SELF-ADAPTIVE POLLING PROTOCOLS FOR WIRELESS LANS: A LEARNING-AUTOMATA-BASED APPROACH

Petros Nicopolitidis, Georgios I.Papadimitriou and Andreas S.Pomportsis

Department of Informatics, Aristotle University of Thessaloniki
Thessaloniki, Greece, Box 888 54006
e-mail: gp@csd.auth.gr

ABSTRACT: A Learning Automata-Based Polling (LEAP) protocol for infrastructure wireless LANs (WLANs), which is capable of operating efficiently under bursty traffic conditions, is proposed. In LEAP, the mobile station that grants permission to transmit is selected by the base station by means of a learning automaton. It is proved that the learning algorithm asymptotically tends to assign to each station a portion of the bandwidth proportional to the station's needs. Simulation results reveal superiority of LEAP over more complex to implement polling protocols for WLANs (RAP, GRAP) under bursty traffic conditions.

1. INTRODUCTION

Modern WLAN MAC protocols should efficiently handle the bursty traffic that is expected to be generated by WLAN applications (such as client/server and file transfer applications). In this paper, Learning Automata-Based Polling (LEAP), a polling protocol designed for bursty traffic infrastructure WLANs is proposed. According to LEAP, the mobile station that grants permission to transmit is selected by the base station by means of a learning automaton. The learning automaton takes into account the network feedback information in order to update the choice probability of each mobile station. The network feedback conveys information both on the network traffic pattern and the base-mobile station condition of the wireless links. The learning algorithm asymptotically tends to assign to each station a portion of the bandwidth proportional to the station's needs.

2. THE LEAP PROTOCOL

According to LEAP, the base station is equipped with a learning automaton which contains the choice probability $P_k(j)$ for each mobile station k under its coordination. Before polling at polling cycle j those probabilities are normalized in the following way:

$$\prod_k(j)=\frac{P_k(j)}{\sum_{i=1}^{N}P_i(j)} \qquad (1)$$

Clearly, $\sum_{i=1}^{N}\prod_i(j)=1$, where N is the number of mobile stations under the coverage of the base station. At the beginning of each polling cycle j, the base station polls according to the normalized probabilities $\prod_i(j)$. Each polling cycle consists of a sequence of packet exchanges between the base station, the selected mobile and a destination mobile station if a packet is to be transmitted by the selected mobile. The protocol uses four control packets, POLL, NO_DATA, BUFF_DATA and ACK whose duration is t_{POLL}, t_{NO_DATA}, t_{BUFF_DATA} and t_{ACK} respectively. Assuming that the base station polls mobile station k at time position t which marks the beginning of polling cycle j, the propagation delay is t_{PROP_DELAY}, and a station's DATA transmission takes t_{DATA} time to complete, the following events are possible:

1. The poll is received at station k at time $t+t_{POLL}+t_{PROP\_DELAY}$. Then:

- If station k does not have a buffered packet, it immediately responds to the base station with a NO_DATA packet. If the base station correctly receives the NO_DATA packet, it lowers the choice probability of station k and immediately proceeds to poll the next station. This poll is initiated at time $t + t_{POLL} + 2t_{PROP\_DELAY} + t_{NO\_DATA}$. In case of no reception at the base station, the choice probability of station k is lowered and the next poll begins at time $t + t_{POLL} + 4t_{PROP\_DELAY} + t_{BUFF\_DATA} + t_{DATA} + t_{ACK}$.
- If station k has a buffered DATA packet, it responds to the base station with a BUFF_DATA packet, transmits the DATA packet to its destination and waits for an acknowledgment (ACK) packet. After the poll, the base station monitors the wireless medium for a time interval equal to $t_{BUFF\_DATA} + t_{DATA} + t_{ACK} + 3t_{PROP\_DELAY}$. If it correctly receives one or more of the three packets, it concludes that station k received the poll and has one or more buffered data packets. Thus, it raises station's k choice probability. On the other hand, if the base station does not receive feedback, it concludes that it cannot communicate with station k, lowers the choice probability of k and proceeds

0-7803-7057-0/01/$10.00 ©2001 IEEE.

with the next poll at time $t + t_{POLL} + 4t_{PROP\_DELAY} + t_{BUFF\_DATA} + t_{DATA} + t_{ACK}$.

2. The poll is not received at station k, k does not respond to the base station and the choice probability of k is decreased. Then the base station proceeds to poll the next station at time $t + t_{POLL} + 4t_{PROP\_DELAY} + t_{BUFF\_DATA} + t_{DATA} + t_{ACK}$.

From the above discussion, it is obvious that the learning algorithm takes into account both the bursty nature of the traffic and the bursty appearance of errors over the wireless medium. Upon conclusion of a polling cycle j, the base station uses the following scheme in order to update the selected station's k choice probability:

$$P_k(j+1)=P_k(j)+ L_{LEAP}(1-P_k(j)),$$
$$\text{if } FEEDBACK_k(j)=TRANSMIT$$

$$P_k(j+1)=P_k(j)-L_{LEAP}(P_k(j)-a),$$
$$\text{if } FEEDBACK_k(j)=IDLE \text{ or } FEEDBACK_k(j)=FAIL \quad (2)$$

where:

- $FEEDBACK_k(j)=TRANSMIT$ indicates that the base station received feedback indicating that station k, upon polled at polling cycle j, transmitted a DATA packet. This means that the base station correctly received one or more of the BUFF_DATA, DATA, and possibly ACK, packets exchanged due to k's transmission.
- $FEEDBACK_k(j)=IDLE$ indicates that the base station received feedback indicating that station k, upon polled at polling cycle j, did not transmit a DATA packet. This means that the base station correctly received the NO_DATA packet transmitted by k.
- $FEEDBACK_k(j)=FAIL$ indicates that the base station failed to receive feedback about k's transmission state at cycle j. This is equivalent either to erroneous reception, or to the reception of no packets at all, at the base station for polling cycle j.

For all j, it holds that L_{LEAP}, $a \in (0,1)$ and $P_k(j) \in (a,1)$. Since the offered traffic is of bursty nature, when the base station realizes that the selected station had a packet to transmit, it is probable that the selected station will also have packets to transmit in the near future. Thus, its choice probability is increased. On the other hand, if the selected station notifies that it does not have buffered packets, its choice probability is decreased, since it is likely to remain in this state in the near future. In general, the background noise and interference at the base station will be the same, if not lower, than that at a mobile station. When the base station fails to receive feedback about the selected mobile's state, the latter is probably experiencing a relatively high level of background noise. In other words, it is "hearing" the base station over a link with a high BER. Since in wireless communications errors appear in bursts, the link is likely to remain in this state for the near future. Thus the choice probability of the selected station is lowered in order to reduce the chance of futile polls to this station in the near future.

When the choice probability of a station approaches zero, this station is not selected for a long period of time. During this period, it is probable that the station transits from idle to busy state. The same holds for the status of a high-BER link between the mobile station and the base station. After a period of time, it is probable that the link's state changes to a low BER one. However, since the mobile station does not grant permission to transmit, the automaton is not capable of "sensing" those transitions. The role of parameter a, is to prevent the choice probabilities of stations from taking values in the neighborhood of zero in order to increase the adaptivity of the protocol.

LEAP updates the choice probabilities of mobile stations according to the network feedback information. The choice probability of each mobile station converges to the probability that this station is ready to transmit, meaning that it has a non-empty queue and it is capable of communicating successfully with the base station. For the sake of brevity the proof is omitted.

3. SIMULATION RESULTS

Using simulation, we compared LEAP against RAP and GRAP [2, 3, 4] under bursty traffic conditions. The bursty traffic was modeled in the following way: We define "time slot" as the time duration required for a DATA packet to be transmitted over the wireless link. Each source station can be in one of two states, S_0 and S_1. When a source station is in state S_0 then it has no packet arrivals. When a source station is in state S_1 then, at each time slot, it has a packet arrival with probability Z. Given a station is in state S_0 at time slot t, the probability that this station will transit to state S_1 at the next time slot is P_{01}. The transition probability from state S_1 to state S_0 is P_{10}. It can be shown that, when the load offered to the network is R packets/slot and the mean burst length is B slots, then the transition probabilities are: $P_{01}=R/(B(NZ-R))$ and $P_{10}=1/B$. Each station uses a buffer to store the arriving packets. The buffer length is assumed to be equal to Q packets. Any packets arriving to find the buffer full are dropped.

In our simulation model, the condition of the wireless link between any two stations (including the base one) was modeled using a finite state machine with three states [1]. Stage G, denotes that the wireless link is in a relatively "clean" condition and is characterized by a small BER, which is given by the parameter *GOOD_BER*. Stage B, denotes that the wireless link is in a condition characterized by a high

BER, which is given by the parameter *BAD_BER*. Stage H, denotes that the pair of stations is out of range of one another. We assume that the background noise is the same for all stations and thus the principle of reciprocity stands for the condition of any wireless link. Therefore, for any two stations *A* and *B*, the BER of the link from *A* to *B* and the BER of the link from B to A are the same. The time periods spent by a link in states G, B and H are exponentially distributed, but with different average values, given by the parameters *TIME_GOOD*, *TIME_BAD* and *TIME_HIDDEN* respectively. The status of a link probabilistically changes between the three states. When a link has spent its time in state G and its status is about to change, the link transits either to stage H, with probability given by the parameter P_h, or to stage B, with transition probability $1\text{-}P_h$. When a link has spent its time in state B and its status is about to change, the link transits either to stage H, with probability given by the parameter P_h, or to stage G, with transition probability $1\text{-}P_h$. Finally, when a link has spent its time in state H, it transits either to state G or B, with the same probability (0.5). It can be seen that setting the parameter P_h to zero yields a fully connected network topology, whereas for values of P_h greater than zero the effect of the well-known "hidden terminal" problem on performance can be studied.

In the process of evaluating the performance of LEAP against RAP and GRAP [2, 3, 4] we used the delay versus throughput characteristic. Furthermore we made the following assumptions in our simulations:

1. Upon polled, a mobile station can initiate a data packet transmission with any other mobile as its destination. This limits the role of the base station to be only the means of executing the polling algorithms and was made to achieve a more generic and fair comparison between LEAP and RAP, GRAP.
2. We did not account for the effect of a Physical layer preamble in our simulations. This turns out to be a conservative choice (albeit of a negligible impact) for LEAP as it would slightly increase its superiority over RAP and GRAP. This is due to the higher overhead per DATA packet transmission for RAP-GRAP, as will be explained later.
3. No error correction is used and we did not take into account the packet-capturing phenomenon for RAP and GRAP. Whenever two packets collide, they are assumed lost. Capturing does not affect LEAP, since it is collision-free.

We simulated the following network configurations:

- Network N_1: N=10, Q=10, B=10, Z=1.0, P_h=0.0
- Network N2: N=10, Q=3, B=200, Z=0.7, P_h=0.0
- Network N3: N=10, Q=10, B=10, Z=1.0, P_h=0.2
- Network N4: N=10, Q=3, B=200, Z=0.7, P_h=0.2

These configurations were simulated twice, once for a value of BAD_BER equal to 10^{-6} and once for a value of BAD_BER equal to 10^{-3}. Moreover, *GOOD_BER*=10^{-10}, *TIME_GOOD*= 30 sec, *TIME_BAD*=10 sec, *TIME_HIDDEN*=5 sec, L_{RAP}=2, P_{RAP}=5 [3], *RETRY_LIMIT*=6. *RETRY_LIMIT* sets the maximum number of retransmission attempts per packet. Finally, the size of all control packets for the three protocols at the MAC layer is set to 160 bits, the DATA packet size is set to 6400 bits and the overhead for the orthogonal transmission of the random addresses in RAP and GRAP is set to five times the size of the poll packet, as in [3]. The wireless medium bit rate was set to 1Mbps. The propagation delay between any two stations was set to 0.05 msec.

Figures 1-4 contain a subset of the simulation results we have obtained. LEAP outperforms RAP and GRAP in all cases, since it is collision-free and the per-DATA packet transmission overhead of the protocol is less. LEAP requires an overhead of three control packets per DATA packet (POLL, BUFF_DATA, ACK). RAP and GRAP on the other hand can transmit at most five DATA packets per polling cycle for five random addresses (P_{RAP}=5), assuming no collisions occur, with an overhead of sixteen control packets, for L_{RAP}=1, (READY, orthogonal transmission of random addresses which is equal to five times the duration of a control packet, five POLL packets, five ACK packets) yielding an overhead of 3.2 control packets per DATA packet. However, this scenario seldomly occurs in practice due to the increased number of occurring collisions and the resulting instability of RAP when the number of active station per polling cycle approaches the number of random addresses, P_{RAP}. Moreover under heavy bursty traffic conditions the number of active stations per polling cycle is significantly less than the number of random addresses P_{RAP} resulting in increased per-DATA packet overhead for RAP and GRAP. These facts are also supported by results of simulation experiments, which are not presented here for brevity.

4.CONCLUSION

Modern WLAN MAC protocols should be able to efficiently handle the bursty traffic that is expected to be generated by WLAN applications. This paper proposes Learning Automata-Based Polling (LEAP), a polling protocol designed for bursty traffic WLANs. According to LEAP, the mobile station that grants permission to transmit is selected by the base station by means of a learning automaton. This mechanism uses the network feedback information in order to adapt to the bursty behavior both of the offered traffic and the wireless channel errors. The protocol is able to achieve significantly higher throughput and lower delay values compared to RAP and GRAP under bursty

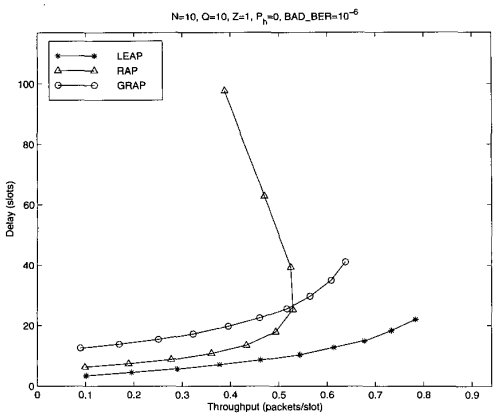

Figure 1. The Delay versus Throughput characteristics of LEAP, RAP and GRAP when applied to network N_1. *BAD_BER* is set to 10^{-6}.

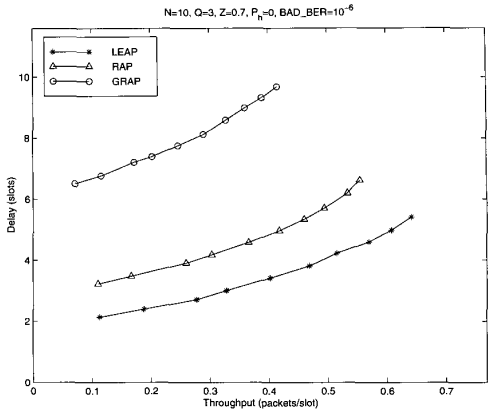

Figure 2. The Delay versus Throughput characteristics of LEAP, RAP and GRAP when applied to network N_2. *BAD_BER* is set to 10^{-6}.

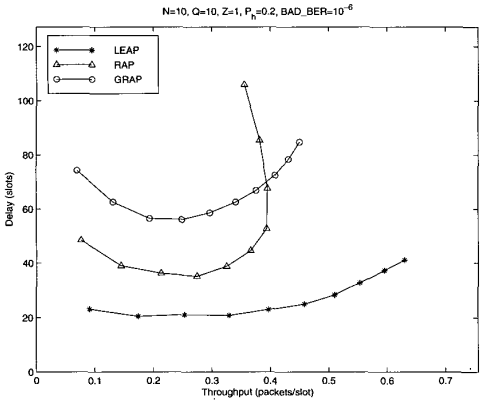

Figure 3. The Delay versus Throughput characteristics of LEAP, RAP and GRAP when applied to network N_3. *BAD_BER* is set to 10^{-6}.

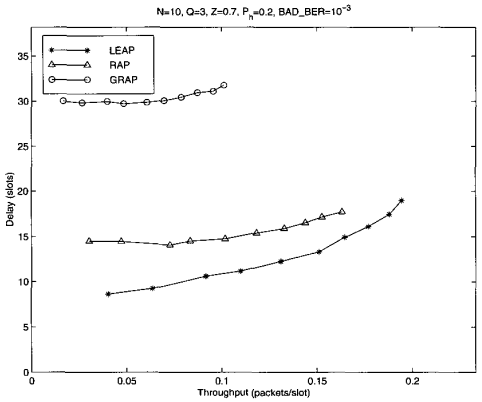

Figure 4. The Delay versus Throughput characteristics of LEAP, RAP and GRAP when applied to network N_4. *BAD_BER* is set to 10^{-3}.

traffic conditions in wireless environments. The main characteristics of LEAP are:

1. It achieves a high performance, even when the offered traffic is bursty.
2. It is self-adaptive. Each station is assigned a fraction of the bandwidth proportional to its needs. Furthermore, LEAP explicitly takes into consideration the condition of the wireless medium between the base station and the mobiles in an effort to prevent possibly futile polls over links with a high BER.
3. It is very simple to implement, much more simple than RAP and GRAP. The only requirement is the existence at the base station of a processor, which implements the learning algorithm. On the other hand, RAP and GRAP call for extra hardware both at the base station and the mobiles for the orthogonal transmission and reception of the random addresses.

5. REFERENCES

[1] E.Gilbert, "Capacity of a burst noise channel", Bell System Technology Journal, vol. 39, September 1960.

[2] K.C.Chen, "Medium Access Control of Wireless LANs for Mobile Computing", IEEE Network, September/ October 1994.

[3] K.C.Chen, C.H.Lee, "RAP-A Novel Medium Access Control Protocol for Wireless Data Networks", Proceedings of IEEE GLOBECOM, 1993.

[4] K.C.Chen, C.H.Lee, "Group Randomly Addressed Polling for Multicell Wireless Data Networks", Proceedings of IEEE ICC, 1994.

[5] G.I.Papadimitriou and A.S Pomportsis, "Learning Automata-Based TDMA protocols for Broadcast Communication Systems with Bursty Traffic", IEEE Communication Letters, March 2000.