A Token-based Greedy Chain Scheduling Algorithm (T-GCSA) for Situation Aware Wireless LANs[1][2]

Akis Spyropoulos, Cauligi Raghavendra
University of Southern California
Los Angeles, CA 90089-2562
213-740-9133
spyro@halcyon.usc.edu, raghu@amazon.usc.edu

Abstract— In this paper we propose a MAC protocol for situation aware, long range wireless LANs consisting of highly mobile nodes. We choose to schedule nodes' transmissions instead of using contention-based protocols for accessing the channel. This approach is attractive for improving the performance of wireless LANs with long propagation delays. An example of such wireless LANs is a set of planes in a range of hundred miles with radio communications. The problem with contention-based MAC protocols is that each node has to wait for the maximum propagation delay (from one end of the coverage area to the other), before it can decide whether the channel is free or the previous node had no packet to transmit. We propose an "educated" token-based adaptive algorithm that schedules individual nodes transmissions in a way that minimizes the total propagation delay for each round, based on global location information. This way, nodes that are near one another are also adjacent in the schedule, and therefore have to wait significantly less time before they can decide whether it's safe to transmit or not. We call this algorithm "Token-based Greedy Chain Scheduling Algorithm". We compare the Token-based Greedy Chain Scheduling Algorithm with a random scheduling algorithm and a worst-case scheduling algorithm. Simulation results show latency and throughput improvements by a factor of up to 5 to 7 times compared to the random scheduling algorithm.

TABLE OF CONTENTS

1. INTRODUCTION

There exists a plethora of media access protocols for local area networks. These include the earlier Aloha [2] (unslotted and slotted), reservation based protocols [11], and the widely deployed CSMA/CD protocol [3]. There are considerable research results on the performance of these MAC protocols in the literature [10]. For example, the performance of CSMA/CD is highly depended on end-to-end propagation delay and this is the reason why there's a restriction on the maximum cable length for Ethernet. With the explosive growth in laptops and hand-held devices using wireless communications, design of MAC protocols is once again becoming an important research area.

Protocols like CSMA/CD and newer ones based on fiber optics networks like FDDI and DQDB, have been widely deployed for wired LANs and can achieve remarkably high throughputs and channel utilizations close to 100%. However, their application in a wireless environment has not been that straightforward. Mobile environments are governed by phenomena like shadowing, multi-path fading and interference that invalidate crucial assumptions made for wired networks. Further, there is the hidden terminal problem in wireless ad hoc networks [1]. Protocols like MACA [7] and MACAW [8] were proposed for ad hoc networks that use explicit handshake mechanisms. A widely deployed MAC protocol for wireless networks is the IEEE 802.11 standard [9] that is based on the MACA and its variants.

A common thread in all these media access protocols for wireless networks is that they are contention-based. However, contention based protocols are inefficient, costly and therefore undesirable in certain contexts. For example, if energy is a concern for the nodes then a potential collision and a following retransmission is rather costly, and should therefore be avoided. When nodes have predictable or periodic traffic, scheduling algorithms or reservation-based protocols can improve both performance and energy consumption. In many applications there is limited traffic from each node, and therefore, it is more important to design MAC protocols that conserve energy,

[1] 0-7803-7231-X/01/$10.00/© 2002 IEEE
IEEEAC paper #261, Updated Sept 29, 2001
[2] This research is supported by the DARPA Contract F33615-C-00-1633 in the Power Aware Computing and Communications Program

reduce access time when there is a need, and improve channel utilization. Schemes like TDMA and reservation-based protocols will perform quite well for these scenarios. When the nodes are highly mobile and/or the propagation delays involved are quite high, new MAC protocols are needed to achieve the objectives stated above.

In this paper we develop a MAC protocol suitable for highly mobile nodes in a single-hop wireless LAN environment. We have n airplanes/nodes randomly distributed over an area (Figure. 1), the size of which is determined by the maximum range of the airplane radios used. The radios used by airplanes are quite strong, though, and can reach up to 100 miles far. This creates an effective area of about 100*100 miles^2 where each node can hear all other nodes. Each node broadcasts its current position (taken by a satellite system – e.g. GPS) to all other nodes, so that every node has an up to date picture of all the other nodes positions (along with its own). Furthermore, other types of data (e.g. voice) can also be sent over the channel, along with the location update data. All nodes are highly mobile, moving with speeds around 500 miles/hour. Their direction of movement is considered to be arbitrary and can change at any time, a fact that creates a rapidly changing topology. Contention and therefore collisions can destroy or significantly delay important location update (or command and control) info and are highly undesirable. Furthermore, the application itself can also provide valuable information in determining a more efficient way to arbitrate packet transmissions among individual nodes. Contention-free schemes like TDMA are more preferable for this scenario and are not too difficult to implement for relatively small networks.

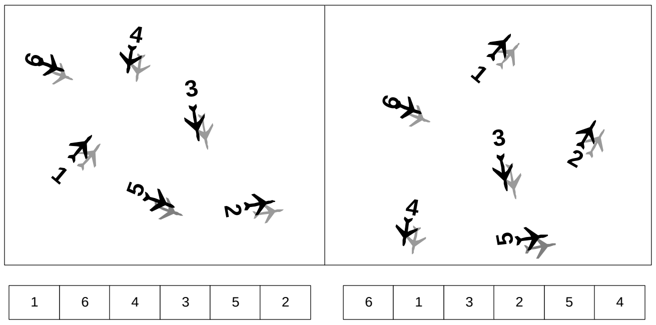


Figure 1 – Positions of jets in two different time instants along with an example of (sub-optimal) schedules for each case

Special care is needed, though, in choosing or designing an appropriate MAC protocol for such an application. The extremely long distances between nodes (a characteristic not found in wired networks) cause accordingly long signal propagation delays. As a matter of fact propagation delay will be higher in many cases than packet transmission time. As in the case of Ethernet (CSMA/CD), end-to-end (and node-to-node) propagation delay will play a major role in determining the media access algorithm’s performance. Careless application of common scheduling or reservation-based schemes in this context may cause the system to perform far from optimal and could prove less effective than contention-based schemes.

In Section 2 we will describe and analyze three different scheduling algorithms, namely the “worst-case random scheduling algorithm”, the “token-based random scheduling algorithm” and the “token-based greedy scheduling algorithm. In Section 3 we will provide simulation results for these algorithms, taken by simulating a large number of random node topologies, and we’ll use those results to evaluate their performance. In Section 4 we will discuss some additional environment-specific and application-specific characteristics that could complicate the problem’s solution and we will outline possible ways to deal with them. Finally, in Section 5 we will conclude the paper and propose some future work to be done.

2. Scheduled Media Access

Our application with n planes has particular characteristics that demand for a careful design of the scheduling algorithm to be used for media access. Some nodes only will have data at any time to send and we want to schedule access to the media to improve performance. There are considerable signal propagation delays involved, which, depending on the size of packets used for transmission, can seriously degrade the performance of the system if overlooked. We therefore need a scheduling scheme that will deal with this propagation delay efficiently.

There are several different scheduling schemes that can be used to arbitrate the access of a shared media. All of these algorithms have in common the fact that they’re designed to prevent contention on the media and therefore avoid collisions. Various performance metrics can be used to measure the effectiveness of media access control algorithms; *throughput* and *latency* are the most widely used. Throughput is the maximum number of packets (or bytes) that can be transmitted over the shared channel per second. Latency, on the other hand, is the amount of time a user has to wait before gaining access to the channel when he or she has a packet to send. For scheduling schemes the *maximum access latency* is directly related to the *total frame duration* and we’ll use them interchangeably to evaluate different algorithm’s performance. Throughput and latency are usually directly proportional, which makes it impossible to optimize for both, simultaneously. Optimizing for throughput will raise the average latency incurred by a single user, while trying to bound latency will probably leave the channel underutilized. There’s an additional performance metric used occasionally to evaluate a media access control scheme, *fairness*. An abstract definition for *fairness* would be that “all users have an equal chance to gain access to the shared media”. Each media access control scheme and, consequently, each scheduled media access control scheme aims at optimizing for one of the above metrics, while retaining an acceptable performance for the other two.

Worst case random scheduling

One straightforward and easy-to-implement approach to schedule individual transmissions over the common channel would be to design for the worst case. An arbitrary schedule is chosen, based perhaps on nodes IDs (e.g. Radio card ID), which could be hardwired or programmed in advance. Each node waits for the previous node in schedule to finish its transmission, before transmitting. However, if the previous node has no packet to transmit, the next node has to wait for the maximum possible propagation time (that is, the time for signal to propagate from one end of the coverage area to the other), in order to be sure that the previous node has no packet to transmit during this frame cycle. Waiting for the maximum propagation time is a commonly used approach in wired networks (e.g. when "sensing" the channel in CSMA protocols) for two major reasons. First, as we already mentioned, it's the safest and simplest thing to do. Second and most important, propagation time in wired LANs is usually bounded by imposing a limit on the maximum physical wire length, thereby securing the protocol's performance. This is not the case, though, for the type of extended area wireless LANs we're considering.

Assuming there are *n* total nodes in the network, it is important to know how many of those nodes on average are actually active (that is, they have a packet to send) during a frame period. If we define the number of active nodes as ***m***, then the *maximum latency* to access the channel or, equivalently, the *total frame duration* will be given by:

$$L_{max} = T_{frame} = m * \frac{k}{B} + n * \frac{\sqrt{2} * D}{c} \qquad (1)$$

K is the number of bits in a packet, $\sqrt{2} * D$ is the network diameter for a square D*D coverage area, B is the radio bandwidth and c is the speed of light in vacuum. The *total network throughput*, which is defined as the maximum amount of traffic (in bps) that can be transmitted over the channel, when there are m active nodes is:

$$\text{Throughput} = \frac{m * k}{m * \frac{k}{B} + n * \frac{\sqrt{2} * D}{c}} \qquad (2)$$

Defining utilization (ρ) as the percentage of time data bits are occupying the channel then:

$$\rho = \frac{\text{throughput}}{\text{channel capacity}} = \frac{m * \frac{k}{B}}{m * \frac{k}{B} + n * \frac{\sqrt{2} * D}{c}} \qquad (3)$$

For an example configuration of k = 512 bytes, B = 10 Mbps, n = 100 and D = 100 miles then, if just one node is active (m = 1), the time to access the channel is 54ms and the channel utilization is only 0.0075, even if the node has packets to transmit all the time. In Figure 2 and Figure 3 we show how channel access latency and channel utilization changes with the number of active nodes:

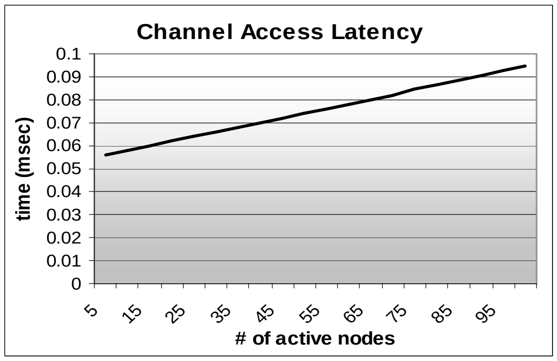


Figure 2 – Channel access latency for the worst-case random scheduling: k=512 bytes, B = 10 Mbps, n =100 and D=100 miles

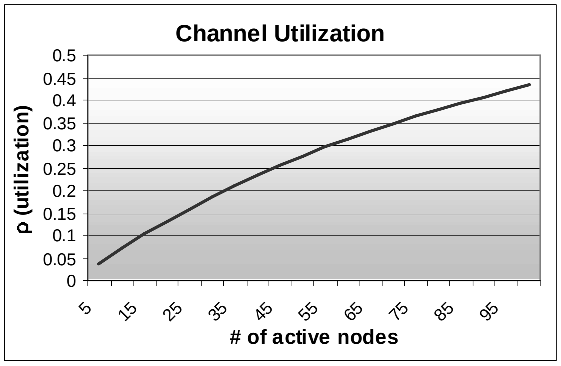


Figure 3 – Channel utilization for the worst-case random scheduling: k=512 bytes, B = 10Mbps, n =100 and D=100 miles

Token-based Random Scheduling

We saw that the simple worst-case random scheduling is extremely inefficient when applied to our scenario. Specifically, for cases where total traffic is contributed by a small number of nodes at a time (m is small), this scheme performs worse than the less sophisticated contention-based media access protocols (i.e. pure Aloha). The reason that the previous scheme performs so poorly is due to the fact that each node has to wait for the maximum possible propagation delay, before deciding whether another node is transmitting or not. This, practically, means that each node has to assume that the preceding node in schedule is always as far away as possible.

An obvious improvement to the random scheduling algorithm is to include a token. A token is a small packet (5-20 bytes) that is exchanged among nodes and is used to arbitrate individual node's transmissions. The node that possesses the token is automatically granted permission to access the channel and transmit. When the node has finished transmitting a packet (or possibly more) or if the node does not have any packet to transmit, it relays the token to the following node in schedule and so forth. This way, the amount of time a node has to wait, before concluding that the previous node does or does not have a packet to transmit, will on average be less than the maximum propagation time.

An important quantity for this scheduling algorithm is the total time it takes for the signal to traverse the path formed by connecting all the nodes according to the transmission schedule. We'll be referring to this path as the "chain" in the rest of this paper. This chain propagation time is given by:

$$T_{prop} = \frac{\text{Chain\_length}}{c} = \frac{\sum_{i=1}^{n} d_{i->i+1}}{c} \tag{4}$$

$d_{i->i+1}$ is the physical distance from node i to the node scheduled to transmit immediately after i.
The *frame duration*, *total throughput* and *channel utilization* for this scheme are therefore given by:

$$L = m * \frac{k}{B} + \frac{\sum_{i=1}^{n} d_{i->i+1}}{c} \tag{5}$$

$$\text{Throughput}_{all} = \frac{m * k}{m * \frac{k}{B} + \frac{\sum_{i=1}^{n} d_{i->i+1}}{c}} \tag{6}$$

$$\rho = \frac{m * \frac{k}{B}}{m * \frac{k}{B} + \frac{\sum_{i=1}^{n} d_{i->i+1}}{c}} \tag{7}$$

As a matter of fact, we can do some further analysis to try to estimate the average inter-node distance for the random scheduling algorithm. This would give us a value of about $\frac{D}{3}$ to $\frac{\sqrt{2} * D}{3}$ for inter-node distance.

We mentioned earlier that there's another type of contention-less media access control protocols, called reservation-based protocols. Nodes make reservations during the *reservation period* and transmit their packet during the *data transmission period*. Although in many cases the reservation-based scheme is very efficient, it doesn't perform well in this context. The reason for this is that the duration of each "mini-slot" within the reservation "sub-frame" should be equal to the maximum propagation time to avoid ambiguity and potential contention among the nodes. That is, a node's reservation should be propagated first throughout the entire network, before another node can reserve the media. We can easily see that this scheme is equivalent to the original worst-case random scheduling scheme, in terms of performance, and is therefore not a candidate for extended area wireless LANs.

Token-based "Greedy Chain" Scheduling

Just by using the token, we managed to improve the performance of the simple worst-case random scheduling algorithm by a factor of about $\frac{3}{\sqrt{2}}$ to 3 (when m or k or both are low). We can do even better by looking more closely at the equations that define the frame length, total throughput, and channel utilization. We saw that for the token-based random scheduling algorithm, the total propagation time is the important quantity in all three equations. Additionally, the smaller the size of packets used for transmission, the more dramatic the propagation time's impact on system's performance. Our goal should be, therefore, to schedule the individual nodes to minimize the total chain length.

The application itself running over this (extended range) wireless LAN is a useful ally in our effort to minimize total propagation time. As described earlier, part of the data exchanged between jets is their actual locations. Each node has, therefore, global knowledge of the network topology and our problem is to find the shortest way to traverse all nodes, starting from an arbitrary node (e.g. the one with lowest ID). This is the well-known Traveling Salesman Problem [5], a very popular problem in the Algorithms literature (specifically Graph Theory) and is known to be NP complete. There are non-optimal solutions available for this, the most widely known of which may be Christofedes algorithm [4]. Our concern, however, is not solely to minimize total propagation time, but to improve the overall system performance. The amount of improvement we can achieve is bounded by the time it takes to transmit a packet. Therefore, the degree of sophistication of the solution we will choose for reducing the total path length should be dependent on the impact on overall system performance.

We propose a simple heuristic to schedule nodes transmissions, which can be easily implemented in a real network and performs close to optimal for most of the scenarios we consider. It's a greedy type algorithm that starts from an arbitrary node (e.g. lowest ID node) and jumps to the nearest node not already traversed, until all nodes are included in the path. Below, we provide a pseudo-code for the greedy algorithm.

```
Set N = {node 0, node 1,…,node N}
Set S = {node 0}
Preceding node = node 0
While $S \neq N$
        begin
        next = node nearest to preceding node among nodes $\notin S$
        S = S U {next}
        Preceding node = next
end
```

Furthermore, we'll keep the token as a means of synchronization among the nodes, without which timing issues could seriously impede the application of our scheduling algorithm. Each node uses the last position updates of all the other nodes to recalculate the schedule after each complete transmission cycle. This means that nodes re-calculate the schedule for every frame. Although the nodes we consider are highly mobile (speeds of 500miles/hour), they can only move in the range of few 10s or 100s of feet within a frame's period, which is quite low relative to the inter-node distances, which are more than 1 mile. Consequently it's relatively safe to base the scheduling for the next frame on the nodes locations from the previous update (which may or may not have occurred during the last frame period). Since all nodes can "hear" everyone, and all nodes perform the same algorithm to compute the schedule, this scheme will work correctly. It would only produce a sub-optimal schedule for one cycle, which would be fixed when the next location update is available.

The performance equations for the greedy algorithm are the same as in the case of the token-based random scheduling algorithm. The algorithm's performance depends on the topology of the nodes and in particular on the node density. If, for example, n nodes were positioned on a D*D 2-dimensional grid, the optimal chain length would be $n * \frac{D}{\sqrt{n}} = \sqrt{n} * D$, which is the bound to our algorithm's performance. Notice that this gives a factor of $\sqrt{2n}$ improvement over the original worst-case random scheduling scheme (which has an equivalent total path length of $n * \sqrt{2} * D$). For an arbitrary distribution of nodes it's difficult to estimate the algorithm's performance and we will use simulation to evaluate our scheme.

3. Simulation Results

We saw that the system's performance is defined in every case by the total frame duration:

$$T_{frame} = m * \frac{k}{B} + \frac{\sum_{i=1}^{n} d_{i -> i+1}}{c} \qquad (8)$$

We can only try, however, to reduce the total chain length, since packet length and radio bandwidth are not an optimization target, but rather input parameters to the problem. The impact of a potential reduction in total frame length depends on the transmission time component of the frame ($m * \frac{k}{B}$) and therefore on the amount of active nodes (nodes that have packets to transmit), on packet size and on radio bandwidth. The applicable packet sizes are between 64 bytes and 512 bytes, while the radio bandwidth is 1-10Mbps. Finally, the amount of improvement of the token-based greedy algorithm over the token-based and worse-case random scheduling algorithms also depends on the node density. The denser the network is the better the chances are for the greedy algorithm to perform closer to optimal. For our simulation we assume that nodes are randomly distributed on a 100*100 miles^2 square area and the minimum inter-node distance is 1 mile. Results are averaged over 100 runs for different random topologies and different random schedules.

In Figure 4, we show the total "idle" frame duration (that is a frame without any transmissions occurring) for all three scheduling algorithms and for different network densities. We can see that for the two random algorithms the frame duration increases linearly with the number of nodes in network, as predicted by our previous analysis. However, the greedy algorithm performs better with higher numbers of nodes. The reason for that is that the denser the network the more flexible our algorithm when making decisions and the lower the penalty for making a sub-optimal decision. The simulation results confirm our analysis that estimated the scaling factor of the path produced by the greedy scheme as $\sqrt{n}$, where n is the number of nodes. The increased scalability of the greedy algorithm results in a 20 times ($\sqrt{2 * 200}$) and 7.5 times improvement over the worst-case and token-based random algorithms, respectively, for a 200 nodes topology. Furthermore, we can observe the anticipated 3 times improvement of the token-based random scheduling over the worse-case random scheduling.

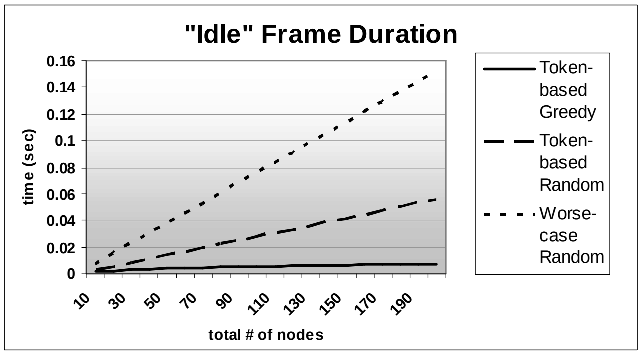


Figure 4 – Idle frame duration for the worse-case random, token-based random and token-based greedy scheduling algorithms

Because the worst-case random scheduling scheme is heavily outperformed by the greedy algorithm, we'll concentrate on the two token-based schemes. Figure 5 depicts the relative performance of the token-based greedy scheduling over the token-based random scheduling for different amounts of network activity. We assume 512 bytes packets and 10Mbps radio bandwidth.

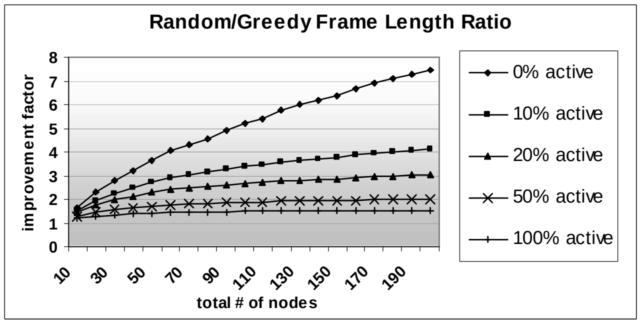


Figure 5 – Token-based random over token-based greedy frame length reduction for different node activity

We can see from the graph above that the performance gain of the greedy algorithm over the random one is becoming less prominent for increasing network traffic. The reason for this is due to the fact that more network traffic means that the transmission time component becomes a larger part of the total frame time, therefore reducing the impact of the propagation time in system performance.

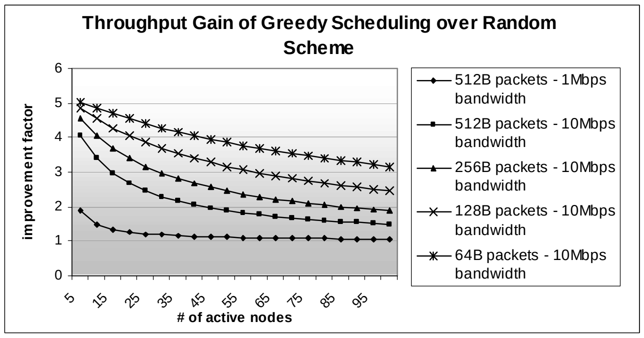
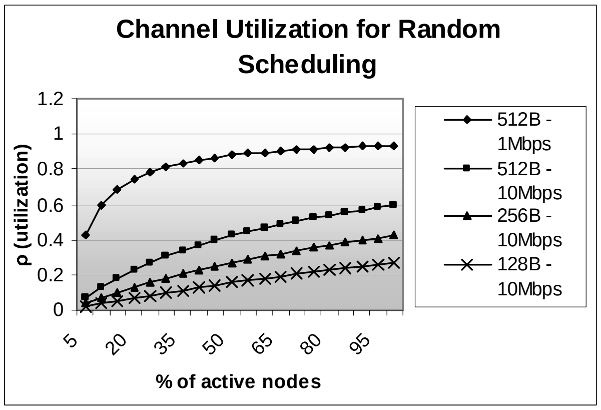


Figure 6 – Throughput gain of a token-based greedy scheduling over a token-based random scheduling for 100 node network

In Figure 6, we show how packet size and bandwidth affect the relative performance gain of the greedy algorithm (or any other algorithm) over a less sophisticated one like the token-based random algorithm. We assume a network size of 100 nodes. There are a couple of observations we can make regarding the system's behaviour for different packet sizes:

i. Packet size used impacts the maximum performance improvement achievable by the greedy scheduling algorithm. Smaller packets imply propagation time is a larger part of the total frame time, and therefore reduced propagation time will have a larger impact on the overall system performance. This is, essentially, Amdahl's law [6] applied to this particular system.
ii. For low node activity the improvement factor on total throughput converges, for increasingly smaller packet sizes, to the absolute improvement factor in propagation time of the greedy algorithm.

Finally, we claimed during our analysis in section 2 that our heuristic, albeit simple, performs close to optimal (regarding overall system performance) for most cases of interest and that the potential use of a more sophisticated heuristic for the Traveling Salesman Problem could not achieve any significant improvement on our system. In Figure 7, we show the channel utilization levels achieved by the token-based random scheduling and the token-based greedy scheduling algorithms for several scenarios.



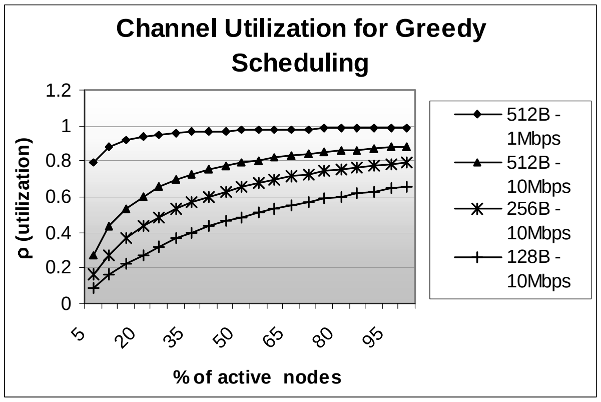


Figure 7 – Channel Utilization for the token-based random scheduling and token-based greedy scheduling algorithms

It is apparent that for large packet sizes or low radio bandwidth (i.e. 512bytes packets and 1Mbps radio bandwidth) the random scheduling algorithm performs relatively well, since it achieves a utilization of 80-100% most of the time (>20-30% active nodes). For such a configuration, there's not much room for improvement. A better algorithm can only push the curve a little up (something that the greedy algorithm actually does), closer to the 100% utilization bound, but obviously not any further. It is for smaller packet sizes that the random scheduling reveals its weaknesses. The greedy algorithm can still achieve high channel utilization levels (>50-60% for most cases), even for really small packet sizes, showing significant improvement over the random one. One could argue that 50-60% or less (for low percentages of active nodes) is certainly not optimal. We have to keep in mind however, that, for small packet sizes, the overall system performance is constrained by the total chain length. The total chain length is finite and cannot be as low as zero. This means that channel utilization can never be one, but will be bounded by the value achieved for the shortest possible path through all the nodes. This shortest path is an NP-complete problem to calculate and even the best heuristic produces a path whose length is two times the value of the optimal one. This gives us enough evidence that the token-based greedy scheduling algorithm is acceptably close to the best we could do for a practical range of values applicable to the situation aware communication application.

4. WIRELESS ENVIRONMENT AND APPLICATION SPECIFIC CONSIDERATIONS

Apart from the basic problem dealt with so far, there are some additional issues to consider stemming from the nature of wireless media and certain application specific characteristics.

One of the problems we haven't considered yet is what happens when new nodes move "into range" or existing nodes move "out of range". Our token-based algorithms are flexible enough to handle potential additions or departures of nodes, because the frame length is not fixed and can accommodate any number of nodes. Furthermore, existing nodes always know the global topology and can detect arrivals or departures of nodes automatically. The only problem is how all nodes can make a common decision regarding when a new node is in or an existing one is out of range. It is obvious that some nodes will detect a newcomer faster than others, something that could cause inconsistencies in the schedules calculated by different nodes. A possible solution would be for every node to have a common definition of the coverage area and consider that a node is in or out of range when a node crosses the boundaries of this particular area and not when they come in or out of their own radio range.

The wireless environment is widely considered as one of the "roughest" environments for communications. Phenomena like *shadowing*, *multi-path fading* and *interference* have often been a problem for many protocol designers for wireless cellular and ad-hoc networks. Fortunately, in our case the high elevation of the jets permits us to assume LOS (Line Of Sight) communication between nodes and ignore any shadowing or multi-path fading effects. However, we still have to deal with interference, like noise or jamming, which can be quite high, since this is a military application we're considering and we have to take into account combat field situations. An important implication of the existence of interference is that it may cause the token to be lost. The nodes should be able to recover from such a token loss and continue with the regular schedule in a short amount of time, since fault-tolerance is crucial in military applications. A simple way to make our algorithm more robust and able to handle token losses is to add timeouts. A node waits first to hear a packet transmission or a token transmission from the previous node in schedule, before it can transmit. If neither of these events occurs after a certain amount of time, the node can timeout and can assume it has permission to access to the channel. If it does not have a packet to send, it creates a new token and passes it to the next one. A safe timeout value would be as large as the end-to-end propagation delay plus the transmission time for the maximum sized packet. If just a few token losses occur per frame, such timeouts would not degrade the system's performance considerably. If, on the other hand, interference is high and token losses may occur very often, smaller timeout values based on node-to-node propagation delays instead of maximum propagation delays, would perform better. Furthermore, a potential loss of a location update packet due to interference, as we explained earlier, is not crucial to our algorithm's operation and is a self-healing event.

5. CONCLUSIONS AND FUTURE WORK

In this paper we explained how large propagation delays can degrade the performance of media access protocols for extended area wireless LANs, like the one used for communication among jet fighters. We also justified why special attention is needed when designing media access algorithms for such environments. We described a simple worst-case random scheduling algorithm that only caters for correctness, but not for efficiency, and proposed two token-based scheduling algorithms, a random schedule and a greedy chain based schedule, that try to solve the problem more efficiently. We proved using analysis and simulation that the token-based greedy scheduling algorithm performs up to 20 times better from the worst-case random scheduling and up to 7 times better than the token-based random scheduling algorithm.

Finally, when packet sizes are small or radio bandwidth is high, additional improvement is possible. This will become increasingly important in the future, as wireless radio data transmission rates get even higher, while data packet sizes practically remain unchanged. Then it could be worthwhile to consider more sophisticated heuristics to approximate the shortest chain length.

REFERENCES

[1] F. A. Tobagi and L. Kleinrock, "Packet switching in radio channels, Part II: The hidden-terminal problem in carrier sense multiple access and the bus-tone solution", *IEEE Transactions in Communications., COM-23, pp.1417-1433,* 1975.

[2] L.G. Roberts. "Aloha Packet System with and without Slots and Capture," *ASS Notes 8, Advanced Research Projects Agency,* Network Information Center, Stanford Research Institute, Stanford, CA, 1972.

[3] IEEE Standard 802.3-1985. "Carrier Sense Multiple Access with Collision Detection CSMA/CD", 1985.

[4] N. Christofedes, "Graph Theory: An Algorithmic Approach," Academic Press Inc., 1975.

[5] http://www.math.princeton.edu/tsp/

[6] J.L. Hennessey and D.A. Patterson, "Computer Architecture: A Quantative Approach, " Morgan Kaufman Publishers, Inc. 1990.

[7] P. Karn., "MACA--a new channel access method for packet radio," *In Proceedings of the 9th ARRL/CRRL Amateur Radio Computer Networking Conference,* September 1992.

[8] V. Bharghavan, A. Demers, S. Shenker, and L. Zhang, "MACAW: A media access protocol for wireless LANs," *In ACM SIGCOMM '94, pages 212--225*, August 1994.

[9] IEEE Standards Department, Wireless LAN medium access control (MAC) and physical layer (PHY) specifications, IEEE standard 802.11-1997, 1997.

[10] N. Shacham and V. B. Hunt, "Performance evaluation of the CSMA/CD (1-persistent) channel-access protocol in common-channel local networks," *In Local Computer Network, IFIP, pages 401--414.* North-Holland, 1982.

[11] K. Eternad, "Enhanced random access and reservation scheme in CDMA2000", *IEEE Personal Communications, Vol. 8, issue 2, pages 30-36,* April 2001