Article:

Lagkas, T.D. orcid.org/0000-0002-0749-9794, Papadimitriou, G.I. and Pomportsis, A.S. (2006) QAP: A QoS supportive adaptive polling protocol for wireless LANs. Computer Communications, 29 (5). pp. 618-633. ISSN 0140-3664

https://doi.org/10.1016/j.comcom.2005.05.001

QAP: A QoS Supportive Adaptive Polling Protocol for Wireless LANs

T. D. Lagkas, G. I. Papadimitriou, *Senior Member*, *IEEE*, and A. S. Pomportsis

Department of Informatics, Aristotle University, Box 888, 54124 Thessaloniki, Greece.
E-mail: g.i.papadimitriou@ieee.org

Abstract

A QoS supportive Adaptive Polling (QAP) protocol for wireless LANs is introduced. QAP operates under an infrastructure wireless LAN, where an Access Point (AP) polls the wireless nodes in order to grant them permission to transmit. The polled node sends data directly to the destination node. We consider bursty traffic conditions, under which the protocol operates efficiently. The polling scheme is based on an adaptive algorithm according to which it is most likely that an active node is polled. Also, QAP takes into account packet priorities, so it supports QoS by means of the Highest Priority First packet buffer discipline and the priority distinctive polling scheme. Lastly, the protocol combines efficiency and fairness, since it prohibits a single node to dominate the medium permanently. QAP is compared to the efficient learning automata-based polling (LEAP) protocol, and is shown to have superior performance.

Keywords

Wireless LANs, Adaptive Polling, Quality of Service, QoS Supportive Adaptive Polling.

1. Introduction

Lately, there has been a great interest in the wireless communication networks which support high quality services and combine asynchronous communication, such as file transfer, and time bounded communication, such as streaming video. In general, the wireless networks have some special characteristics, which make the design of an appropriate medium access control protocol

This work appears in Elsevier Computer Communications, Volume 29, Issue 5, pp. 618-633, March 2006. DOI: 10.1016/j.comcom.2005.05.001

rather difficult [1]–[4]. Specifically, the bit-error rate (BER) is significantly increased in comparison to the BER of a LAN cable. The increased BER is due to the noise, the signal propagation and interference, and the node mobility, which meet in the wireless topology. Also, the “hidden nodes” phenomenon does not allow us to consider constant links between the nodes, since they are mobile and the status of every link changes dynamically. The conclusion is that in a wireless network the links are not reliable, the bit-errors are more often, and the topology changes in a continuous way. Furthermore, a modern wireless network needs QoS support.

It has been shown that the nature of the traffic in an ordinary computer network is bursty. So, the WLAN protocol must be capable of operating efficiently under bursty traffic conditions. In this paper, we propose QoS supportive Adaptive Polling (QAP), a new WLAN protocol designed for bursty traffic that supports QoS. An adaptive polling algorithm tends to poll the nodes, which are actually active, without having direct feedback about their current status [5]. An infrastructure WLAN topology is considered, where there is an access point (AP) that is only responsible for polling the mobile nodes in order to give permission to transmit. This network topology is depicted in Fig. 1, where it is also shown that our simulation environment assumes communication of every node (k) with its previous (k-1) and its next ($k+1$) node. The adaptive polling algorithm takes into account the priorities of the data packets that are broadcasted by the mobile nodes, in order to decide which node to poll [6]. Furthermore, every node implements a Highest Priority First (HPF) packet buffer discipline, which contributes in the QoS support. It is shown that the introduced protocol manages bandwidth assignment in an effective and fair way.

The paper is organized as follows. Section 2 discusses other WLAN MAC protocols, presents a classification of them, and emphasizes on the learning automata-based polling (LEAP) protocol. In Section 3, the QAP protocol is analyzed, and specifically the polling scheme is examined, the priority model of QAP is presented, and the node choice mechanism is approached in analytical way. Section 4 presents the simulation environment and the simulation results, which show the performance superiority of the QAP protocol, comparing the proposed protocol and the LEAP protocol. Also, the QoS support of QAP is revealed. Section 5 concludes the paper and gives some general guidelines for further research.

2. WLAN MAC Protocols

Any wireless network demands the presence of a set of rules that moderate the access to the shared medium. Medium access control (MAC) protocols therefore play a crucial role by ensuring efficient and fair sharing of the limited wireless bandwidth. They have been studied extensively for almost three decades. There are various MAC protocols proposed in the literature for different kinds of network and traffic conditions [7]. In this section, the WLAN MAC protocols are classified and the most characteristic are presented.

2.1. Classification of WLAN MAC Protocols

A basic classification of the WLAN MAC protocols is the one that takes into account the assumed network architecture. They can be divided into two classes: distributed and centralized [8]–[12]. This classification is depicted in Fig. 2.

2.1.1. Distributed Protocols

The distributed WLAN MAC protocols assume an ad hoc network architecture, where all the wireless nodes communicate with one another with no pre-existing infrastructure. There is no admission control, so the medium access mechanism works in a distributed manner. The corresponded protocols are mainly based on carrier sensing and collision avoidance.

A representative distributed WLAN MAC protocol is the Distributed Foundation Wireless MAC (DFWMAC) protocol, which is the basic access protocol in the IEEE 802.11 standard. It is based on the CSMA/CA (Collision Avoidance) mechanism. An RTS-CTS handshake is used to avoid collisions in the regions of the transmitter and the receiver. Many variants of this protocol are proposed in the literature. For example, multiple access with collision avoidance

(MACA) was proposed to provide the ability to perform per-frame transmission power control. MACAW (MACA with CW optimization) has been proposed to extend MACA by adding link level ACKs and a less aggressive backoff policy. Also, the EDCA protocol used in the IEEE 802.11e standard [13] enriches the DFWMAC (also called DCF in the IEEE 802.11 standard) by providing QoS support. Lastly, The elimination yield – non – preemptive priority multiple access (EY-NPMA) protocol is worth mentioning. It is the channel access protocol used in the HIPERLAN system, which was standardized by ETSI (European Telecommunications Standards Institute). The specific WLAN MAC protocol uses active signaling in order to avert simultaneous data transmissions.

Concerning the distributed wireless protocols in general, their great advantage is the ability to operate satisfactory in environments where no infrastructure is present. However, this kind of protocols assume a stochastic operation where there are no transmission guarantees. The fact that they exhibit a significant collision rate and phenomena like the “hidden” and the “exposed” nodes, are responsible for their relatively poor and unstable performance. Specifically, the performance metrics (throughput, delay, jitter, fairness etc.) of the distributed WLAN MAC protocols show that they are insufficient to support demanding applications, like real time voice and video transmission or video on demand.

2.1.2. Centralized Protocols

Centralized wireless networks are usually extensions to wireline networks. They have a base station (this work refers to it as “Access Point”) which acts as the interface between wireless and wireline networks. In this type of networks, the existence of the AP denotes that some infrastructure is required. Specifically, there is a cellular topology, where the AP decides which mobile node has the permission to transmit. This kind of centralized access control, gives the AP the ability to schedule the transmissions in order to provide QoS. The duplexing mechanism used by a centralized WLAN is also worth mentioning. Most of the protocols define that the AP acts as the packet forwarder among the nodes of the cell. Time division duplex (TDD) refers to multiplexing of the transmission (uplink channel) and reception (downlink channel) in different time periods in the same frequency band. Using different frequency bands for the uplink and downlink is called the frequency division duplex (FDD) mode of operation. In FDD it is feasible for the node to transmit and receive data at the same time: this is not possible in TDD. Although the centralized WLAN MAC protocols could be further classified according to their mode of operation, here we distinguish between the protocols that provide random access and those that require bandwidth reservation.

Idle sense multiple access (ISMA) is a random access protocol for centralized wireless networks. According to it, when the medium is idle, the AP broadcasts an idle signal. All nodes that have data to send transmit with the same probability. If two or more nodes transmit, a collision results, that is the reason why the protocol exhibits a great collision probability. Reservation ISMA (R-ISMA) and slotted ISMA (S-ISMA) are variations of the original protocol that improve its performance. Resource auction multiple access (RAMA) is also a random access protocol that achieves resource assignment using a deterministic access algorithm. It is quite unfair, as the node with the highest ID always wins the contention and captures the channel. F-RAMA (Fair RAMA) is a version of the RAMA protocol which tries to address the fairness problem. The random address polling (RAP) protocol and its improved variant the GRAP (Group RAP) protocol are maybe the most representative centralized random access WLAN MAC protocols. Lastly, the learning automata-based polling (LEAP) protocol belongs to the class of the random access wireless protocols, too. The RAP, GRAP, and LEAP protocols will be presented in more detail in the next two subsections, since they are considered for the analysis of the proposed QAP protocol. Generally, the random access wireless protocols provide high medium utilization, since the number of collisions are decreased or even nullified. They can provide QoS by scheduling the transmissions according to packet-node priorities. They are not able to provide transmission guarantees at the degree that the reservation protocols do, but they minimize the overhead regarding the feedback the AP demands from the nodes.

The last class of the centralized protocols, the reservation protocols, can be further classified into random reservation and demand assignment protocols. Every random reservation protocol has two components: random access and reservation. All nodes that have data to transmit use a random access mechanism to make their first transmission in order to reserve uplink bandwidth for the following data transmissions. In PRMA (Packet Reservation Multiple Access) the transmission of voice packets requires reservation of uplink slots, while no reservation is made for a data transmission. It has been shown that the introduction of data traffic in a voice-only system decreases the performance of PRMA. Different versions of the PRMA protocol have been proposed to improve its performance. The FRMA (Frame Reservation Multiple Access) and the PRMA++ protocols operate more efficiently by separating the different kinds of traffic. Centralized PRMA (C-PRMA) uses scheduling to give QoS guarantees. The RRA-ISA (Random Reservation Access – Independent Stations Algorithm) protocol proposes a different access policy, which tries to distribute access rights among nodes so as to maximize the throughput. It exhibits improved efficiency compared to the PRMA protocols family.

Demand assignment protocols also belong to the reservation protocols, however they try to allocate bandwidth to nodes according to their QoS requirements. The phases usually assumed by a demand assignment protocol are: request, scheduling, and data transmission. In DQRUMA (Distributed-Queuing Request Update Multiple Access) the uplink consists of a request channel, used to send contention requests, and a data channel, used to send data. The downlink slot is responsible for acknowledging the contention requests, granting transmit-permission and carrying data to the nodes. It has been shown that DQRUMA exhibits good performance with guaranteed bandwidth and minimum delay scheduling. MASCARA (Mobile Access Scheme based on Contention and Reservation for ATM) uses variable-length time frame, which consist of three periods: broadcast, reservation, and contention. The broadcast period contains control information from the AP to the nodes, the reservation period consist of the uplink and downlink data transmissions, and the contention period is used to send requests to the AP. MASCARA shows relatively high delays because of the TDD operation mode. DSA++ (Dynamic Slot Assignment ++) schedules the data transmissions using a heuristic algorithm. This algorithm prioritizes the requests and assigns the next slot to the node with the highest priority. The recently defined HCCA (HCF Controlled Channel Access) protocol in the IEEE 802.11e standard [13] is also worth mentioning. It is actually a part of the HCF (Hybrid Coordination Function) protocol, proposed for QoS supportive WLANs. In HCCA, the AP (controlled by the Hybrid Coordinator) assigns transmission time periods to the nodes, according to their QoS requests. This way, it manages to guarantee the traffic specific requirements.

At this point, it must be mentioned that the proposed QAP protocol is a centralized random access WLAN MAC protocol, so it should be compared with protocols of the same class. RAP and GRAP are representative protocols of this category and they are clearly based on the principles of the random access polling that characterizes the specific class of protocols. LEAP is also a random access protocol. It outperforms RAP and GRAP and it has been shown that it exhibits high performance. It operates efficiently under bursty traffic conditions, it provides zero collisions, and it assumes direct communication between the mobile nodes. Since the proposed QAP protocol adopts these features, it is reasonable to be compared with LEAP in order to demonstrate its performance. Lastly, it should be noticed that this kind of protocols combine simplicity, flexibility, and adaptiveness with deterministic operation, reliability, and stability.

2.2. The RAP, GRAP Protocols

All the wireless MAC polling protocols try to reduce the wrong polls (polls to inactive nodes), the overhead, and the collisions among the nodes. The Randomly Addressed Polling (RAP) protocol provides zero wrong polls, but it gives a rather increased overhead and high collision probability [14]. According to this protocol, there is no direct communication between the mobile nodes; instead the AP forwards all the packets to their destinations. Initially, the AP informs the nodes that it is ready to collect packets. Then, the active nodes (nodes that have packets to send) transmit to the AP a number of random addresses using orthogonal transmission (CDMA or FDMA). The AP collects these addresses and selects one of them in order to poll the node that sent it. The problem is that the addresses are randomly selected from a pre-defined set of numbers; so more than one mobile nodes can select and transmit the same random address. This leads to polling more than one nodes, which transmit their packets simultaneously, so they collide. If the AP successfully receives a data packet, it sends an acknowledgement (ACK).

Apart from the high collision probability, RAP supports no QoS at all. GRAP is an improvement of RAP [15]. It uses super-frames and divides active nodes to groups. This protocol does not allow all the active nodes to compete at the same time, so it reduces the number of collisions. After the formation of the node-groups, the polling procedure, according to RAP, begins for all nodes inside each group. GRAP provides a minimum QoS support by allowing the nodes that carry time bounded packets to join any group for contention. This protocol performs better than the original RAP protocol, but the provided throughput and packet delay are still not satisfactory.

2.3. The LEAP Protocol

The LEAP protocol is also a wireless polling protocol, but it is based on a different concept [16]. It assumes a cellular topology as it was described above, however it considers direct communication between the mobile nodes (the AP is not a packet forwarder). This protocol defines that the AP chooses the node that will be given permission to transmit by using choice probabilities. Four control packets are used: POLL, NO_DATA, BUFF_DATA, and ACK, with duration t_{POLL}, t_{NO_DATA}, t_{BUFF_DATA}, and t_{ACK}, respectively. The propagation delay is t_{PROP_DELAY}, and a data packet transmission lasts t_{DATA}. According to its polling scheme, the maximum polling cycle duration is $t_{POLL} + t_{BUFF\_DATA} + t_{DATA} + t_{ACK} + 4t_{PROP\_DELAY}$.

When the AP detects that the polled node transmits data then it is assumed that it is active and has more packets to send, so its choice probability is increased. Respectively, when the polled node responds with a NO_DATA packet or the AP fails to receive feedback, then it is assumed that the node will remain inactive for a short period of time or that there is a bad link, so the node's choice probability is decreased [17]. According then to this protocol, AP examines the feedback that gets during a polling cycle (*j*) in order to update the choice probabilities and select the node to poll at the next polling cycle (*j + 1*). When the choice probability of node *k* is increased, it becomes $P_k(j+1) = P_k(j) + L(1 - P_k(j))$, and when it is decreased it becomes $P_k(j+1) = P_k(j) + L(P_k(j) - a)$, where *L*, *a* are constants.

At each polling cycle *j*, the basic choice probabilities P_k for each mobile node *k* are normalized in the following way: $\Pi_k(j) = P_k(j) \big/ \sum_{i=1}^{N} P_i(j)$. Obviously, $\sum_{k=1}^{N} \Pi_k(j) = 1$, where *N* is the number of the mobile nodes in the cell. In the beginning of each polling cycle, the AP polls mobile nodes according to the normalized probabilities $\Pi_k(j)$. For all *j*, it holds $L, a \in (0, 1)$, and $P_k(j) \in (a, 1)$. *L* governs the speed of the automation convergence, and *a* enhances adaptiveness to the protocol, by not allowing the choice probability to get the value *0*.

LEAP is an efficient WLAN protocol and performs clearly better than RAP and GRAP. It provides higher throughput, and lower packet delay and packet loss rate. The main drawback of the protocol, which is rather important, is the lack of QoS support.

3. The QAP Protocol

3.1. The Polling Scheme

The network topology assumed by QAP is a cellular one where the AP polls the nodes in order to give them permission to transmit. The used polling scheme is more efficient than the polling scheme of LEAP, due to the lower overhead. The QAP protocol uses the POLL, NO_DATA, and ACK control packets. No BUFF_DATA packet is considered, since it is proved to be rather useless. We assume a single channel, where the whole provided bandwidth is available for all transmissions. However, some enhancements are possible in order to support a secondary channel for the control packets transmissions. In either case, we consider that the sequence of the transmitted packets is kept. The possible polling events are depicted schematically in Fig. 3, and summarized below.

- *The AP polls an inactive node*

  The AP sends POLL to the node and waits for feedback for $t_{POLL} + t_{DATA} + t_{ACK} + 3t_{PROP\_DELAY}$. The node responds with a NO_DATA packet. If the AP successfully receives this packet, it proceeds to poll another node. In this case, it just had to wait for $t_{POLL} + t_{NO\_DATA} + 2t_{PROP\_DELAY}$. Else if the AP has not successfully received the NO_DATA control packet, it has to wait for the whole $t_{POLL} + t_{DATA} + t_{ACK} + 3t_{PROP\_DELAY}$ time duration before polling another node. Either way, the node is considered inactive.

- *The AP polls an active node*

  The AP sends POLL to the node and waits for feedback for $t_{POLL} + t_{DATA} + t_{ACK} + 3t_{PROP\_DELAY}$. The node sends a data packet (DATA) directly to the destination node and waits for an ACK packet. The AP monitors the wireless medium during all that time. If it successfully receives one or more of these two packets (DATA, ACK), then it assumes that the polled node is active. At the end of the waiting time, the AP polls another node. In case the AP fails to receive one of the above packets, it assumes that there is a bad link between it and the mobile node, so the node is considered inactive.

- *The AP fails to poll the node*

  The AP sends POLL to the node and waits for feedback for $t_{POLL} + t_{DATA} + t_{ACK} + 3t_{PROP\_DELAY}$. If the node fails to receive the POLL control packet, then there can be no feedback for the AP. So, the latter has to wait for the maximum cycle duration before polling another node. Also, it assumes that there is a bad link between it and the mobile node, so the node is considered inactive.

It is obvious that this polling scheme reduces the overhead, since no BUFF_DATA control packet is considered. This results in shorter waiting times, and finally in a shorter polling cycle. Specifically, the maximum duration of the polling cycle of QAP is $t_{BUFF\_DATA} + t_{PROP\_DELAY}$ shorter than the polling cycle of LEAP. We must make clear that the BUFF_DATA packet is removed from the polling scheme of QAP, because the AP can make sure that the polled node transmits, by just detecting transmission (no need to identify the broadcasted packets) during the time period starting at $t + t_{POLL} + t_{NO\_DATA} + 2t_{PROP\_DELAY}$ and ending at $t + t_{POLL} + t_{DATA} + t_{ACK} + 3t_{PROP\_DELAY}$. Obviously, this time period starts when the NO_DATA packet is expected and ends at the end of the polling cycle. If the AP just senses broadcast during this time, this means that the polled node transmits data, so it is assumed active. Also, we must mention that there is no need for BUFF_DATA, because it is most probable that the AP will be able to sense the polled node's broadcast, since this node was able to successfully receive the POLL packet in the first place. This means that at the beginning of the polling cycle there was a good link between the AP and the node, and probably this good link will remain for the next milliseconds, so the AP will probably manage to detect the node's broadcast.

It is clear that the QAP protocol is collision free, because a node starts data transmission only when it is polled. The nodes do not contest to gain medium access, so there are not any collisions or lost bandwidth. The longer waiting time is $t_{POLL} + t_{DATA} + t_{ACK} + 3t_{PROP\_DELAY}$, which is enough for any transmission to be completed. This is the maximum duration of a polling cycle, which is an independent unit of the polling procedure, and ensures that there will be no dead ends or unexpected events.

The above polling scheme takes into account the bursty nature of the traffic, the bursty appearance of bit-errors, and the need for minimal overhead. The AP uses the network feedback in order to characterize the polled node as active (has buffered data packets) or not. It also examines the broadcasted data packets to determine the packet priority and set this value as the specific node's priority. Later on, it will be clarified that sequential packets are probably of the same priority, because of the bursty nature of the traffic and the Highest Priority First (HPF) packet buffer discipline. We will see that the simulation results prove that this priority mechanism operates efficiently.

Upon conclusion of a polling cycle, the AP examines the network feedback information in order to decide whether the polled node belongs to the set of the active nodes or not, and updates the node's priority. The probability to choose one of the active nodes (P_{AM}) is also updated, and finally the AP selects the node to poll according to their new choice probabilities. A more detailed analysis of the node choice mechanism is presented in Section 3.3.

3.2. The Priority Model

The QAP protocol supports QoS by using packet priorities. The default number of the available priorities is $PLevels = 4$. In this case, *0* is the lowest priority and *3* is the highest priority. When a data packet is generated, it is assigned a priority according to the source application, its importance, the need for synchronous communication, and the lifetime of the packet. It is obvious that the overhead due to the packet priorities is minor, since only two extra control bits are enough for the default 4-level priority scheme. The first utilization of the packet priorities takes place in the packet buffer. QAP uses the Highest Priority First (HPF) buffer discipline, according to which the packets that carry the highest priorities are served first. Among the packets of the same priority we use First In First Out (FIFO) buffer discipline, based on the generation time of the packets. Below is the algorithmic description of this packet buffer discipline inside each node. The returned value is the index of the packet that will be chosen for sending (PacketToSendIndex).

```
MaxPriority = -1; //initialization
EarliestGenerationTime = CurrentTime + 1; //initialization
for (i=0;i<=NumberOfPackets-1;i++)
 {
  if (Packet[i].Priority>MaxPriority){
   MaxPriority = Packet[i].Priority;
   EarliestGenerationTime = Packet[i].GenerationTime;
   PacketToSendIndex = i;}
  else if(Packet[i].Priority==MaxPriority &&
          Packet[i].GenerationTime<EarliestGenerationTime){
   EarliestGenerationTime = Packet[i].GenerationTime;
   PacketToSendIndex = i;}
 }
```

The priority of a node is defined by the priority of the last packet sent by the specific node. Initially, the priority of a node is assigned the mean value $\lfloor PLevels/2 \rfloor$. When the AP polls a node and examines the broadcasted data packet, it updates the node's priority according to the priority of the packet sent. The packet priorities play a significant role in the node choice mechanism. The probability that an active node will be chosen (P_{AM}) is affected by the average priority of the active nodes. Furthermore, the choice among the active nodes is based on their priorities. A detailed description of this procedure is presented in Section 3.3.

The whole concept is based on the fact that the nature of the traffic is bursty and the buffer discipline is the HPF. Specifically, when a burst of packets is generated and arrives in the buffer, it is most probable that these packets belong to the same source application, implement the same communication service, and have the same attributes. So, they are probably assigned the same priority. Besides, the HPF algorithm actually groups the packets in the buffer according to their priorities. This means that two packets that are sent sequentially from one node, are probably of the same priority. The conclusion is that it is a good practice to assign the priority of a node according to the last packet sent priority, since the next packet that will be chosen from the buffer for sending will probably carry the same priority.

3.3. The Node Choice Mechanism – Analytical Approach

The QAP protocol updates the choice probabilities of the nodes according to their status (active or not) and their priority. According to the "active nodes" concept, it is clearly considered that under bursty traffic conditions it is most probable that a node (*k*) which transmits a data packet has more packets in the buffer [5]. So, this node is inserted in the set of the active nodes, which are more probable to be polled. If the AP assumes that the polled node did not transmit data, then it is considered to be inactive. The algorithmic description of the AP update procedure is presented below in pseudo-code form.

```
If AP receives NO_DATA { //at t + t_POLL + t_NO_DATA + 2*t_PROP_DELAY
      Node[k].active = false }
Else if AP recognizes DATA {
      Node[k].active = true
      Node[k].priority = DATA.priority }
Else if AP detects any transmission from t + t_POLL + t_NO_DATA + 2*t_PROP_DELAY till
t + t_POLL + t_DATA + t_ACK + 3*t_PROP_DELAY {
      Node[k].active = true }
Else //at the end of the polling cycle
      Node[k].active = false
```

The probability that the AP will choose one of the active nodes is P_{AM}. Obviously, the probability that an inactive node will be chosen is $1 - P_{AM}$. When there are no active nodes P_{AM} is set to 0, and when all the nodes are active P_{AM} is set to 1. If the AP has decided to poll one of the active nodes, then their polling probabilities are updated according to their priorities. If the AP has decided to poll an inactive node, one of them is polled randomly. This node choice procedure is depicted in Fig. 4, where N is the total number of nodes in the cell and M is the number of active nodes.

The probability P_{AM} is given by the equation $P_{AM} = P_A + P_Q$ (*1*). The variable P_A depends on the number of active nodes, and it holds that $P_A = P_{A1} + (M-1) \times \frac{1-P_{A1}}{N-1}$ (*2*). We define that the probability to choose one of the active nodes when there is only one active node is P_{A1}. The parameter P_{A1} is a predefined number which provides better performance when its value is close to *1*. In order to provide fairness, we set by default $P_{A1} = 0.9$. The idea is to increase step-by-step the probability to poll an active node, while the number of active nodes increases. We examine P_A for $0 < M < N$, since for $M = 0$ and $M = N$ the standard values of P_{AM} are *0* and *1* respectively. It is obvious that we get the minimum value of P_A equal to P_{A1}, when $M = 1$. Every active node other than the first contributes in P_A equally by the quantity $(1 + P_{A1})/(N - 1)$. The 3D-plot of the two-variable function $P_A(M, P_{A1})$ is shown in Fig. 5, where we assume $N = 10$. We can see the variation of P_A, and the way M and P_{A1} affect the value of P_A.

P_Q is the second addendum in the equation that gives P_{AM}. The intention is to affect the value of P_{AM} according to the average priority of the active nodes (A_Q). More specifically, the concept is to increase P_{AM} (positive P_Q) when A_Q is greater than the mean priority level ($Q_{max}/2$), and decrease P_{AM} (negative P_Q) when A_Q is less than the mean priority level. Q_{max} is the highest packet priority and it holds $Q_{max} = PLevels - 1$ (*3*). It finally holds: $P_Q = P_{Qm} \times \frac{A_Q - \frac{Q_{\max}}{2}}{\frac{Q_{\max}}{2}}$ (*4*). The parameter P_{Qm} is a predefined number that defines the maximum and minimum values of P_Q, and affects it in a proportional way. Specifically, the maximum value of P_Q is P_{Qm} when $A_Q = Q_{max}$, and the minimum value is $-P_{Qm}$ when $A_Q = 0$. Obviously, $P_Q = 0$ when $A_Q = Q_{max}/2$. It becomes clear that this method enhances QoS support in the node choice mechanism. The 3D-plot of the two-variable function $P_Q(A_Q, P_{Qm})$ is shown in Fig. 6, where we assume $PLevels = 4$. We can see the variation of P_Q, and the way A_Q and P_{Qm} affect the value of P_Q.

The precise definition of the function that gives the probability to poll one of the active nodes (P_{AM}), which includes the condition $0 \leq P_{AM} \leq 1$, is the following:

$$P_{AM} = \begin{cases} P_A + P_Q \text{, when } 0 \leq P_A + P_Q \leq 1 \\ 0 \text{, when } P_A + P_Q < 0 \\ 1 \text{, when } P_A + P_Q > 1 \end{cases} \qquad (5)$$

where $P_A + P_Q = P_{A1} + (M-1) \times \frac{1-P_{A1}}{N-1} + P_{Qm} \times \frac{A_Q - \frac{Q_{\max}}{2}}{\frac{Q_{\max}}{2}}$ (6). The 3D-plot of the two-variable function $P_{AM}(M, A_Q)$ is shown in Fig. 7, where we consider $N = 10$ and we assume the default values $P_{A1} = 0.9$, $P_{Qm} = 0.03$, and $Q_{max} = 3$.

If the AP finally decides to choose one of the inactive nodes, then it just polls one of them randomly. Otherwise, if an active node is going to be polled, then the AP chooses according to the node priorities. Specifically, the relative probability of choosing node (k) is given by the equation $P_C(k) = q + 1$ (7), where q is the priority of the specific node. The choice probabilities of all the active nodes are calculated and then are normalized. So, the actual choice probability of node (k) is $\Pi_C(k) = P_C(k) \Big/ \sum_{i=1}^{N} P_C(i)$ (8). Clearly, it holds that $\sum_{i=1}^{N} \Pi_C(i) = 1$ (9). Thus, the probability that an active node is chosen by the AP is proportional to its priority, as it is logically expected. This procedure completes the QoS support that QAP provides.

4. Performance Evaluation

4.1. Simulation Environment

In order to compare the QAP protocol against LEAP, we developed a simulation environment in C++. It has been shown that LEAP performs better than RAP and GRAP. So, the comparison between QAP and LEAP can show that the QAP protocol is an effective WLAN polling protocol, in general. The bursty traffic was simulated based on the method described in [17]. We assume that a "time slot" is the time duration of a data packet transmission. Each source node can be in one of four states, S_0, S_1, S_2, S_3. When a source node is in state S_0, then it has no packet arrivals. When a source node is in state S_1, then, at each time slot, it has one packet arrival. State S_2 denotes that there is on the average one packet arrival every two time slots. Lastly, when a source node is in state S_3, then it has two packet arrivals at each time slot. Given a station is in state S_i at time slot t, the probability that this station will transit to state S_j at the next time slot is P_{ij}. 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: $P_{01} = R/(2B(N - R))$ (10), $P_{02} = R/(4B(N - R))$ (11), $P_{03} = R/(4B(N - R))$ (12), $P_{10} = 1/B$ (13), $P_{12} = (1 - 1/B)/4$ (14), $P_{13} = (1 - 1/B)/4$ (15), $P_{20} = 1/B$ (16), $P_{21} = (1 - 1/B)/2$ (17), $P_{23} = (1 - 1/B)/4$ (18), $P_{30} = 1/B$ (19), $P_{31} = (1 - 1/B)/2$ (20), $P_{32} = (1 - 1/B)/4$ (21). The buffer size is Q packets. Any packets arriving to find the buffer full are dropped. When a packet is generated, it is assigned a packet priority (range [$0, PLevels - 1$]). The packets of the same burst are assigned the same random priority and the same destination.

In the developed simulation environment, the condition of any wireless link was modeled using a finite-state machine with three states. These are the following [18], [19]:

- State *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 *G_BER*.
- State *B* denotes that the wireless link is in a condition characterized by increased BER, which is given by the parameter *B_BER*.
- State H denotes that the pair of communication nodes is out of range of one another (hidden nodes).

We assume that the background noise is the same for all nodes, and thus, the principle of reciprocity stands for the condition of any wireless link. Therefore, for any two nodes 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 spent by a link in states *G*, *B* and *H* is exponentially distributed, but with different average values, given by the parameters *TG*, *TB*, *TH*, respectively. The status of a link probabilistically changes between the three states. When a link is in state *G* and its status is about to change, the link transits either to state *H*, with probability given by the parameter P_h, or to state *B*, with transition probability $1 - P_h$. When a link is in state *B* and its status is about to change, the link transits either to state *H*, with probability given by the parameter P_h, or to state *G*, with transition probability $1 - P_h$. Finally, when a link spent its time in state *H*, it transits either to state *G* or *B*, with the same probability (0.5). It can be easily seen that by setting the parameter P_h to zero, a fully connected network topology can be assumed, whereas for values of P_h greater than zero, the effect of the well-known "hidden node" problem on protocol performance can be studied. For example, for $P_h = 0.1$, there is a 10% probability that two nodes A and B are out of range of one another. Thus, for a third node C in range both of A and B, A and B are hidden nodes for transmissions from B to C and A to C, respectively. By changing the values of the various parameters of the above described model, the protocol can be simulated for a variety of network environments. The simulation parameters are presented below.

The variables concerning the link status were described above, and the default values are: $TG = 3$ *sec*, $TB = 1$ *sec*, $TH = 0.5$ *sec*, $G\_BER = 0$, $B\_BER = 10^{-6}$ for relatively "clean" network conditions and $B\_BER = 10^{-4}$ for rather not "clean" wireless links. Also, we set $P_h = 0$ for relatively "clean" network conditions and $P_h = 0.1$ for rather not "clean" wireless links. These values provide a fading environment, and in the next subsection it is shown that in any case QAP exhibits high performance.

Most of the default values of the network parameters presented below were also used in the original analysis of the LEAP protocol [16]. The number of the nodes in the simulated networks is $N = 10$, the buffer size is $Q = 50$, and the average burst length is $B = 10$. These values ensure that there are sufficient mobile nodes for our simulations, the buffer size is neither too small nor too large, so packets are not dropped very fast neither delayed very long in the queue, and the average burst length is large enough so as to simulate traffic in bursts and at the same time small enough so as to have various discrete bursts. The maximum number of transmission attempts per packet is *R_LIM = 6*, and the medium bit rate is *MRate = 11* Mbps. The medium rate was chosen to be *11 Mbps*, because this is the bit rate of the widely used IEEE 802.11b standard and it can be actually reached under realistic conditions. The propagation delay between any two nodes is $t_{PROP\_DELAY} = 0.0005$ ms, corresponding to distances between the nodes of *150* m. The special parameters *L_LEAP* and *a_LEAP* that concern the LEAP protocol are set to *0.1* and *0.03*, respectively, according to the original analysis found in [16]. At the MAC layer, the size of the control packets is *cpSize = 160* bits, and the default size of the data packets is *dpSize = 6400* bits. These are realistic packet size values that could be used in a working network. Every simulation was carried out until *SucRecPackets = 400000* data packets where successfully received. This simulation time is enough to provide secure metric results.

The default values that follow concern the special parameters of the QAP protocol. First of all, the number of the priority levels is *PLevels = 4*. Four priority levels are enough to characterize different kinds of traffic with different QoS requirements. The probability of polling an active node when there is only one active is $P_{A1} = 0.9$. This value ensures that the protocol efficiently benefits the active nodes and also provides fairness by giving the opportunity to non-active nodes to be polled. Lastly, the maximum variation of the probability of polling an active node depending on the average priority of the active nodes is $P_{Qm} = 0.03$. So, the QoS requirements of the active nodes affect the probability to poll a node that is active, but in a way that the parameter P_{A1} and the number of active nodes always play the primary role, while fairness is also provided. It should be noticed that the above mentioned values were decided after careful analysis and various simulations.

4.2. Simulation Results

First of all, it must be mentioned that one simulation environment was developed, capable of simulating both the QAP and the LEAP protocols, and adapted to the special features of each one of them. Also, the simulation results that concern the LEAP protocol coincide with those presented in [16], when using the same parameter values. The random number generator that is used by the simulator is a classic multiplicative congruential random number generator with period 2^{32} provided by ANSI C. The simulation results presented in this section are produced by a statistical analysis based on the "sequential simulation" method [20]. Specifically, we perform simulations in a sequential way, until the relative statistical error of the estimated mean value falls below an acceptable threshold. When the relative statistical error is low, then the confidence interval is narrow, since the relative statistical error is defined as the ratio of the half-width of the given confidence interval at the point estimate. For this statistical analysis we used 95% confidence intervals. The relative statistical error threshold varies depending on the meaning of the metric and the magnitude of the produced value. However, this threshold was usually assumed to be lower than 2% and never exceeded 5%.

Initially, we assume a "clean" network, where $B\_BER = 10^{-6}$ and $P_h = 0$. Under low BER conditions QAP performs better than LEAP. This happens because of the lower overhead of QAP, which is due to the optimized polling model and the shorter polling cycle. Also, the adaptive polling algorithm of QAP is more probable to poll a truly active node compared to the learning automata-based algorithm of LEAP. Under these network conditions, when the load is lower than 80% the two protocols have equal throughput values, because these values are almost identical to the offered load values, which means that they perform almost perfectly. When the load is higher, QAP provides higher throughput. Also, QAP provides lower packet delays, which is shown in Fig. 8. It must be noticed that we assume that high priority packets are the packets which are assigned a priority higher than $(PLevels - 1)/2$. The corresponding curve shows that the high priority packets meet significant lower delays, especially when the throughput is high, which is a proof of the QoS support of QAP.

In the next simulated network, the BER is increased, and the “hidden nodes” problem is present. Specifically, we assume $B\_BER = 10^{-4}$ and $P_h = 0.1$. Under these network conditions, the difference between the QAP performance and the LEAP performance is greater. The throughput provided by QAP is higher than the one provided by LEAP, when the load is greater than 0.65, as it can be seen in Fig. 9. Below this value, the two protocols exhibit almost “perfect” performance. In a rather harsh environment like this, the effective characteristics of the QAP protocol, especially the low overhead and the efficient adaptive polling algorithm, become obvious. In Fig. 10, we can see that QAP provides packet delays clearly lower than the delays provided by LEAP. The average delay of the high priority packets are significantly low, which shows that QoS is supported in any network condition.

The default value of the data packet size was assumed to be by default 6400 bits, in the simulations. It is known that the performance of any MAC protocol is usually worse, when the value of the data packet size gets small compared to the control information. This happens, because the overhead is greater, which finally causes low throughput and high packet delay. However, some kinds of networks use rather small data packets, like the ATM networks. As it was expected, the QAP protocol has great advantage compared to the LEAP protocol, when using data packets that are not many times bigger than the control packets. The efficiency of QAP, in this case, is proved by the simulation results of a network with small data packet size equal to 800 bits. These results showed that QAP provides clearly higher throughput and significantly lower packet delays. In Fig. 11, the results that concern the packet delay are plotted. The high priority packets are favored again, since their delays remain particularly low.

The packet priority model gives to the proposed protocol the ability to distinguish between different kinds of traffic. This means, for example, that QAP is capable of supporting streaming video, while file transfers take place, even under harsh network conditions. Fig. 12 shows that the delay of the high priority packets remains impressively stable, while the overhead alters. Specifically, we plot the average packet delay for different values of the data packet size, while keeping the control packet size stable. The QAP protocol provides lower packet delays, and the high priority packet delays are considerably low. In this case, we assumed a high load network environment (*R = 1*).

In the simulations, we have also measured the packet loss rate and the high priority packet delay as a percentage of the low priority packet delay. The results show that QAP provides lower packet loss rate than LEAP, when the offered load is above 0.55, since the packet loss rate for lower values of load is zero for the both protocols. The corresponding curves are depicted in Fig. 13. This behavior is due to the fact that QAP utilizes the provided bandwidth in a more efficient way, since the wrong polls and the waiting times are reduced. In the analysis of the QAP protocol, we wanted to find out in what degree the high priority packets are favored. So, we got simulation results of the high priority packet delays and we plotted them as percentage of the low priority packet delays in Fig. 14. The graph makes clear that the increase of the throughput, which means that the average packet delay is increased at the same time, provides a decrease of the high priority packet delay related to the low priority packet delay. Obviously, the high priority packets are favored in a relatively greater degree under harsh network conditions, which means that the QAP protocol ensures QoS support in any case.

The influence of the number of nodes on the protocol behavior was also examined. The simulation results show that the network throughput remains stable, while the number of nodes increases, for both QAP and LEAP protocols. However, QAP exhibits always higher throughput than LEAP. These results are depicted in Fig. 15. The number of wrong polls was also measured. In Fig. 16, it can be seen that QAP identifies the active nodes more efficiently, so it provides less wrong polls than LEAP.

The QAP network was also simulated for different values of the buffer size. As it was expected, the results showed that a small buffer size leads to increased packet losses, because many packets arrive to find the buffer full, so they are dropped. By keeping the rest of the network parameters constant, we notice that when the buffer size value becomes higher than 50 packets, the packet loss rate stabilizes, as it can be seen in Fig. 17. In comparison with the LEAP protocol, QAP offers lower packet loss rates for any buffer size. It should be also mentioned that the increase of the buffer size causes increased average packet delay, since the packets stay, on the average, longer in the buffer.

Lastly, the influence of the average burst length on the network performance was examined via simulations. Both QAP and LEAP protocols, base their polling algorithm on the fact that the traffic is bursty. This is a realistic assumption, as it has been shown that most of the network applications produce traffic in a bursty way. This is especially true for traffic concerning multimedia applications, which have special QoS requirements. Fig. 18 shows that QAP and LEAP offer lower packet delays when the average burst length increases, while QAP guarantees steady low delays for the high priority packets irrespective of the burst length. One obvious reason that explains this behavior is the fact that the two protocols recognize the active nodes more efficiently, when they transmit sequentially a great number of packets. Whatever the burst length, QAP always offers lower packet delays than LEAP.

5. Conclusion

This work proposed the QoS supportive Adaptive Polling (QAP) protocol for wireless LANs. The protocol is capable of operating efficiently under bursty traffic conditions. It exhibits high performance, by providing high throughput and low packet delays. The comparison between the QAP protocol and the LEAP protocol has shown that, in any case, QAP performs better and, in addition to that, it supports QoS. The protocol is based on a self-adaptive polling algorithm [21], which decreases the number of wrong polls to inactive nodes. The overhead is reduced and the polling scheme is generally optimized. A special characteristic is the support of packet priorities. QAP provides low delays for the high priority packets, so it is able to support different kinds of traffic at the same time (e.g. asynchronous and time-bounded communication). QoS is supported even under harsh network conditions with increased BER. The proposed model is not difficult to implement, since the polling scheme based on the active nodes and the node priorities is rather simple. Furthermore, the protocol is collision free and no simultaneous transmissions take place. As future work, the packet priorities can be corresponded to packet lifetimes, the priorities could dynamically change according to the importance, the nature, and the deadline of the packets, and specific services could assign specific packet priorities according to the network settings.

References

[1] P. Nicopolitidis, M. S. Obaidat, G. I. Papadimitriou, and A. S. Pomportsis, *Wireless Networks*, New York : Wiley, 2003.

[2] P. Nicopolitidis, G. I. Papadimitriou, and A.S. Pomportsis, "Design alternatives for wireless local area networks," *Int. J. Commun. Syst*., vol. 14, pp.1-42, Feb. 2001.

[3] A.C.V. Gummalla and J.O. Limb, "Design of an access mechanism for a high speed distributed wireless LAN," *IEEE J. Select. Areas Commun*., vol. 18, pp. 1740-1750, Sept. 2000.

[4] A. S. Tanenbaum, *Computer Networks*. Englewood Cliffs, NJ: Prentice-Hall, 2002.

[5] G. I. Papadimitriou, and A. S. Pomportsis, "Adaptive MAC protocols for broadcast networks with bursty traffic," *IEEE Trans. Commun*., vol. 51, no. 4, pp.553-557, April 2003.

[6] G. I. Papadimitriou, T.D. Lagkas, and A. S. Pomportsis, "HIPERSIM: A sense range distinctive simulation environment for HiperLAN systems," *J. Simulation*, vol. 79, no.8, pp.462-481,Aug. 2003.

[7] K.-C. Chen, "Medium access control of wireless LANs for mobile computing," *IEEE Network Mag*., pp. 50-63, Sept./Oct. 1994.

[8] Ajay Chandra, V. Gumalla, and John O. Limb, "Wireless Medium Access Control Protocols," *IEEE Communications Surveys*, pp. 2-15, Second Quarter 2000.

[9] Teerawat Issariyakul, Ekram Hossain, and Dong In Kim, "Medium access control protocols for wireless mobile ad hoc networks: issues and approaches," *Wireless Communication and Mobile Computing*, 3:935-958, 2003.

[10] Imrich Chlamtac, Marco Conti, Jennifer J.-N. Liu, "Mobile ad hoc networking: imperatives and challenges," *ELSEVIER Ad Hoc Networks* 1, 13-64, 2003.

[11] Jaime Sanchez, Ralph Martinez, Michael W. Marcellin, "A survey of MAC protocols proposed for Wireless ATM," *IEEE Network*, vol. 11, no. 6, pp. 52-62, Nov/Dec 1997.

[12] Ian F. Akyildiz, Janise McNair, Loren Carrasco, Ramon Puigjaner, "Medium Access Control Protocols for Multimedia Traffic in Wireless Networks," *IEEE Network Magazine*, vol. 13, no. 4, pp. 39-47, July / August 1999.

[13] IEEE 802.11e WG, "P802.11e: Draft Amendment to Standard [for] for Information Technology-- Telecommunications and Information Exchange Between Systems-- LAN/MAN Specific Requirements-- Part 11 Wireless Medium Access Control (MAC) and Physical Layer (PHY) specifications: Amendment 7: Medium Access Control (MAC) Quality of Service (QoS) Enhancements," Draft Version: D13 2005.

[14] K.-C. Chen and C.-H. Lee, "RAP – a novel medium access-control protocol for wireless data networks," in *Proc. IEEE GLOBECOM*, 1993, pp. 1713-1717.

[15] Kwang-Cheng Chen, Cheng-Hua Lee, "Group randomly addressed polling for wireless data networks," in *Proc. IEEE ICC*, 1994, pp. 1713-1717.

[16] P. Nicopolitidis, G. I. Papadimitriou and A.S.Pomportsis, "Learning automata-based polling protocols for wireless LANs," *IEEE Trans. Commun.*, vol. 51, no. 3, pp.453-463, March 2003.

[17] G. I. Papadimitriou, and A. S. Pomportsis, "Learning automata-based TDMA protocols for broadcast communication systems with bursty traffic," *IEEE Commun. Letters*, vol.4, no.3, pp.107-109, March 2000.

[18] E. Gilbert, "Capacity of a burst noise channel," *Bell Syst. Tech. J.*, vol. 39, pp.1253-1265, Sept. 1960.

[19] M. Zorzi, R. R. Rao, and L. B. Milstein, "On the accuracy of a first-order Markov model for data transmission on fading channels," in *Proc. ICUPC*, Tokyo, Japan, Nov. 1995, pp. 211-215.

[20] Krzysztof Pawlikowski, Hae-Duck Joshua Jeong, and Jong-Suk Ruth Lee, University of Canterbury, "On Credibility of Simulation Studies of Telecommunication Networks", *IEEE Communications Magazine*, vol.40, no.1, pp.132-139, Jan. 2002.

[21] A. Farago, A. D. Myers, V. R. Syrotiuk, and G. V. Zaruba, "Meta-MAC protocols: automatic combination of MAC protocols to optimize performance for unknown conditions," *IEEE J. Select. Areas Commun.*, vol. 18, pp. 1670-1681, Sept. 2000.

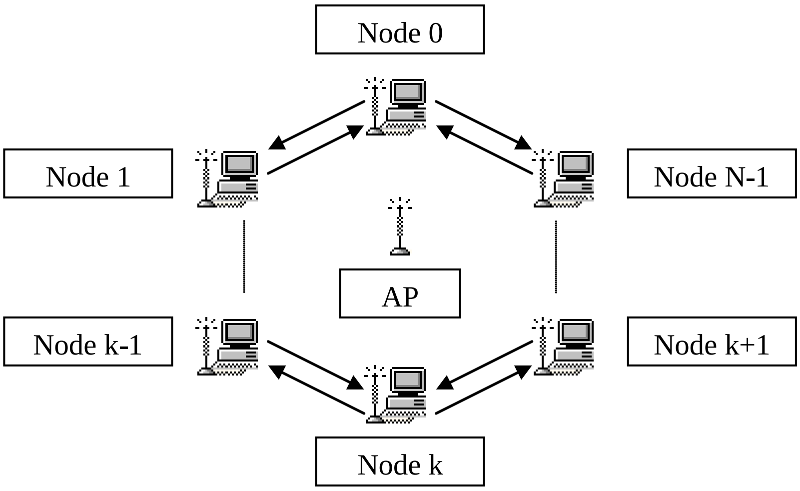

Fig. 1. Topology of the considered wireless LAN with N mobile nodes

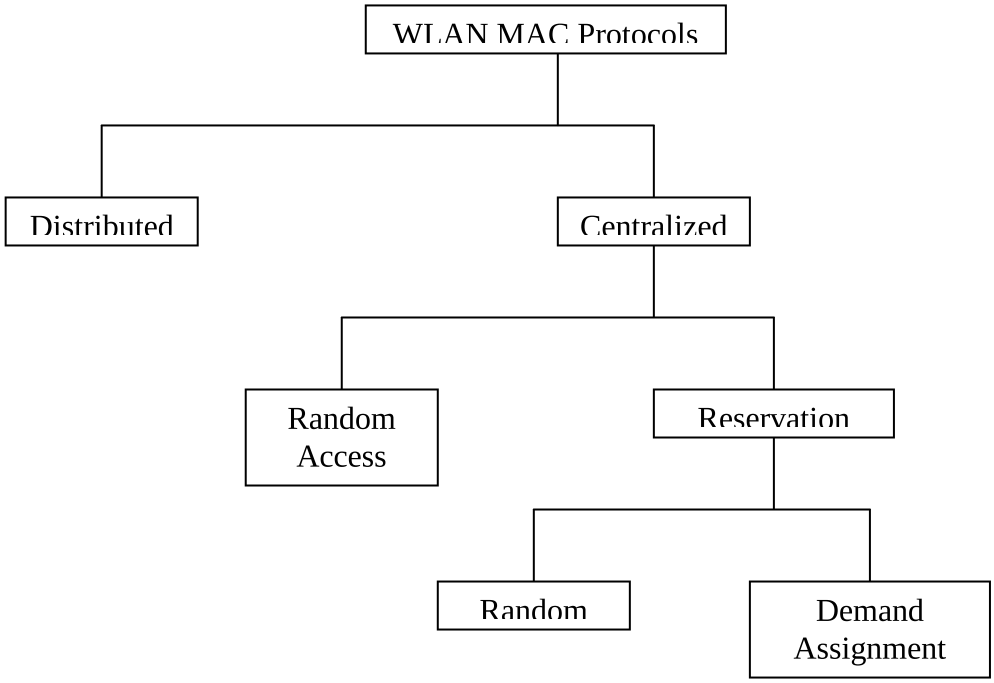

Fig. 2. Classification of WLAN MAC Protocols

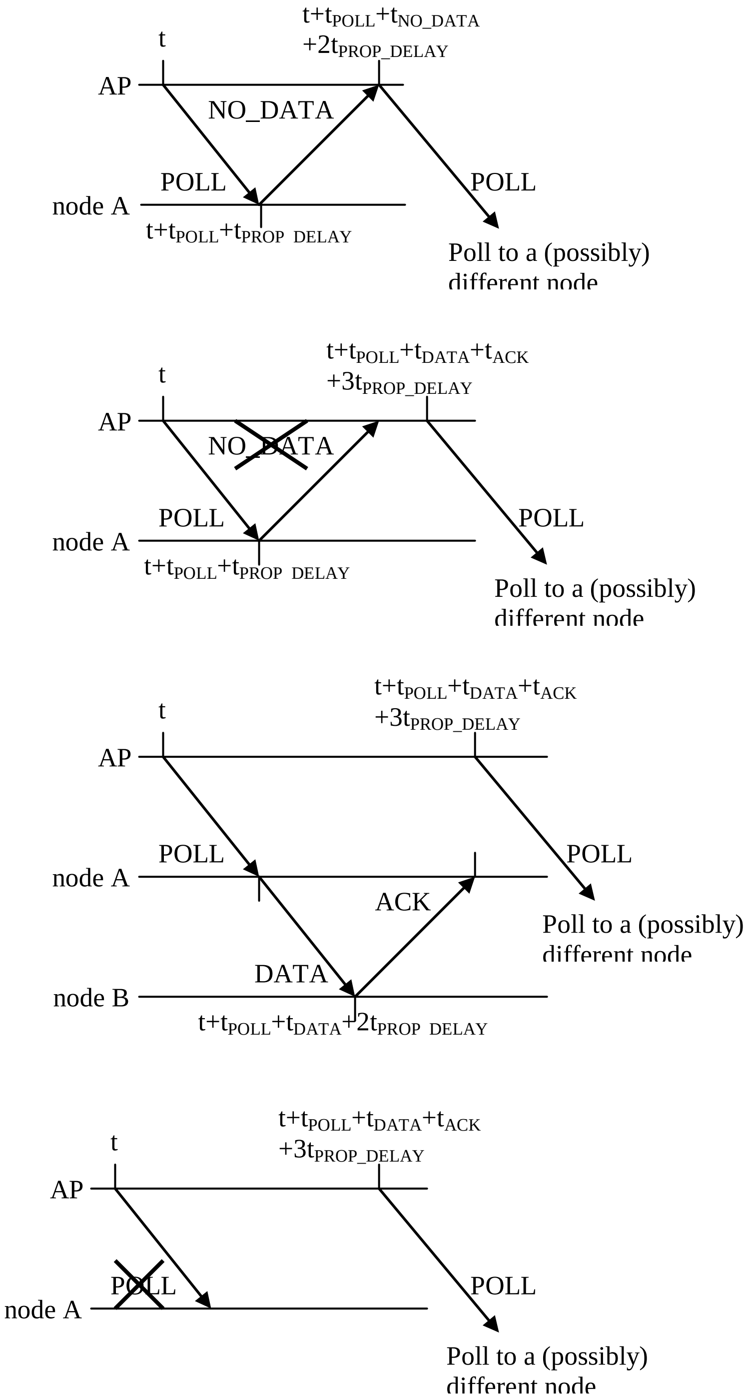

Fig. 3. The polling scheme of the QAP protocol

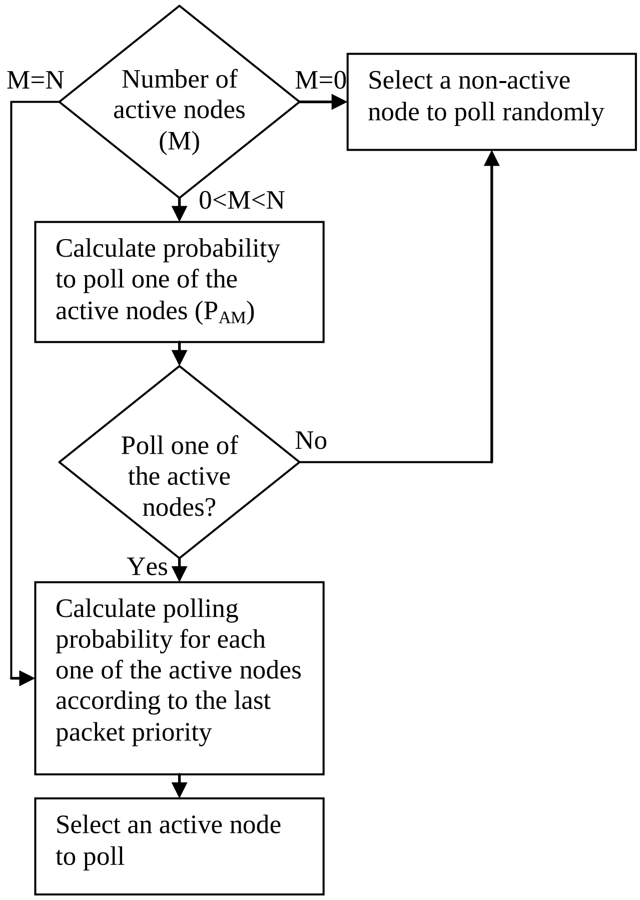

Fig. 4. Overview of the node choice mechanism of QAP

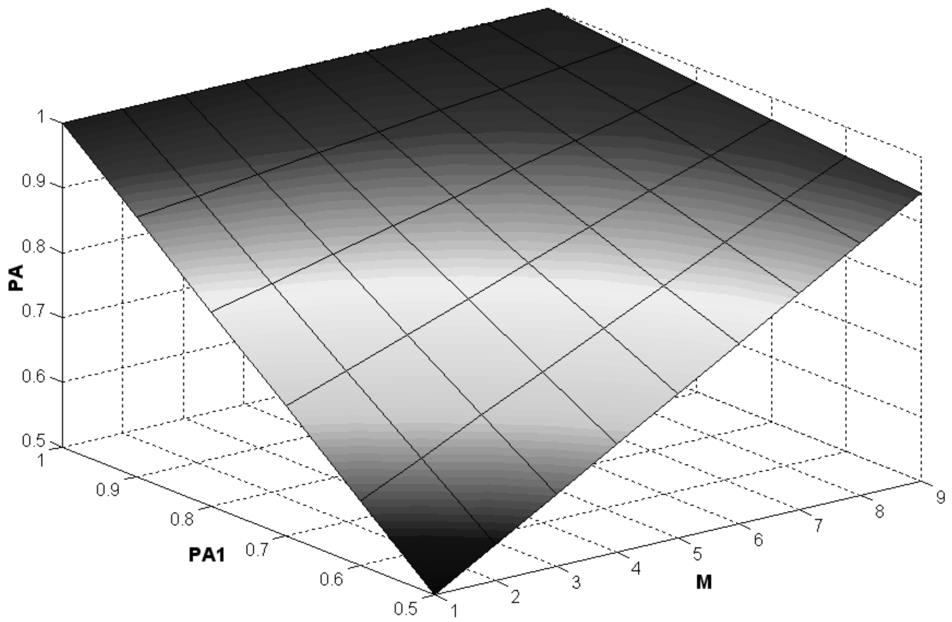

Fig. 5. Probability polling an active node (PA), without taking into account priorities, as a function of the number of active nodes (M) and the probability polling a single active node (PA1)

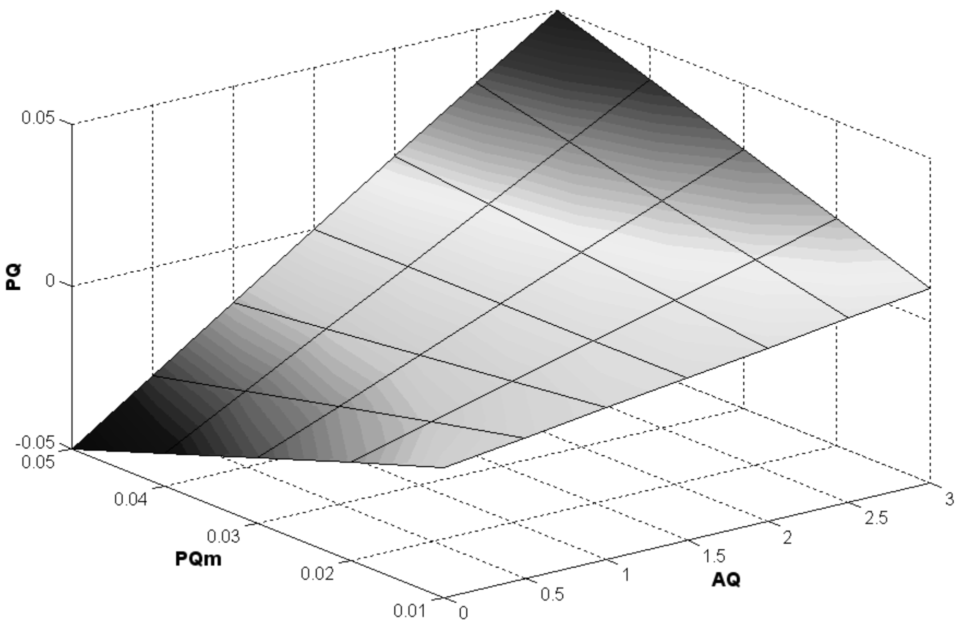

Fig. 6. The variation of the probability polling an active node (PQ), depending on the packet priorities, as a function of the maximum variation (PQm) and the average priority of the active nodes (AQ)

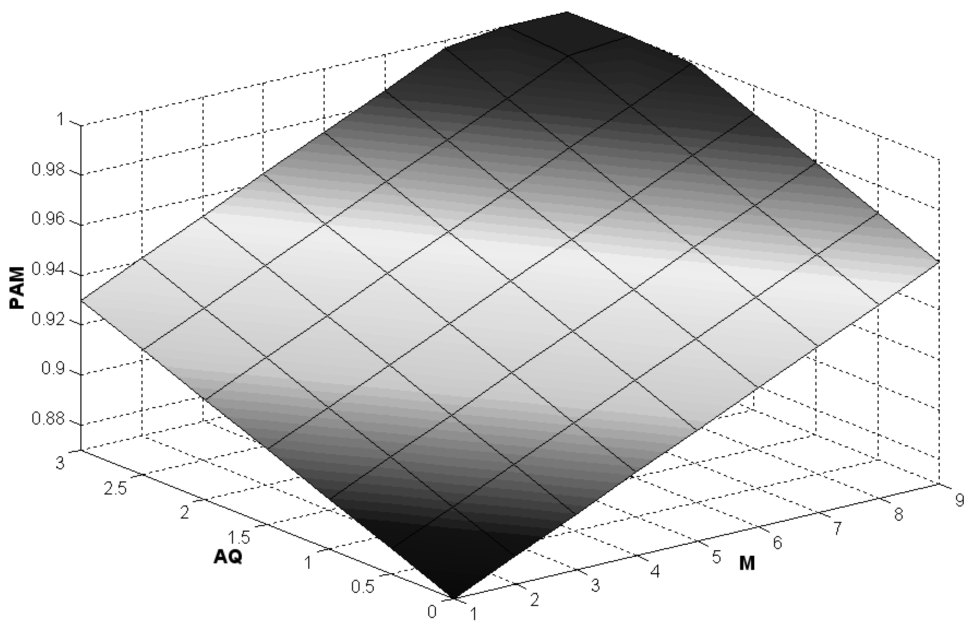

Fig. 7. Probability polling one of the active nodes (PAM) as a function of the number of active nodes (M) and the average priority of the active nodes (AQ)

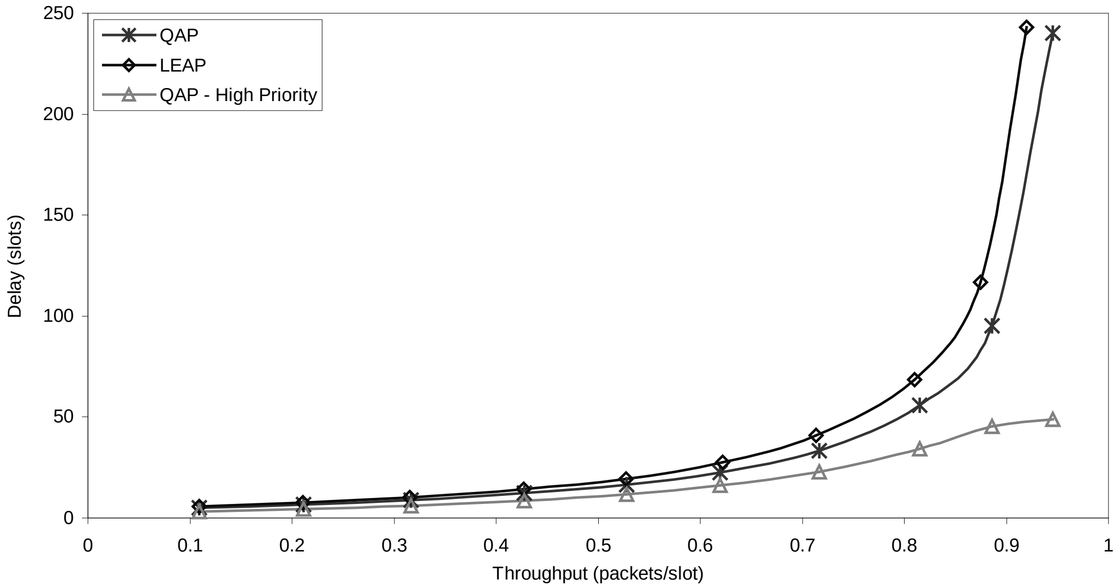

Fig. 8. ($B\_BER = 10^{-6}$, $P_h = 0$) Average packet delay versus throughput, where we plot for packet loss rate lower than 15%

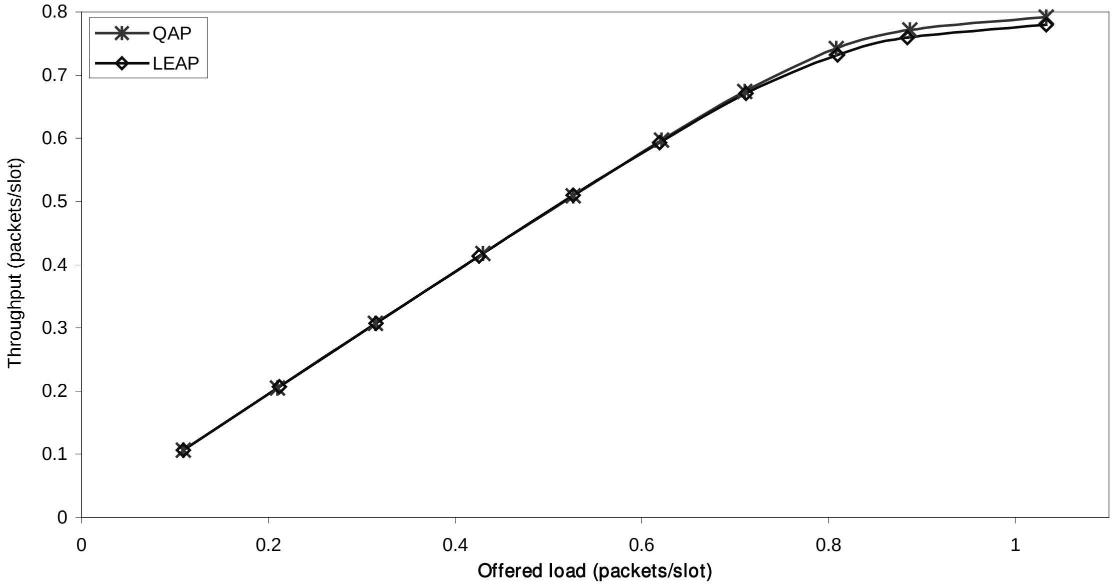

Fig. 9. ($B\_BER = 10^{-4}$, $P_h = 0.1$) Throughput versus offered load

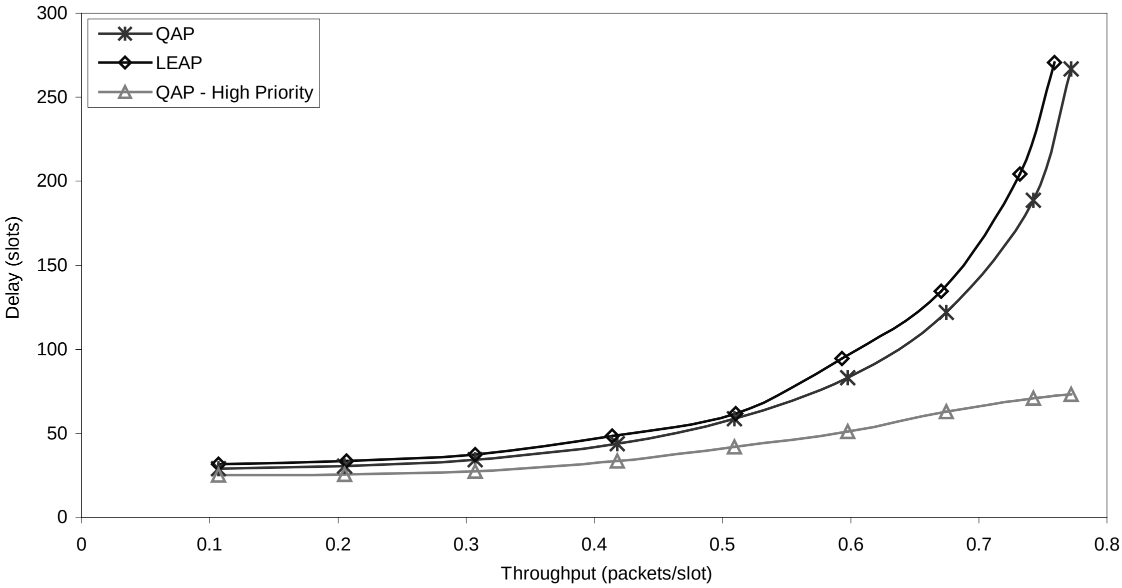

Fig. 10. ($B\_BER = 10^{-4}$, $P_h = 0.1$) Average packet delay versus throughput, where we plot for packet loss rate lower than 15%

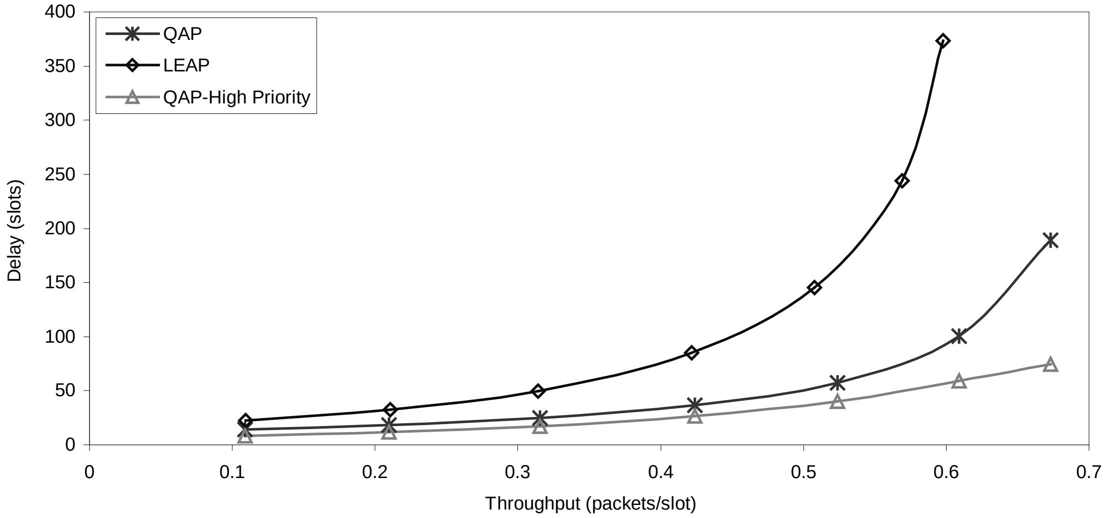

Fig. 11. ($B\_BER = 10^{-6}$, $P_h = 0$, $dpSize = 800$) Average packet delay versus throughput, where we plot for packet loss rate lower than 20%

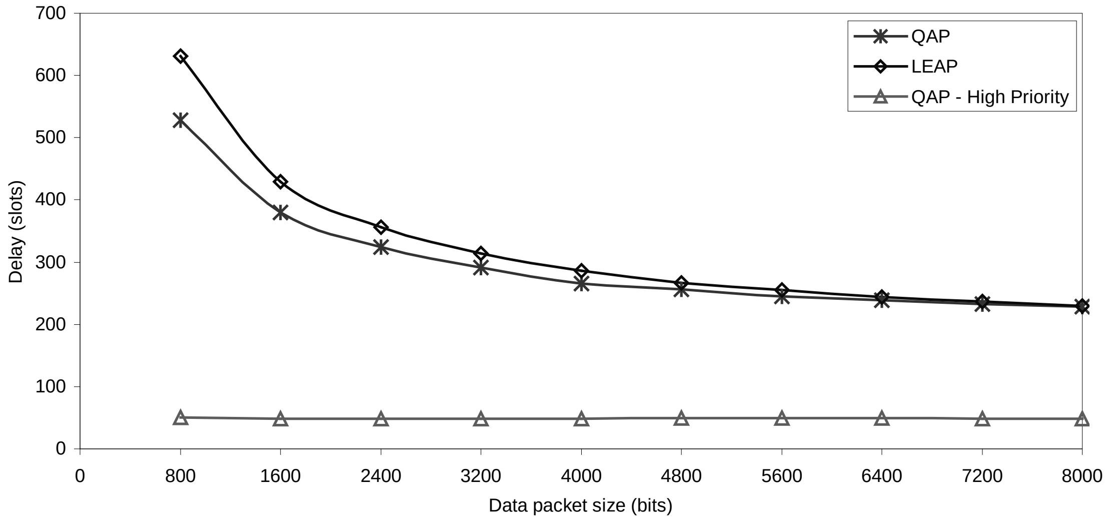

Fig. 12. ($B\_BER = 10^{-6}$, $P_h = 0$, $R = 1$) Average packet delay versus data packet size, where we plot for any packet loss rate

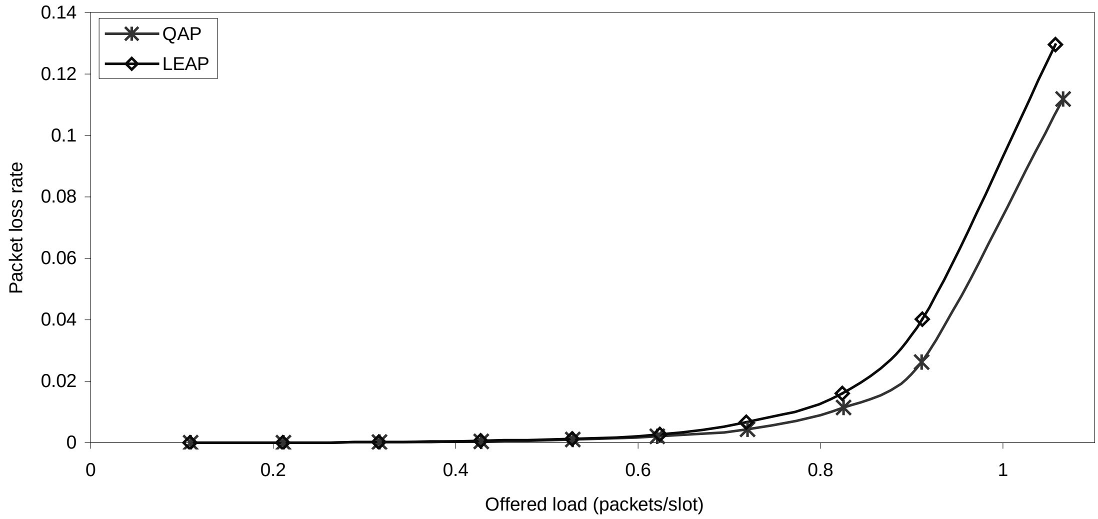

Fig. 13. ($B\_BER = 10^{-6}$, $P_h = 0$, $R = 1$) Packet loss rate versus offered load

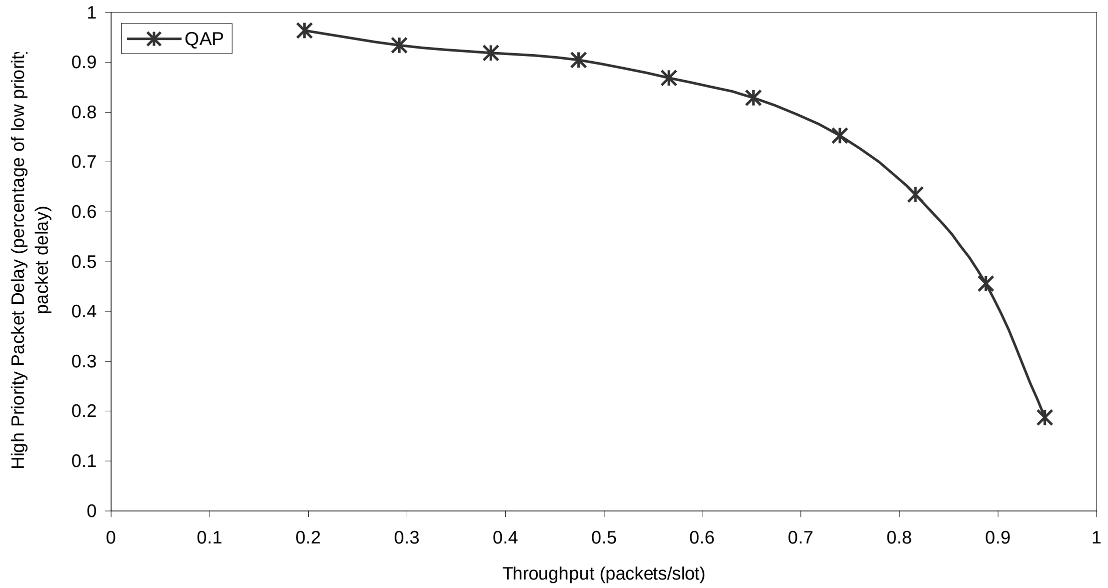

Fig. 14. ($B\_BER = 10^{-6}$, $P_h = 0$) High priority packet delay as a percentage of the low priority packet delay versus throughput

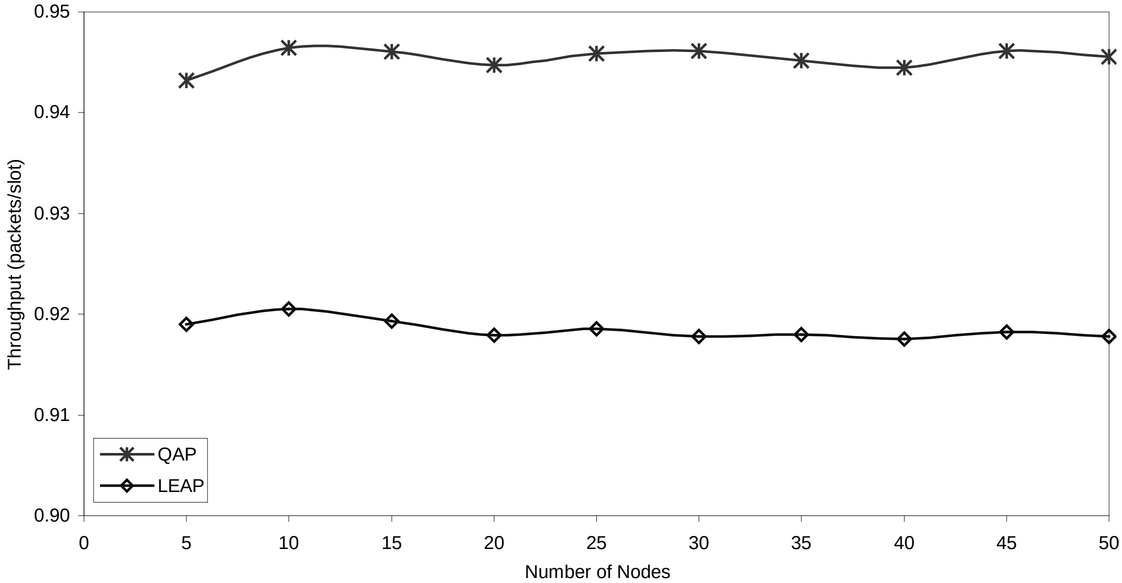

Fig. 15. ($B\_BER = 10^{-6}$, $P_h = 0$) Throughput versus number of nodes

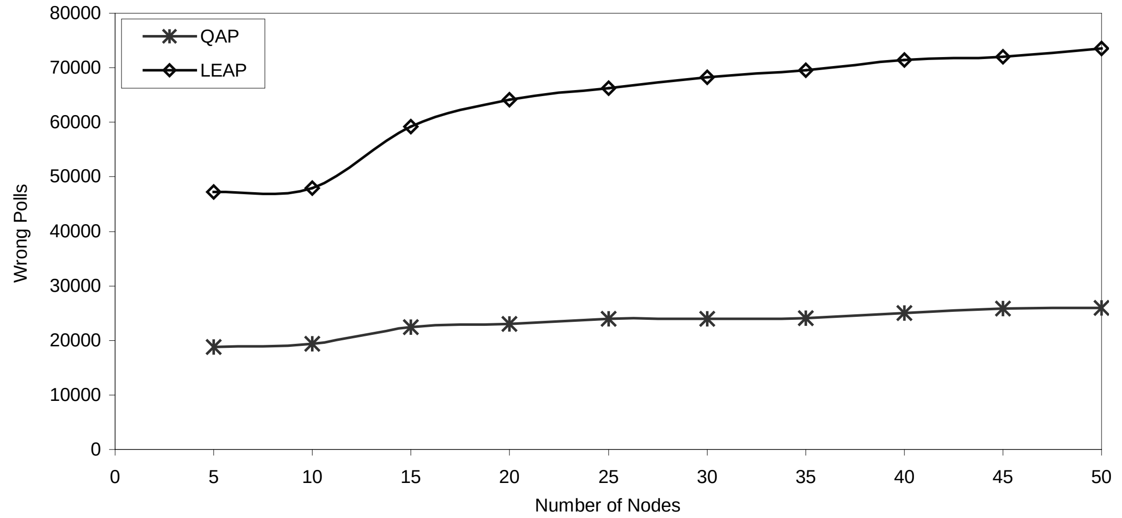

Fig. 16. ($B\_BER = 10^{-6}$, $P_h = 0$) Number of wrong polls versus number of nodes

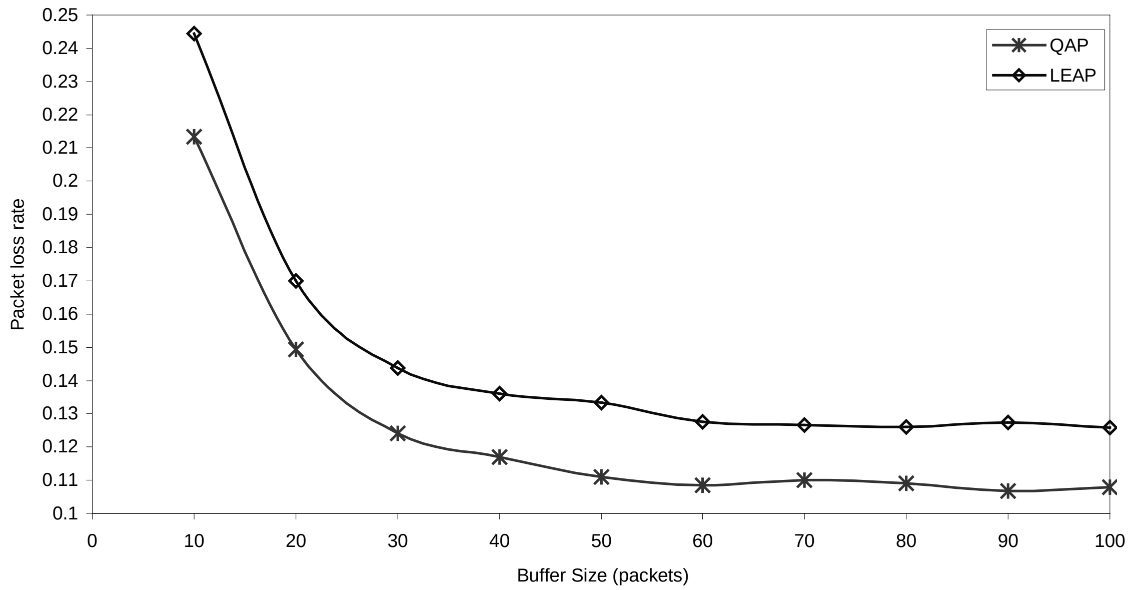

Fig. 17. ($B\_BER = 10^{-6}$, $P_h = 0$) Packet loss rate versus buffer size

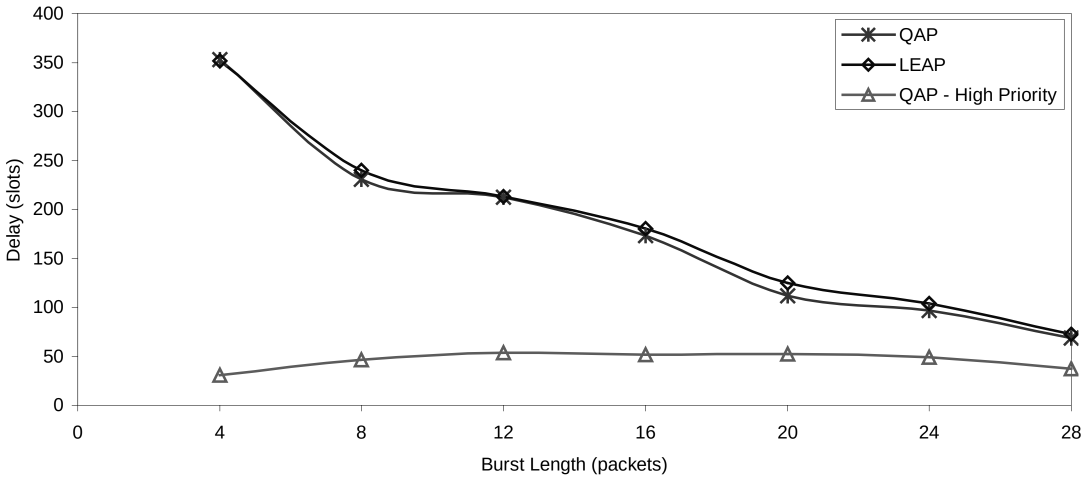

Fig. 18. ($B\_BER = 10^{-6}$, $P_h = 0$) Delay versus burst length