A Self-Adaptive Protocol for Broadcast LAN's With Variable Packet Length

Georgios I. Papadimitriou, *Senior Member, IEEE*, Georgios D. Pallas, *Student Member, IEEE*, and Andreas S. Pomportsis, *Member, IEEE*

***Abstract*—An adaptive medium access control (MAC) protocol for broadcast networks is introduced, which can operate efficiently under highly bursty traffic conditions and can handle stations transmitting packets of arbitrary length. The proposed Variable Packet Length Adaptive Protocol (VPLAP) belongs to the family of learning-automata-based protocols, which resolves most drawbacks that fixed-assignment protocols like time-division multiple access (TDMA) and RTDMA have. LTDMA, although belonging to this same family, suffers from reduced channel usage, wasted timeslots as well as network equipment overloading due to its fixed-sized timeslots. VPLAP, on the other hand, is tested using real LAN traffic traces and is proved not only to be successfully resolving the above issues but also being able to achieve a high throughput-delay performance when operating under realistic traffic conditions.**

***Index Terms*—Broadcast LANs, bursty traffic, learning automata, time-division multiple access (TDMA), variable packet length.**

I. Introduction

BROADCAST networks present us with the key issue of who gets to use the common channel. Protocols like LTDMA [1] came up with a way to dynamically assign a fraction of the available bandwidth to each station proportionally to its needs. However, LTDMA suffers from a couple of serious weaknesses.

a) Since the timeslots are fix-sized, the stations are only allowed to transmit fix-sized packets. This results in reduced performance due to stations transmitting dummy padding bits as well as overhead for stations and network components, since packets have to be properly fragmented prior to transmission.
b) In case an idle station is selected to transmit, a whole timeslot is wasted.

In this letter, a new protocol resolving the above issues is introduced. Variable Packet Length Adaptive Protocol (VPLAP) is a protocol for broadcast networks based on learning automata [2]–[5] which allows stations to transmit variable length packets and operates efficiently even under heavy, bursty traffic conditions. VPLAP uses a station-to-transmit scheme which adjusts the frequency with which each station is selected, according to the probability it has to be ready. A penalty system is also applied to prevent stations transmitting large packets from monopolizing the bandwidth utilization. This way, the proper bandwidth is assigned to each station according to its needs, and the resulting throughput converges to the offered load.

This letter is organized as follows. In Section II the main concepts of the proposed protocol, VPLAP, are presented. In Section III the inner workings of VPLAP's learning algorithm are given, followed by Section IV which introduces and describes the necessity of the "idle timeslot" parameter. The next section describes how new stations connecting to the network synchronize their state and find out the number of the other stations, as well as how stations which have long been idle are signed out of the protocol. Section VI presents the simulation results for VPLAP compared to other TDMA protocols like TDMA [6]–[10], RTDMA [11] and a limited-contention protocol, URN [12]. Finally the major characteristics of VPLAP are summarized in Section VII.

II. The VPLAP Protocol

One of the main concepts that VPLAP is based on, is that a station currently transmitting packets will probably continue to transmit packets in the near future. This is justified by the fact that traffic generated by computer networks is highly bursty [13].

VPLAP takes advantage of this fact by implementing a learning algorithm for selecting stations, which assigns different choice probabilities to each station according to their needs: Idle stations are assigned a low choice probability, whereas stations which send traffic across the network have high choice probabilities.

For example, each time that an idle station is selected, its choice probability will be decreased. Therefore, during the next timeslots, this station will have reduced probability to be selected, resulting in fewer wasted timeslots. At the same time, the bandwidth saved, will be allocated to stations that will make use of it.

The proposed protocol is based on the broadcast nature of the network, which allows all stations to receive the same feedback information and update their automata accordingly. All stations always select the same station for transmission thanks to a pseudorandom number generator, seeded with a common initial value.

We should observe that although VPLAP is completely decentralized and requires no central coordination between the stations, it is collision free, thus achieving a high performance, as the following sections will show.

Manuscript received May 30, 2003. The associate editor coordinating the review of this letter and approving it for publication was Prof. M. Devetsikiotis.

The authors are with the Department of Informatics, Aristotle University, 54124 Thessaloniki, Greece (e-mail: gp@csd.auth.gr).

Digital Object Identifier 10.1109/LCOMM.2003.822504

III. The VPLAP Learning Algorithm

In VPLAP, each station is provided with a learning automaton containing the basic choice probabilities $\mathrm{P_i}$ of all stations. Before the protocol selects a new station to transmit, the choice probabilities are normalized. The station selection scheme is based on a common pseudorandom number generator seeded with a common initial value, so that all stations select the same station to transmit according to the choice probabilities, without need for centralized coordination.

Each time a station that was selected to transmit has an empty packet buffer, its choice probability is decreased because it is probable that it will also remain idle in the near future. Accordingly, a station which successfully transmits a packet, increases its choice probability. Below follow the functions, which control the choice probability changes:

$$\begin{aligned} &\mathrm{P_i} \leftarrow \mathrm{P_i} + \mathrm{L}(1-\mathrm{P_i}), \quad \text{if station } \mathrm{u_i} \text{ transmitted a packet} \\ &\mathrm{P_i} \leftarrow \mathrm{P_i} - \mathrm{L}(\mathrm{P_i}-\mathrm{a}), \quad \text{if station } \mathrm{u_i} \text{ did not transmit a packet,} \\ &\text{where } \mathrm{L},\ \mathrm{a} \in (0,1) \end{aligned}$$

In order for the protocol to retain the property of matching the offered load with the resulting throughput and thus share the available bandwidth to the stations according to their demands, VPLAP keeps track of $\mathrm{ttb_i} = \text{total transmitted bytes}$ for each station $\mathrm{u_i}$. VPLAP then imposes a penalty (decreases the choice probability using the above formula) on stations for which $\mathrm{ttb_i}$ exceeds the ratio in (1).

$$\mathbf{ttb_i} \rightarrow \frac{\sum_{i=1}^{N} ttb_i}{N} \tag{1}$$

This way, the algorithm succeeds in making all active stations to converge the total size of the frames they have transmitted to the ratio in (1). Therefore, the resulting throughput tends to match the offered load. The result is that no station transmitting large packets can monopolize the bandwidth. The gain in performance is significant because stations with network applications which require interaction and thus small response delays will be selected very frequently, whereas other stations will be able to transmit large data packets while being selected with a lower frequency.

The two parameters "L" and "a" which were mentioned above, demand further explanation. It is obvious from the above choice probability updating scheme that as "L" increases, the choice probability convergence gets quicker. Since the traffic on a local area network is bursty, it is reasonable to set L to a high enough value (about 0.9), so that when a station turns idle, its choice probability quickly decreases and, as a result, no further timeslots are wasted. After several failures of a station to transmit a packet, its choice probability would fall to zero (and therefore not be selected anymore) if it were not for the parameter "a" which, having a near-zero value lets the automata select the station even after many failures, in order to sense a transition to an active state of this station in the future.

Parameter "a" is determined with two things in mind: a) how to achieve the lowest possible loss in timeslots due to stations that get selected but are idle and b) how to achieve the lowest possible delay when a station turns active again before it gets selected and can transmit packets.

These two requirements are directly conflicting: If we set the parameter "a" to a relatively high value (in order to reduce the delay), then the stations which remain idle will get selected very frequently since their choice probability will not drop below "a", thus wasting a significant fraction of the bandwidth. If we set parameter "a" to zero, then stations which remain idle for sometime, will have their choice probability dropped to zero and will therefore not be able to transmit packets again (they should then wait for the join period to join again).

By simulating various traffic patterns (from the real traffic traces that we collected) using different values for the parameter "a", we concluded that first, the bursty traffic model is very insensitive to small changes of the parameter "a" due to the largely abrupt changes in traffic patterns, and second, that a value around a narrow neighborhood of 0.05 represents the best tradeoff between cases a) and b) as they were previously mentioned.

IV. The Idle Timeslot Parameter

In VPLAP, since the stations transmit packets of arbitrary size, each transmission completes in a variable time period, which is not already known to the other stations. That means that a station B only understands the transmission completion of station A, just after the last bit transmitted from station A reaches B. Let us call the time needed for a single bit to traverse the distance between the most remote points of the LAN an "idle timeslot".

There is no problem as long as all stations transmit packets, but when a station does not have a packet to transmit, it has to remain silent for a time period equal to one "idle timeslot" in order for all stations to understand that no packet was transmitted and, consequently, to update their automata accordingly (decrease the station's choice probability).

Inevitably, the idle timeslot introduces an overhead to the protocol, so it is of major importance to minimize its size without interfering with the normal function of the protocol. Supposing that our network is based on copper wiring and taking into consideration the data transmission speed on copper, we find that one single bit needs about 5 μs to traverse a LAN of 1 km. So, according to what was said above, the idle timeslot should be at least 5 μs for such a LAN or else the automata would lose their synchronization.

V. VPLAP's Station Join Period

During the protocol's normal function, every k seconds, the station that has just been selected to transmit, transmits a Call For Join (CFJ) packet that signifies the beginning of the station join period.

During this period, any new stations wanting to connect to the LAN, send their join packets, declaring that after the join period they will become active. The CFJ packet contains the learning automaton with all the choice probabilities of the stations and also the stations that were signed out due to long-term inactivity. This way, new stations connecting to the LAN fully synchronize

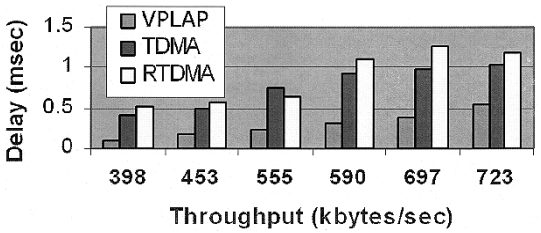

Fig. 1. Delay versus throughput characteristics of VPLAP, TDMA and RTDMA (idle timeslot 5 μs).

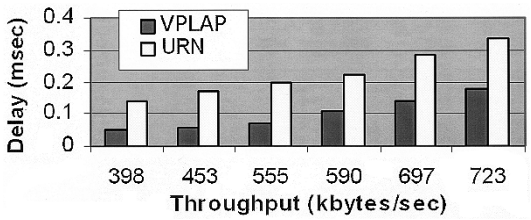

Fig. 2. Delay vsersus throughput characteristics of VPLAP and URN (idle timeslot 5 μs).

their state with the rest of the stations and are ready to transmit packets after the join period.

The learning automata that stations have to keep only contain active stations, since stations which were physically disconnected from the LAN will be rejected from the automata after a certain number of times that the station will be selected and will (obviously) remain idle. Those stations, if again connected to the LAN, will have to wait for a join period to start transmitting packets.

The join period ends after a certain amount of time has passed (an idle timeslot, see Section IV), without any new join packets having been sent. During the join period, if a collision between CFJ packets occurs, the stations retransmit their join packets after a random delay (smaller than an idle timeslot).

VI. Simulation Results

This paragraph sums up the results from the simulations which were conducted in order to compare VPLAP with TDMA [6]–[10] and RTDMA [11], as well as URN [12] which is a limited contention protocol. Since these protocols use fix-sized timeslots, they were converted in order to be able to handle packets of variable size.

For the needs of the simulations, real traffic traces from a 10 Mbps LAN of our university were gathered using the unix tool "tcpdump" [14]. Instead of using a data model producing bursty traffic, we preferred the real traffic traces in order to have a more realistic view upon the new protocol's characteristics.

Fig. 1, shows that in a wide range of throughputs, the delay transmission times that VPLAP achieves, are only a small fraction of the corresponding times for TDMA and RTDMA, thus proving that the available bandwidth is very efficiently managed. In Fig. 2 we can see that VPLAP outperforms even URN which is known to be better than both TDMA and RTDMA, and constantly achieves better (lower) response times.

VII. Conclusion

A new MAC protocol, which is capable of handling packets of arbitrary size, has been presented. The proposed protocol uses a dynamic bandwidth allocation scheme that assigns a fraction of the available bandwidth to each station, proportional to its needs.

References

[1] G. I. Papadimitriou and A. S. Pomportsis, "Learning-automata-based TDMA protocols for broadcast communication systems with bursty traffic," *IEEE Commun. Lett.*, vol. 4, pp. 107–109, Mar. 2000.

[2] P. Nicopolitidis, G. I. Papadimitriou, and A. S. Pomportsis, "Using learning automata for adaptive push-based data broadcasting in asymmetric wireless environments," *IEEE Trans. Veh. Technol.*, vol. 51, pp. 1652–1660, Nov. 2002.

[3] G. I. Papadimitriou and A. S. Pomportsis, "Self-adaptive TDMA protocols for WDM star networks: a learning-automata-based approach," *IEEE Photon. Technol. Lett.*, vol. 11, pp. 1322–1324, Oct. 1999.

[4] G. I. Papadimitriou and D. G. Maritsas, "Learning automata-based receiver conflict avoidance algorithms for WDM broadcast-and-select star networks," *IEEE/ACM Trans. Networking*, vol. 4, pp. 407–412, June 1996.

[5] K. Najim and A. S. Poznyak, *Learning Automata: Theory and Applications*. New York: Pergamon, 1994.

[6] J. M. Capone and I. Stavrakakis, "Delivering QoS requirements to traffic with diverse tolerances in a TDMA environment," *IEEE/ACM Trans. Networking*, vol. 7, pp. 75–87, Feb. 1999.

[7] W. Stallings, *Local and Metropolitan Area Networks*. Englewood Cliffs, NJ: Prentice-Hall, 1997.

[8] I. Rubin and J. Baker, "Media access control for high speed local area and metropolitan networks," *Proc. IEEE*, vol. 78, pp. 168–203, Jan. 1990.

[9] G. Anastasi, D. Grillo, and L. Lenzini, "An access protocol for speech/data/video integration in TDMA-based advanced mobile systems," *IEEE J. Select. Areas Commun.*, vol. 15, pp. 1498–1509, Oct. 1997.

[10] I. Rubin, "Access control disciplines for multiaccess communications channels: reservation and TDMA schemes," *IEEE Trans. Inform. Theory*, vol. 25, pp. 516–526, 1979.

[11] A. Ganz and Z. Koren, "Performance and design evaluation of WDM stars," *J. Lightwave Technol.*, vol. 11, pp. 358–366, Feb 1993.

[12] L. Kleinrock and Y. Yemini, "An optimal, adaptive scheme for multiple access broadcast communication," in *Proc. IEEE ICC'78*, Toronto, Canada, pp. 7.2.1–7.2.5.

[13] R. Jain and S. Routhier, "Packet trains—measurements and a new model for computer network traffic," *IEEE J. Select. Areas Commun.*, vol. 4, pp. 986–995, Sept. 1986.

[14] Linux.org documentation team. Tcpdump Documentation and Applications. [Online] Available: http://www.linux.org