A Distributed Learning Automata based Gateway Load Balancing Algorithm in Wireless Mesh Networks

Maryam Kashanaki
Department of Computer and Information Sciences
Institute of Qazvin Azad University
Qazvin, Iran
m.kashanaki@gmail.com

Zia Beheshti
Department of Computer Engineering
Institute of Azad University, South Tehran Branch
Tehran, Iran
z_beheshti@azad.ac.ir

Mohammad Reza Meybodi
Department of Computer and Information Sciences
Institute of Amirkabir University
Tehran, Iran
mmeybodi@aut.ac.ir

***Abstract*— Wireless Mesh Networks (WMNs) are a rapidly maturing technology for providing high bandwidth broadband service to a large community of users. In WMNs, gateway nodes act as a central point of connectivity to the wired infrastructure (typically the Internet). Therefore traffic aggregation occurs in the paths leading to a gateway and due to the limited wireless link capacity, these nodes are likely to be potential bottlenecks. In this paper, we propose a distributed load balancing algorithm to achieve load balancing on gateway nodes which leads to efficient traffic allocation as well as maximum use of network capacity. This algorithm uses Learning Automata in order to select the appropriate gateway node to send traffic. Evaluation results demonstrate that the proposed scheme largely avoids congestion and can effectively balance the traffic.**

Keywords-wirelss mesh networks; learning automata; load balancing

I. INTRODUCTION

Wireless Mesh Networks (WMNs), as a new generation of wireless multi-hop networks, draws lots of attention in recent years due to inspiring numerous applications. They provide a cost-effective way of deploying a network and providing wireless broadband internet connectivity. WMNs are formed by a set of mesh routers and mesh clients, where mesh routers have minimal mobility and form the backbone of WMNs. They provide network access for both mesh and conventional clients. A small subset of mesh routers is directly connected to a fixed infrastructure (e.g. a wired network such as Internet) called Gateway nodes. [1]

Gateways are a key component of WMNs since most traffic will be directed to/from gateways which lead to traffic aggregation in the paths leading to a gateway and therefore congestion. An important strategy is to associate nodes with a particular gateway, i.e. each node should be served by a particular gateway to send/receive traffic [2, 3].

Past works has focused mainly on finding high throughput paths in wireless multi-hop networks. To this end, various routing metrics such as ETX [4], ETT [5] and MIC [6] have been proposed. Associating nodes to the nearest gateway in terms of a routing metric (hop-count or any of the above) can lead to load imbalance. Because these metrics are load-agnostic, shortest path routing can lead to situations where gateways are subject to different load, and in a worst case where a few gateways are overloaded and others are under-utilized. Several factors can lead to this situation. For example, heterogeneous traffic demands, uneven number of nodes served by gateways (motivated for example by unplanned gateway placement, or user behavior), or gateway failure. Load imbalance can lead to inefficient use of network capacity, throughput degradation and unfairness between flows in different domains.

There are number of works concerning the problem of load-balancing in WMNs which most of them focus on balancing load between all gateway nodes whereas all gateways have similar load and most of them don’t take congestion and interference into account. WCETT-LB [7] and AODV-ST [8] allows nodes to switch to alternate gateways when their default gateway is congested. Proposed load balancing scheme in [9] have similar objectives. Most of these can suffer from route flapping or poor performance for not taking contention into account. From proposed algorithms in [10], DA and CA just consider their load and don’t take their distance to gateway nodes as well as interference with other flows into account, so perform very poorly. NGW and MLI associate each node to its nearest gateway which can result in gateways that are overloaded while others are under-utilized [10]. With MLI, every time that there is an imbalance between gateways, border nodes switch to another domain to even the load of the domains so makes high switching overhead.

Because the traffic volume in a WMN is expected to be very high, performing load-balancing in a multi-gateway WMN is important.

In this paper, a load balancing algorithm based on learning automata is proposed to balance load between gateways and therefore improving network throughput and efficient use of network capacity. In proposed algorithm, routers create routing tables based on gateways broadcasting. Then a domain of nearest gateways will be created for each router. Every time that a router wants to send data, it will chose a gateway from its domain using its learning automaton to send data. If the selected gateway is appropriate this act will be rewarded so that probability will be increased, otherwise will be penalized.

We analyze performance of our scheme with ns-2 [15, 16] and compare it with other algorithms. Results show that our scheme outperforms all other algorithms.

The rest of the paper is organized as follows. In section II, learning automata is briefly reviewed. In section III, we describe proposed algorithm. Section IV shows and analyzes protocol performance based on simulation results achieved from ns-2. Finally, in section V we summarize our conclusions.

II. Learning Automata

A LA formalizes a general stochastic system in terms of states, actions, state or action probabilities and environment responses. The design objective of an automaton is to guide the action selection at any stage of a system by past action and environment responses, so that some overall performance function is improved. At each stage, the automaton chooses a specific action from its finite action set and the environment provides a random response (see fig. 1) [11].

A variable structure stochastic automaton, is defined by a quadruple {α, β, P, T} for which α is the action or output set $\{\alpha_{1,}\alpha_2,\ldots, \alpha_{rs}\}$of the automaton, β is random variable in the interval [0,1], *P* is the action probability vector of the automaton or agent and T denotes an update scheme. The output of the automaton is actually the input to the environment. The input β of the automaton is the output of the environment, which is modeled through penalty probabilities c_i, with $c_{i\,=}\, P\,[\beta \mid \alpha_i]$, i:1,…, r.

Important examples of linear update schemes are linear reward-penalty, linear reword-inaction and linear reword-ε-penalty. The general reward- penalty algorithm is given by:

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

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

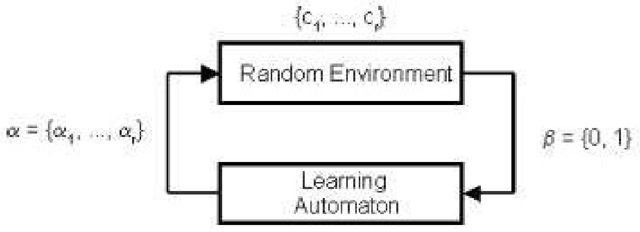


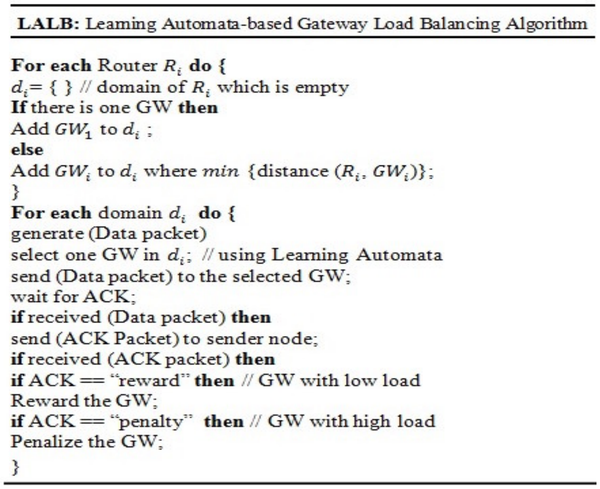
Figure 1. Automaton-environment feedback loop

The constants *a* and *b* are the reword and penalty parameters, respectively. When a=b, the algorithm is referred to as linear reward- penalty ($L_{R\text{-}P}$); when b=0, it is referred to as reward-inaction ($L_{R\text{-}I}$); and when b is small compared to a, it is called linear reward-ε- penalty ($L_{R\text{-}\varepsilon P}$).

III. Proposed Load Balancing Scheme

WMNs are comprised of mesh routers and mesh clients where mesh routers are generally static (or quasi-static). These nodes form a wireless multi-hop network and provide connectivity to mesh clients, also known as users. A subset of routers has direct connectivity to a wired network, such as Internet, and serves as gateways to the mesh routers. A set of nodes served by a gateway is called a *domain*. A user can access the Internet through one or more links leading from its router to a gateway.

In general, the proposed algorithm (LALB) associates each router with a domain of its nearest gateways. Then each router that have data to send, select one gateway from its domain using its learning automaton. If the selected gateway node is appropriate, this act will be rewarded, otherwise, will be penalized which means to increase or reduce the probability of selecting that gateway respectively. Fig. 2 shows how LALB algorithm works.



LALB: Learning Automata-based Gateway Load Balancing Algorithm

```
For each Router $R_i$ do {
$d_i$= { } // domain of $R_i$ which is empty
If there is one GW then
Add $GW_1$ to $d_i$ ;
else
Add $GW_i$ to $d_i$ where $min$ {distance ($R_i$, $GW_i$)};
}
For each domain $d_i$ do {
generate (Data packet)
select one GW in $d_i$; // using Learning Automata
send (Data packet) to the selected GW;
wait for ACK;
if received (Data packet) then
send (ACK Packet) to sender node;
if received (ACK packet) then
if ACK == "reward" then // GW with low load
Reward the GW;
if ACK == "penalty" then // GW with high load
Penalize the GW;
}
```

Figure 2. LALB Algorithm

Load Index (*LI*) parameter is introduced in order to quantify the traffic load and is defined as follows:

$$LI_{\mathrm{i}} = \frac{\sum_{\mathrm{k \epsilon N}} \mathrm{\beta k_i} \times \mathrm{T_k}}{\mathrm{C_i}} \tag{3}$$

Where LI_i refers to the load index of a particular gateway *i*. Here the factor βk_i is the fraction of node *k*'s traffic that is sent through gateway *i*. T_k is the total traffic generated by node k, and C_i is the capacity or bandwidth of the backhaul links connected to the gateway node *i*. The LI values range from zero to one where one represents a 100% loaded gateway.

LALB mainly includes 2 phases: “Domain Creation” and “Load Balancing”. In the first phase, a domain of nearest gateways will be created for each router and in the second phase, source starts sending data and routing tables at routers will be updated based on responses received from environment. The algorithm uses three packet types: Flood, Data and Ack. Flood packet is just used in the “Domain Creation” phase. Data and Ack packets are used in the second phase.

Routing tables are consisting of four fields as: 1. Gateway node number, 2. The probability of selecting it 3. Its load index and 4. Hop count to it. Each router has a learning automaton where the number of its acts is equal to the number of gateway nodes in the router’s domain. In follows, these two phases are described in details.

A. Domain Creation

Routing table of each router as well as its domain will be created during this phase so that gateway nodes broadcast the flood packets throughout the network containing three fields: sender node number, hop count and load index. Gateway node will initialize these fields before sending the flood packet so that will set the sender node number as its own number, hop count as zero and load index as its own index. Other nodes will start updating their routing tables as well as their domains by receiving flood packets as follows:

If node just receives one flood packet, will add the information as one record of its routing table and also will add it to its domain and the probability of selecting it would be set as 1.

Otherwise, it will choose at least 2 and at most 5 gateway nodes where their distance is minimum. These nodes will be added to its domain and routing table. The probability of selecting each of them can be formulated as:

$$\forall i\; i \leq N\; P_i = h \frac{\frac{1}{numhop_i}}{\sum_{i=1}^{N} \frac{1}{numhop_i}} + (1-h) \frac{LI_i}{\sum_{i=1}^{N} LI_i} \tag{4}$$

Where *i* indicates sender gateway number, LI_i, load index of *i*th gateway node and $numhop_i$ is the hop count to that node and N is the number of received packets. The *h* is a constant and ranges from zero to one. It indicates the effectiveness ratio of hop count to the load index of selected gateway node. The closer this parameter is to one, the more effective is hop count. Each node creates a learning automaton with the number of acts equals to the number of gateway nodes in its domain. If the selected act (gateway node) is appropriate, then selection probability will be increased based on learning algorithm and otherwise will be decreased. At the end of this phase, each node will have a learning automaton and a routing table to lead the Data packet to destination.

B. Load Balancing

The source will start sending Data packet to a gateway node based on its routing table and domain. The receiver sends Ack packet to the sender containing two fields: sender number and its load index. In order to deducting the number of messages and therefore load, the parameter *p* is defined. Each node will send *p* number of Data packets to the selected gateway node and gateway will just send one Ack packet by receiving *p* Data packets. By receiving Ack packet, corresponding act will be rewarded or penalized as follows:

If the load index of gateway node (sender of Ack):

- Is less than %50 of average load index of other gateways in the same domain, will be rewarded.
- Is more than %50 and less than 80% of average load index of other gateways in domain and the hop count is minimum will be rewarded.
- Is more than average load index of other gateways in domain will be penalized.

Reward and penalty can be formulated as follows respectively:

$$\alpha = a_1 + \frac{\gamma * LI_{i+}(maxnumhop - numhop_i)}{\gamma * maxLI + maxnumhop} + b_1 \tag{5}$$

$$\beta = a_2 + \frac{\gamma * (avgLI - LI_i) + numhop_i}{\gamma * avgLI + maxnumhop} + b_2 \tag{6}$$

Where LI_i indicates the load index of Ack sender and $numhop_i$ is the hop count to this gateway. *avgLI* is the average load index of gateways in domain, *.maxnumhop* is the hop count from the Ack receiver to the farthest gateway in that domain and *maxLI* is the maximum load index among gateways in a domain. The parameters a_1 and a_2 indicate the minimum value of reward and penalty respectively. γ is chosen such that these terms will be almost equal. b_1 and b_2 is chosen such that α and β don’t get very large.

IV. Simulation results

In order to evaluating the proposed algorithm, ns-2 [15, 16] is used and the goal is comparing performance of LALB with CA[1], DA[2], NGW[3] and MLI[4] [10] based on Throughput

[1] Centralized Assignment
[2] Distributed Assignment
[3] Nearest Gateway

and PDR metrics. Simulation runs for 500 seconds with 100 nodes placed in a 2000×2000 meters area. Nodes are considered static. 10 of them act as gateways and the rest as mesh routers. The MAC layer employs 802.11b with a data rate of 11Mbps. Gateways are connected to the wired network (Internet) through 100Mbps links. Network topology is connected and traffic is considered as CBR/UDP. Simulation parameters are briefly listed on Table. 1.

TABLE I. SIMULATION PARAMETERS

Parameter	Value
Simulator	ns-2
Traffic type	CBR/UDP
Simulation area	2000m × 2000m
Simulation time	500s
#of Nodes	100 (10 as GWs and the rest as routers)
Gateway placement	Fixed for all topologies
Node placement	Randomly
Topology	Connected
MAC Layer protocol	802.11b with data rate of 11Mbps
Links from GWs to Internet	100Mbps links

The parameters a_1 and a_2 in learning algorithm are set as 0.1 and b_1 and b_2 as 0.15.

Fig. 3 and 4 present the results of simulation. As we can see in Fig. 3, LALB outperforms all other schemes. CA and DA perform very poorly. This is because they just consider the load and don't take their distance to gateways into account. MLI builds upon the NGW solution, improving it. With MLI, every node is initially associated with its nearest gateway. Periodically, gateways advertise their load and every time that a node receive message from a low loaded gateway will switch to other domain. It means that MLI switches every time there is an imbalance between gateways, therefore switching overhead would be high. Another drawback of MLI is that convergence time will depend on the frequency of gateway advertisements, so it performs poorly due to the overhead of these messages. But still, on average, it improves NGW. NGW doesn't take contention into account and can result in gateways which are congested while others are not. LALB not only consider the distance to gateways but also highly prevents congestion by deducting the selection probability of high loaded gateway.

Fig. 4 shows the PDR of the protocols. Congestion builds up as the number of nodes increase which leads to the less PDR. MLI performs well until the number of nodes is less than 50 but quickly drops after that. Because it tends to balance the load between all gateway nodes and make the wrong decision while congestion happens thus leading to more packet loss.

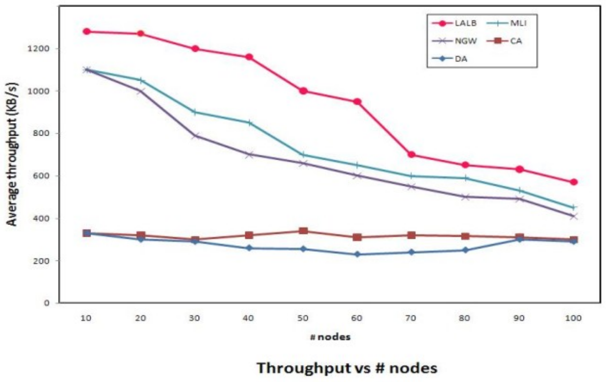


Figure 3. Throughput comparison of CA, DA, NGW, MLI, LALB

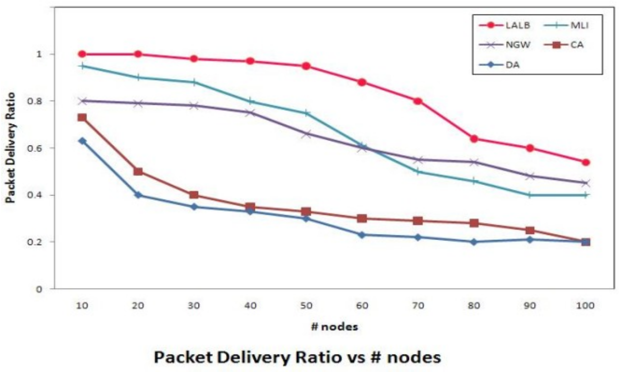


Figure 4. PDR comparison of CA, DA, NGW, MLI, LALB

V. CONCLUSION AND FUTURE WORKS

Due to the high traffic volume and the limited wireless link capacity in WMNs, whereas this traffic should be routed through gateways, gateways are likely to be potential bottlenecks. Therefore load balancing between gateways can be of crucial importance. In this paper, LALB is proposed as a distributed gateway load balancing algorithm in WMNs which performs load balancing very well and by preventing the congestion outperforms other algorithms. The simulations conducted in ns-2 prove its effectiveness.

As future works, we proposed to consider multi-radio WMNs and also consider other routing metrics [12, 13] such as ETX and ETT or load balancing routing metrics [14] instead of hop-count.

REFERENCES

[1] AL-Kharasani, N. M., Zukarnain, Z. A., "Performance Evaluation of Routing with Load-Balancing in Multi-Radio Wireless Mesh Networks", International Journal of Digital Content Technology and its Applications. Vol. 5, No. 2, 2011.

[2] Akyildiz, Ian F., "A Survey on Wireless Mesh Networks", Georgia Institute of Technology Xudong Wang, Kiyon, Inc., IEEE Radio Communications, 2005.

[3] Zhang, Y., Luo, J., Hu, H., Wireless Mesh Networking: Architectures, Protocols and Standards, Auerbach Publications, 2007.

[4] Couto, D. S. J. D., Aguayo, D., Bicket, J., and Morris, R., "A high throughput path metric for multi-hop wireless routing", in

[4] Minimum Load Index

MobiCom'03: Proc. of the 9th annual international conference on Mobile computing and networking. ACM, 2003, pp. 134–146.

[5] Draves, R., Padhye, J., and Zill, B., “Routing in multi-radio, multi-hop wireless mesh networks”, in MobiCom ’04: Proc. of conference on Mobile computing and networking. ACM, 2004, pp. 114–128.

[6] Yang, Y., Wang, J., and Kravets, R., “Interference-aware Load Balancing for Multihop Wireless Networks”, University of Illinois at UrbanaChampaign, Tech. Rep.

[7] Ma, L., and Denko, M. K., “A Routing Metric for Load-Balancing in Wireless Mesh Networks”, in Ainaw ’07, Advanced Information Networking and Applications Workshops, 2007, pp. 409–414.

[8] Ramachandran, K., Buddhikot, M., Chandranmenon, G., Miller, S., Belding-Royer, E., and Almeroth, K., “On the Design and Implementation of Infrastructure Mesh Networks”, in Proceedings of the IEEE Workshop on Wireless Mesh Networks (Wimesh), 2005.

[9] Nandiraju, D., Santhanam, L., Nandiraju, N., and Agrawal, D. P., “Achieving Load Balancing In Wireless Mesh Networks Through Multiple Gateways”, Proc. IEEE International Conference on Mobile Adhoc and Sensor Systems (Mass), 2006, pp. 807–812.

[10] Huang, C.-F., Lee, H.-W., and Tseng, Y.-C., “A Two-Tier Heterogeneous Mobile Ad Hoc Network Architecture and Its Load-Balance Routing Problem”, Mobile Networks and Applications, vol. 9, no. 4, 2004.

[11] Narendra, K.S., and Thathachar, M. A. L., “Learning Automata A Survey”, IEEE Transactions on Systems, Man and Cybernetics, vol. 4, no. 4, 1974.

[12] Nxumalo, S.L., Adigun, M.O., Mudali, P., Ntlatlapa, N., Ncanana, SM., “Comparison of Routing Metrics for Wireless Mesh Networks”, SATNAC, 2011.

[13] Zhao, Liang and Al-Dubai, Ahmed, “Routing Metrics for Wireless Mesh Networks: A Survey”, In: World Congress on Computer Science and Information Engineering, IEEE CSIE, 2011.

[14] Siraj, M., Abu Bakar, K., “A Load Balancing Interference Aware Routing Metric (LBIARM) for Multi hop Wireless Mesh Networks”, International Journal of the Physical Sciences Vol. 7(3), pp. 456 - 461, 16 January, 2012.

[15] K. Tan, D. Wu, A. Chan, P. Mohapatra, "Comparing Simulation Tools and Experimental Testbeds for Wireless Mesh Networks," Pervasive and Mobile Computing Journal, 2011.

[16] K. Fall and K. Varadham, The NS Manual, http://www.isi.edu/nsnam/ns/ns-documentation.html.