) n=5 (b) n=20

Fig. 5: Throughput for different algorithms



(a) n=5 (b) n=20

Fig. 6: Packet delay for different algorithms

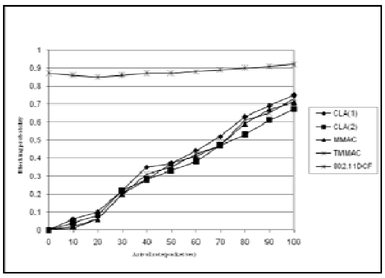


Fig. 7: Blocking probability of calls for different algorithms

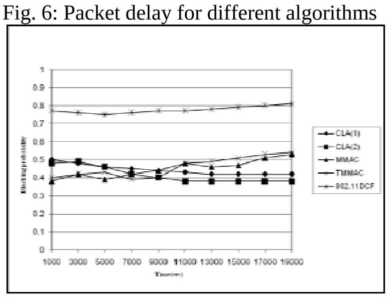


Fig. 8: Blocking probability of calls for different algorithms for a typical run

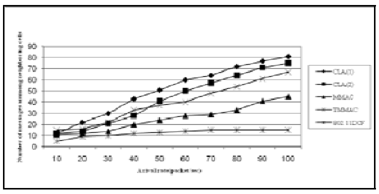
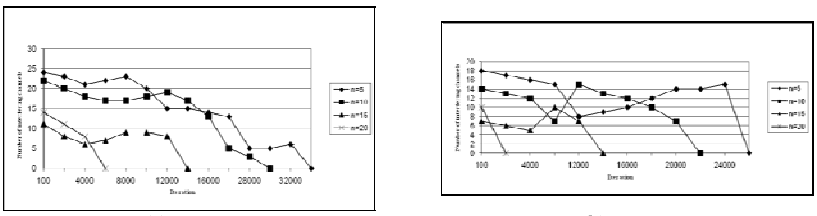


Fig. 9: Number of messages transmitted among the cell and its neighboring cells



(a) CLA-1 (b) CLA-2

Fig. 10: The interference for different channels for proposed algorithms