An Adaptive Polling Algorithm for PCF Mode of 802.11 Wireless LANs

Xuanming Dong, Pravin Varaiya, and Anuj Puri
Department of Electrical Engineering & Computer Science
University of California at Berkeley
Berkeley, CA 94720

Abstract— **IEEE 802.11 PCF mode is defined to support time-bounded applications in wireless LANs. The poll scheduling plays an important role in PCF mode operation. It differs from the packet scheduling of routers in that incomplete queueing information in each pollable station is known to the scheduler. This paper proposes an adaptive polling algorithm to improve the wireless medium utilization. In our scheme, each mobile station is assigned a priority by the point coordinator. Based on recent poll feedbacks, the priority for each mobile station will be dynamically updated using the Additive Increase/Multiplicative Decrease algorithm. The proposed polling algorithm is compatible with the IEEE 802.11 standard and requires a simple extension. our simulation studies show that the performance of wireless LANs are improved in terms of the successful poll rate and the aggregate throughput.**

I. Introduction

The IEEE 802.11b standard [4], [6], in which the 2.4-GHz unlicensed ISM spectrum can be used to transmit data with rates up to 11 mbps, is widely supported by wireless manufacturers nowadays. With dropping hardware prices and enhanced security mechanisms, 802.11b-compliant wireless equipments have already been deployed in office buildings, residential buildings, and public areas, etc. The wireless market is witnessed to grow from a few experts to more than millions of users within a few years. Because of its increased mobility and flexibility without the constraint of wiring, wireless LANs have become essential front-end components of current networks. With a mobile station connected to a wireless LAN, users can access the networking resources anywhere and anytime.

The networking applications roughly fall into three categories: intolerant applications, tolerant applications, and elastic applications [11]. Intolerant applications, for example, real-time control systems, require a reliable maximum packet delay bound without packet loss. In tolerant applications, like video and audio transmissions, a certain loss of packets can be handled. Elastic applications don't impose delay constraints on packets, and will wait for packets until packets arrive or are detected to be dropped by networks. As we'll discuss late, IEEE 802.11 MAC layer defines two operation modes: Distributed Coordination Function (DCF) and Point Coordination Function (PCF). DCF aims to provide best effort services to upper layer protocols, so it is most suitable for elastic applications. PCF is designed particularly for intolerant applications and tolerant applications, although it can't guarantee to support these applications completely. While DCF has been extensively investigated, PCF is still in its infancy. As the wireless users continue to increase and the performance of wireless equipments continue to improve, we believe PCF will get more attentions because of its potential ability to provide services for time-bounded applications.

The polling algorithm for IEEE PCF mode differs from the packet scheduling of routers in that incomplete traffic information of each pollable station is known to the scheduler [10]. Currently round robin-based polling algorithm is presumed to be implemented in the Point Coordinator [4], [8]. However, since the traffic is not evenly distributed over all mobile stations, the wireless LAN performance in PCF mode suffers from the inefficient scheduling. Some stations may be active to generate traffic, while others are idle for long time. In most cases, the polls to idle stations waste scarce radio resource. Actually, more efficient polling schemes can be found by utilizing past poll history to stations. Although the poll history provides incomplete information about the queueing buffer in each station, it is helpful to predict current state of each station. In this paper, we propose an adaptive polling algorithm to improve the overall throughput and medium utilization. In our scheme, each mobile station is assigned a priority by the point coordinator. Based on recent poll feedbacks, the priority for each mobile station will be dynamically updated using the Additive Increase/Multiplicative Decrease algorithm. We have implemented our proposed polling algorithm in ns2. Our simulation studies show that the performance of wireless LANs are improved in terms of the successful poll rate and the aggregate throughput.

II. Overview of IEEE 802.11 PCF Mode

A. IEEE 802.11

IEEE 802.11 is a standard that defines the MAC layer and Physical layer for wireless LANs operating in the unlicensed Radio frequency band. As shown in figure 1, IEEE 802.11 shares the same LLC layer, IEEE 802.2 standard, with Ethernet. Several amendments to 802.11, for example, 802.11a, 802.11b, defines the physical layer with different operation characteristic [5], [6], like data rate, modulation, frequency, etc. Some other ongoing amendments aim to provide QoS mechanisms, enhance security, and push the data rate higher [7].

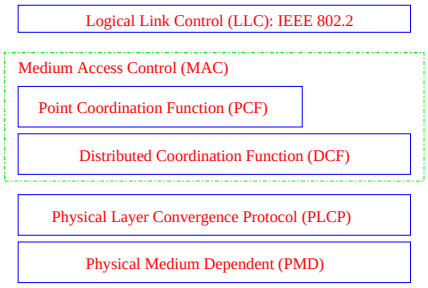

Fig. 1. The protocol architecture for IEEE 802.11

Mobile stations that have a wireless network interface card and access points are the basic equipments of IEEE 802.11 wireless LANs. These equipments can be configured into one of the two modes: ad-hoc and infrastructure. With the ad-hoc configuration, each station performs the same functionality and communicates directly with other stations within its range without the support of access points. If two stations are not in the range of each other, they need intermediate stations to forward the traffic. With the infrastructure configuration, the wireless network consists of access points and stations. Each station communicates with the access point instead of other stations within its range. The access point can be connected to a wired network and function as bridge between wireless networks and wired networks.

B. PCF Mode

The 802.11 MAC layer defines two access methods: DCF and PCF. The basic access method of DCF is known as Carrier Sense Multiple Access/Collision Avoidance (CSMA/CD). DCF is also enhanced with binary exponential backoff procedure and RTS/CTS mechanisms, etc. DCF mode is mandatory to all implementations [4].

PCF access method is derived from the classical Time Division Multiplexing (TDM) technique. The point coordinator works as a master and stations work as slaves. In PCF mode, the transmission time is spliced into poll slots for stations. Stations are allowed to send data only when they receive the polling frames from the point coordinator. Each station can transmit one data frame upon receiving a polling frame. The point coordinator, which shall reside at the access point, determines which station should be polled for data transmissions. Without the uncertain delay caused by collisions, PCF provides bounded delay and is suitable for transmitting data generated by intolerant applications and tolerant applications, like audio, video, etc. PCF is above DCF in the sense that PCF needs to understand the frames emitted by stations in DCF mode. PCF is only for the infrastructure configuration and is optional for the access point, which decides whether PCF mode is enabled or not. Even if the access point enables the PCF mode, the stations still can decide whether they want to be polled or not.

If the DCF mode and PCF mode coexist at the access point, transmission slots are divided into Contention Free Period (CFP, for PCF mode) and Contention Period (CP, for DCF mode). CFP and CP appear alternately. The CFP duration is constrained by CFPMaxDuration, a MIB parameter, while the CFP frequency is defined by CFPRate, another MIB parameter, as shown in figure 2.

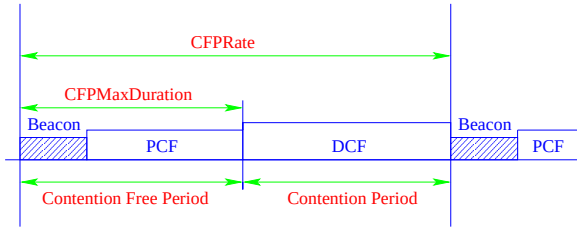

Fig. 2. The DCF mode and PCF mode of IEEE 802.11

C. Polling List

Only the stations in the polling list maintained by the point coordinator are eligible to receive polls. When a station joins a BSS, it can ask to be appended to the polling list by sending the Association Request frame with the CF_Pollable subfield set to be TRUE. If a station is not in the polling list at the beginning, it also can send ReAssociation Request frame to do that. If a station is in the polling list initially but it doesn't want to be polled late, it can send ReAssociation Request frame to the point coordinator. Accordingly, the point coordinator will remove the station from the polling list.

This paper assumes that there are two way traffic between the point coordinator and stations. Traffic from the point coordinator to stations and traffic from a station to the point coordinator or other stations are called outbound traffic and inbound traffic namely. The point coordinator may use CFP to transmit broadcast or multicast frames. We only consider the poll scheduling for inbound traffic.

III. An Adaptive Polling Algorithm

A. Basic Notations

Definition successful poll: If the point coordinator sends a poll frame to a station and receives data from the polled station before starting to poll other stations, the poll is called a successful poll.

Definition missed poll: If the point coordinator sends a poll frame to a station and doesn't receive data from the polled station within a given time, the poll is called a missed poll.

Good polling algorithms should keep the medium busy for payload data instead of management and control frames and keep the medium shared by all stations fairly. In this section, we will formulate an adaptive polling algorithm based on recent poll feedacks from wireless stations, to achieve more successful polls and higher aggregate throughput.

To facilitate following discussions, some main variables and constants for the polling algorithm are shown in table I.

B. Dynamic Priority Assignment

In our new polling algorithm, each pollable station has a priority, which is dynamically assigned by the point coordinator based on the recent number of successful polls and missed

1	the highest priority
m	the lowest priority and the number of priorities
$round$	the round number, $1 \leq round \leq m$
$priority$	the priority number of the list or station being polled
$pollreply$	a boolean variable. $TRUE$ means a successful poll and $FALSE$ means a missed poll
$station$	pointer to a station in a list
$listIndex$	index to the list being polled for current round $1 \leq listIndex \leq m$
$L[i]$	pointer to the list for pollable stations with priority i

TABLE I

MAIN VARIABLES AND CONSTANTS FOR THE ALGORITHM

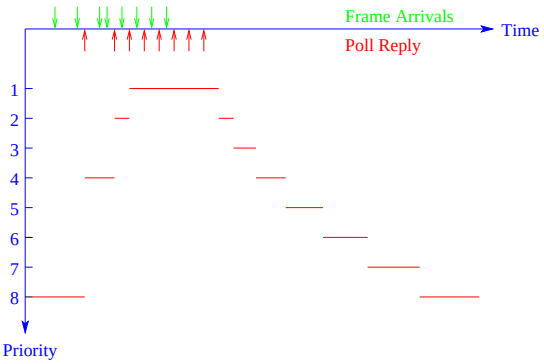

Fig. 3. The dynamic priority assignment algorithm

polls. The priority of a station is related to the predicated amount of traffic from it. Note that the priority in this paper is different from the concept presented in [8]. Their concept of priority is relevant to the service quality defined in QoS parameters. The priority in this paper is internal to the point coordinator, whereas the priority in [8] is external to users or applications. The polling algorithm proposed in this paper can be applied to traffic with the same service priority.

Assume that there are m level of priorities, 1, 2, ..., m, respectively. The larger the number, the lower the priority. Stations with high priority are expected to have more recent traffic. The priority of each station is updated by the point coordinator using an Additive Increase/Multiplicative Decrease (AIMD) algorithm [1], which is shown below.

UpdatePriority($priority, pollreply$)
if $pollreply$ **then**
  if $priority > 1$ **then**
    $priority = \lfloor priority/2 \rfloor$
  end if
else
  if $priority < m$ **then**
    $priority = priority + 1$
  end if
end if

Because of the autocorrelation property within the same traffic flow and the correlation property among different traffic flows [2], [3], a successful poll means more potential traffic from a station. Thus the point coordinator decreases the number of a station priority by half whenever it gets a successful poll from that station. If the point coordinator gets a missed poll from a station, it only increases the number of that station priority by 1 to avoid the case that the station is temporarily short of traffic.

If a station has bursty traffic, it takes at most $\log_2 m$ consecutive polls for the station to reach the highest priority and sending the data as fast as possible. Figure 3 shows a priority update scenario of a station. At the beginning, the station is idle for long time and has the lowest priority m ($m = 8$ in this scenario). Suppose that the station needs to transmit 9 frames to the access point. After $\log_2 8 = 3$ consecutive successful poll, the priority of this station changes to the highest 1. The station gets more polls from the point coordinator and transmit the rest frames rapidly. After the transmissions, the station becomes idle and its priority decreases gradually to the lowest again.

C. The Poll Scheduling Algorithm

In the new polling algorithm, the polling list is conceptually splitted into station lists with different priorities to increase the computation efficiency. All stations with the same priority are placed in the same list, as illustrated in figure 4.

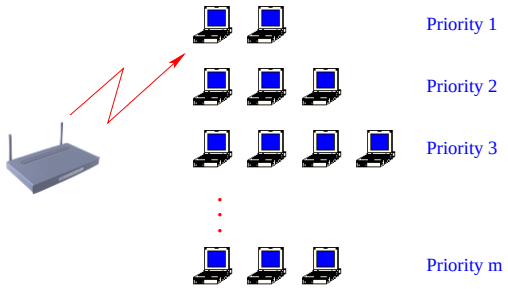

Fig. 4. Station lists with different priorities

The most important goal of this scheduling algorithm is to avoid polling idle stations and allocate more polling slots to stations with heavy traffic. To avoid polling idle stations only means that those stations will be polled less frequently. They will still be polled by the point coordinator within a relatively long duration. If their traffic accumulates, their priorities will be updated because of the successful polls. Accordingly, they will be polled more frequently. To achieve this goal, the polling algorithm provides one more polling slot to stations with priority i than those to stations with priority $i+1$. Supposing the number of stations with priority i ($1 \leq i \leq m$) is n_i and each station with priority m gets one poll slot, then each station with priority i ($1 \leq i \leq m$) will get $m - i + 1$ poll slots.

All poll slots are organized into cycles. In each cycle, stations with priority i ($1 \leq i \leq m$) will get $m - i + 1$ poll slots. It means, if there are any station with priority m, it will get exactly one polling slot during a cycle if its priority remains m during the cycle. If the priority of each station remains the same, the total number of poll slots in one cycle

is approximately $\sum_{i=1}^{m} n_i * (m - i + 1)$. Actually, there are always priority changes in one cycle.

The next step is how to arrange the poll slots for all stations sequentially. One way is to scan all stations and give each station $m - i + 1$ consecutive poll slots. However, this is not a good way. Firstly, the priority for each station is not fixed during one cycle and the number of slots allocated to a station is not predetermined. It depends on the poll feedbacks from the station. Possibly if the access point gets a missed poll from a station with priority 1, the priority of that station will become 2. After one missed poll, the station can only get $m-2$ polling slots instead of $m-1$. Secondly, If consecutive polling slots are given, the resource is not used evenly by all stations. This might change the link characteristic for a flow, like peak throughput, delay, and jitter, etc. Thus the upper layer protocols like TCP will be confused. Therefore, a better way is to intermingle the polling slots for stations with different priorities.

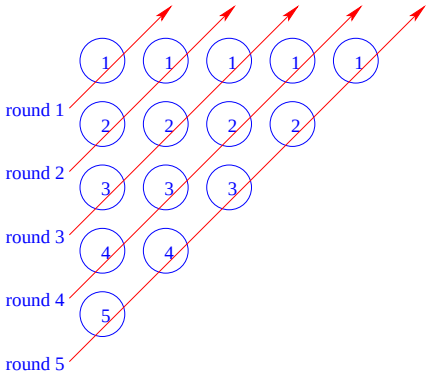

Fig. 5. The station scanning procedure

As depicted in figure 5, We further divide a cycle into m rounds. In round i ($1 \leq i \leq m$), stations with priorities from 1 to i will be polled. The initial value of round counter is m. Then the round counter decreases by one after one round transmission until it reaches 0. The round counter 0 means one cycle transmission is over and it should be reset to m to start a new cycle transmission. The pseudocode to select next pollable station is given below.

NextPoll()
$station = L[listIndex] \rightarrow current$
if $station \neq NULL$ **then**
  $L[listIndex] \rightarrow current = station \rightarrow next$
  return ($station$)
else
  $listIndex - -$
end if
if $lisIndex \leq 0$ **then**
  $round - -$
  $listIndex = round$
  $L[listIndex] \rightarrow current = L[listIndex] \rightarrow head$
end if
if $round \leq 0$ **then**
  $round = m$
  $listIndex = round$
  $L[listIndex] \rightarrow current = L[listIndex] \rightarrow head$
end if

IV. Simulation and Performance Evaluation

In this section, we present the simulation setup and results. We compare the performance of round robin-based polling algorithm and the adaptive polling algorithm proposed in this paper.

A. Simulation Setup

The ns-2 simulator [9] with CMU wireless extensions is used for our simulation. To support the PCF mode and access point in wireless LANs, we patch the ns-2 simulator with the code developed by Lindgren et al. [8]. The PCF implementation is conceptual, but it has all the components we need to implement our poll scheduling algorithm.

The physical radio characteristics of each mobile stations and the access point, such as the antenna gain, transmit power, and receiver sensitivity, were chosen to approximate the IEEE 802.11 Direct Sequence Spread Spectrum (DSSS) radio, at data rate of 2mbps. The PLCP header uses the long preamble and the RTS/CTS is always enabled for DCF mode. The wireless channel is assumed to be error free. Above the physical model, the 802.11 MAC is implemented with the DCF functions such as carrier sense, RTS/CTS, and backoff mechanisms. The patched PCF functions were built on the basic mechanisms of DCF. To concentrate on the performance study of PCF mode, we make the $CFPDuration$ large enough for the PCF mode to dominate the wireless medium.

The network in our simulations consists of a wireless LAN with the access point connected to a wired node via a 10 mbps Ethernet link. The high speed link between the access point and the wired node makes sure the wired network will not become the bottleneck link that affects the wireless performance. The wireless LAN consists of the access point and 10 pollable wireless stations associated with the access point through the wireless links. All stations and the access point are close enough to receive signals from each other. In addition, all stations don't move around.

The traffic in the simulations include 10 constant bit rate (CBR) flows from 10 different stations to the wired node through the access point. These CBR flows send packets at rates of 200 packets per second with packet size 1000 bytes. These flows are started one by one with a 2 second interval. They last 20 seconds and terminate. By this way, we construct a dynamic traffic pattern that can be used to measure how the polling algorithm reacts to the traffic changes in networks. During the low traffic load time, there is only one flow. During the high traffic load time, all 10 flows are being transmitted concurrently. The number of flows being transmitted increase from 1 to 10 gradually.

B. Performance Metrics

The following two metrics are chosen to evaluate the performance of round robin scheduling and the new scheduling algorithm:

- Successful poll rate: the percentage of successful polls over total polls within a given time interval. Successful poll rate is important, since it is the key to increase wireless medium utilization rate. Missed poll means the waste of bandwidth and medium resource.
- Average throughput: measured at the access point. It is the aggregate throughput that goes through the access point within a given time interval.

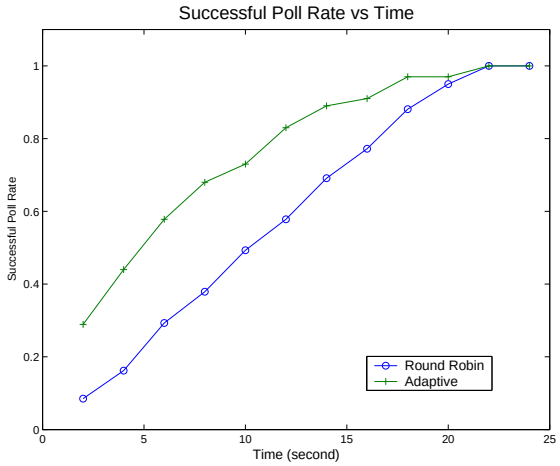

Fig. 6. The successful rate vs time

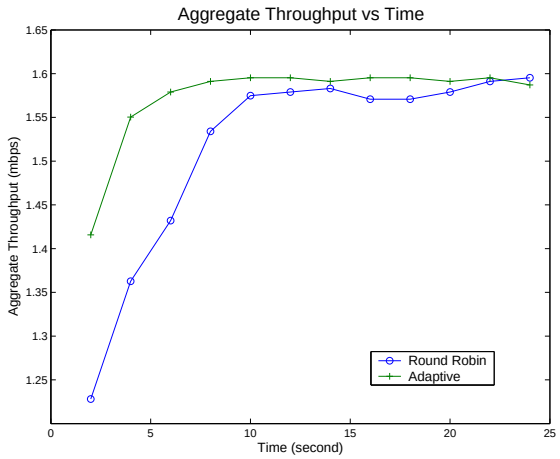

Fig. 7. The aggregate throughput vs time

C. Simulation Results

In the figures of this section, the round robin-based polling algorithm is labeled as "round robin", and the adaptive polling algorithm is labeled as "adaptive".

Figure 7 shows the percentage of successful polls over total polls. We find that, the successful poll rate increases for both poll scheduling schemes, as the number of active stations increases. However, if the percentage of active stations is small, our proposed algorithm performs better than round robin. As the percentage of active stations increases (more CBR flows are started in our simulation), the successful poll rate of round robin-based scheme increases but still below our proposed scheme. If the queueing buffers in all pollable stations are not empty for some time, all station will have the same priority because of the successful polls. In this case, the new polling algorithm functions the same as round robin-based polling algorithm. If all stations are inactive, they will also have the same priority finally because of the missed polls.

As you can see in figure 7, the aggregate throughput of the access point can be improved by approximately 15% in best case. Compared to the significantly improvement of successful poll rate, the aggregate throughput increases relatively slow. The reason is that the PCF mode is not a strict TDMA scheme and the duration of a poll slot is not fixed. If the poll is successful, the slot is long enough to transmit a frame. If the poll is missed, the point coordinator just waits for a short time, and starts to poll other stations.

Our simulation studies show that the performance of wireless LANs are improved in terms of the successful poll rate and the aggregate throughput.

V. Conclusion and Future Work

To increase the wireless medium utilization rate of IEEE 802.11 wireless LANs in PCF mode, this paper present an efficient polling algorithm based on dynamic priority assigned by the point coordinator. The priority of each station is updated regularly based on the recent poll feedback from the station, using the Additive Increase/Multiplicative Decrease algorithm. The simulation results show that our proposed polling algorithm improves the performance of wireless LANs in terms of the successful poll rate and the aggregate throughput of access point. Our future work focuses on the theoretical analysis of the new polling algorithm. It will also be interesting to study how the DCF mode affects PCF performance. That is, when varying the $CFPMaxDuration$ and fixing $CFPRate$, how the PCF performance changes.

References

[1] D.-M. Chiu and R. Jain, Analysis of the Increase and Decrease Algorithms for Congestion Avoidance in Computer Networks, Computer Networks and ISDN Systems, Vol. 17, 1989, pp. 1-14.

[2] A. Feldmann, A. C. Gilbert, W. Willinger and T. G. Kurtz, The changing nature of network traffic: Scaling phenomena, ACM SIGCOMM Computer Communication Review, Volume 28, No. 2, April 1998, pp. 5-29.

[3] S. Floyd, V. Paxson, Difficulties in simulating the Internet, IEEE/ACM Transactions on Networking, vol. 9, no. 4, pp. 392-403, Aug. 2001.

[4] IEEE 802.11 Standard Part II: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, 1999.

[5] Supplement to IEEE 802.11 Standard Part II: Higher-speed Physical Layer Extension in the 5 Ghz Band, 1999.

[6] Supplement to IEEE 802.11 Standard Part II: Higher-Speed Physical Layer Extension in the 2.4GHz Band, 1999.

[7] IEEE 802.11 Working Group, http://grouper.ieee.org/groups/802/11/

[8] Anders Lindgren, Andreas Almquist, Olov Scheln, Quality of Service Schemes for IEEE 802.11: A Simulation Study, 9th International Workshop on Quality of Service (IWQoS 2001), Karlsruhe, Germany, June 6-8, 2001,

[9] http://www.isi.edu/nsnam/ns/

[10] A. K. Parekh and R. G. Gallager, A generalized processor sharing approach to flow control in integrated services networks: The single-node case, IEEE/ACM Transactions on Networking, vol. 1, June 1993.

[11] http://www.rfc-editor.org/rfc/rfc1633.txt