

# A Wireless Token Ring Protocol for Ad-Hoc Networks

Duke Lee, Anuj Puri, Pravin Varaiya  
 {duke,anuj,varaiya}@eecs.berkeley.edu  
 Department of Electrical Engineering  
 University of California at Berkeley, CA 94720  
 (510) 642-5649, 643-5889, 642-6330

Raja Sengupta  
 sengupta@eecs.berkeley.edu  
 Dept of Civil and Environmental Engineering  
 University of California at Berkeley, CA 94709  
 (510) 231-9484

Roberto Attias  
 roberto@teja.com  
 Teja Technologies, Inc.  
 2 West Santa Clara St, 6th Floor, San Jose, CA 95113  
 (408) 288-2560

Stavros Tripakis  
 tripakis@imag.fr  
 VERIMAG, Centre Equation  
 2, avenue de Vignate, 38610 GIERES, FRANCE  
 +33(4)-76-63-48-43

*Abstract*—The Wireless Token Ring Protocol (WTRP) is a medium access control protocol for wireless networks in Unmanned Aerial Vehicles. It supports quality of service in terms of bounded latency and reserved bandwidth. This quality of service guarantee is critical in mesh stability of the formation of the vehicles. The communication of the speed and the velocity of the lead vehicle to all other vehicles in the formation had been shown to be sufficient for mesh stability of the system.

WTRP is efficient in the sense that it reduces the number of retransmissions due to collisions. It is fair in the sense that each station takes a turn to transmit and is forced to give up the right to transmit after transmitting for a specified amount of time. It is a distributed protocol that supports many topologies since not all stations need to be connected to each other or to a central station. It can be used with an admission control agent for bandwidth or latency reservations. WTRP is robust against single node failure. WTRP is designed to recover gracefully from multiple simultaneous faults.

*Keywords*—Medium Access Control, Wireless Token Ring Protocol, IEEE802.11, IEEE802.4, Quality of Service, Intelligent Transportation Systems, Automated Highway Project, Mission Critical Systems.

## 1. INTRODUCTION

In an unreliable medium such as wireless, problem of providing quality of service (QoS) at the network layer using queuing and routing techniques is not sufficient. QoS must also be addressed at the data-link layer. The IEEE 802.11[?] in PCF (Point Coordination Function) mode, the HiperLAN[?], and Bluetooth[?] achieve bounded latency by having a central station poll the slave stations. Most academic research has focused on this centralized approach [?] [?]. The centralized approach is suitable for networks where only the last hop is wireless. In the centralized ap-

proach, the network is managed centrally from a central station. This is a limitation in wireless networks with ad-hoc topologies.

The Wireless Token Ring Protocol (WTRP) discussed in this paper is a distributed medium access control protocol for ad-hoc networks. The advantages of a distributed medium access control protocol are its robustness against single node failure, and its support for flexible topologies, in which nodes can be partially connected and not all nodes need to have a connection with a master.

WTRP is to be deployed initially in the University of California at Berkeley PATH Advanced Vehicle Safety Systems Program[?], the CALTRANS-PATH Demonstration 2002, and the Berkeley Aerobot project[?]. These projects impose stringent bandwidth, latency, and speed of failure recovery requirements on the medium access protocol. The platoon mode of the automated highway project involves up to 20 nodes in each platoon, and requires that information (approximately 100 bytes per vehicle for speed, acceleration, and coordination maneuvers) be transmitted every 20ms. The failure recovery time for the communication system must be within 40ms.

As in the IEEE 802.4[?] standards, WTRP is designed to recover from multiple simultaneous failures. One of the biggest challenges that the WTRP overcomes is partial connectivity. To overcome the problem of partial connectivity, management, special tokens, additional fields in the tokens, and new timers are added to the protocol. When a node joins a ring, it is required that the joining node be connected to the prospective predecessor and the successor. The joining node obtains this information by looking up its connectivity table. When a node leaves a ring, the predecessor of the leaving node finds the next available node to close the ring by looking up its connectivity table. Partial connectivity also affects the multiple token resolution protocol (deleting all multiple tokens but one). In a partially connected network, simply dropping the token whenever a station hears another transmission is not sufficient. To delete tokens that a station is unable to hear, we have designed

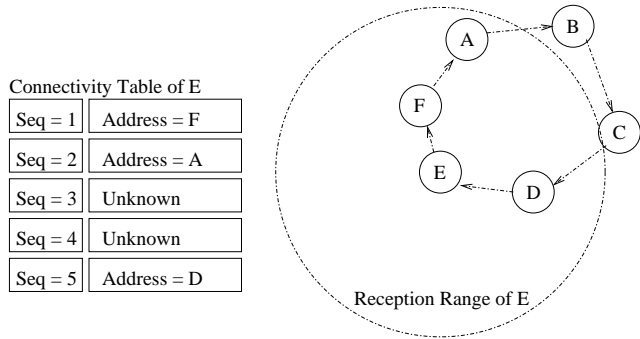


Figure 1. Connectivity Table

a unique priority assignment scheme for tokens. Stations only accept a token that has greater priority than the token the station last accepted. The WTRP also has algorithms for keeping each ring address unique, to enable the operation of multiple rings in proximity.

2. PROTOCOL OVERVIEW

In the WTRP, the successor and the predecessor fields of each node in the ring define the ring and the transmission order. A station receives the token from its predecessor, transmits data, and passes the token to its successor. Here is an illustration of the token frame.

FC	RA	DA	SA	Seq	GenSeq
1	6	6	6	4	4

bytes

FC stands for Frame Control and it identifies the type of packet, such as Token, Solicit Successor, Set Predecessor, etc. In addition, the source address (SA), destination addresses (DA), ring address(RA), sequence number(Seq) and generation sequence(GenSeq) number are included in the token frame. The ring address refers to the ring to which the token belongs. The sequence number is initialized to zero and incremented by every station that passes the token. The generation sequence number is initialized to zero and incremented at every rotation of the token by the creator of the token.

The Connectivity manager resident on each node tracks transmissions from its own ring and those from other nearby rings. By monitoring the sequence number of the transmitted tokens, the Connectivity Manager builds an ordered list of stations in its own ring. The Connectivity Table of the manager holds information about its ring (Figure ??). In Figure ??, station E monitors the successive token transmission from F to D before the token comes back to E. At time 0, E transmits the token with sequence number 0, at time 1, F transmits the token with the sequence number 1, and so on. E will not hear the transmission from B and C, but when it hears transmission from D, E will notice that the sequence number has been increased by 3 instead of 1. This indicates to E that there were two stations that it could not

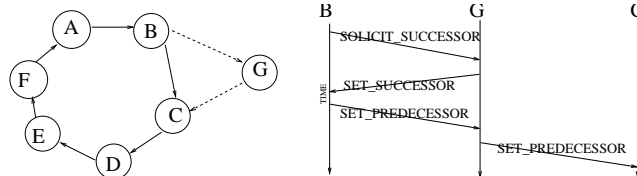


Figure 2. Joining

hear between A and D.

The Ring Owner is the station that has the same MAC address as the ring address. A station can claim to be the ring owner by changing the ring address of the token that is being passed around.

Stations rely on implicit acknowledgements to monitor the success of their token transmissions. An implicit acknowledgement is any packet heard after token transmission that has the same ring address as the station.

Each station resets its IDLE\_TIMER whenever it receives an implicit acknowledgement. If the token is lost in the ring, then no implicit acknowledgement will be heard in the ring, and the IDLE\_TIMER will expire. When the IDLE\_TIMER expires, the station generates a new token, thereby becoming the owner of the ring.

To resolve multiple tokens (to delete all tokens but one), the concept of priority is used. The generation sequence number and the ring address define the priority of a token. A token with a higher generation sequence number has higher priority. When the generation sequence numbers of tokens are the same, ring addresses of each token are used to break the tie. The priority of a station is the priority of the token that the station accepted or generated. When a station receives a token with a lower priority than itself, it deletes the token and notifies its predecessor without accepting the token. With this scheme, it can be shown that the protocol deletes all multiple tokens in a single token rotation provided no more tokens are being generated[?].

The ring recovery mechanism is invoked when the monitoring node decides that its successor is unreachable. In this case, the station tries to recover from the failure by reforming the ring. The strategy taken by the WTRP is to try to reform the ring by excluding as small a number of nodes as possible. Using the Connectivity Manager, the monitoring station is able to quickly find the next connected node in the transmission order. The monitoring station then sends the SET\_PREDECESSOR token to the next connected node to close the ring.

WTRP allows nodes to join a ring dynamically, one at a time, if the token rotation time (sum of token holding times per node, plus overhead such as token transmission times) would not grow unacceptably with the addition of the new

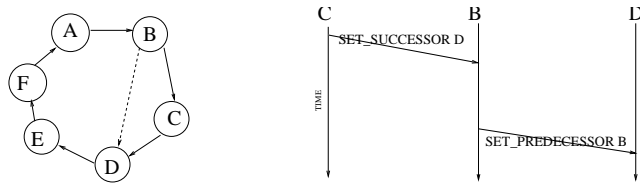


Figure 3. Exiting

node. As illustrated in Figure ??, suppose station G wants to join the ring. Let us also say that the admission control manager on station B invites another node to join the ring by sending out a SOLICIT\_SUCCESSOR. The Admission Control Manager waits for the duration of the response window for interested nodes to respond. The response window represents the window of opportunity for a new node to join the ring. The response window is divided into slots of the duration of the SET\_SUCCESSOR transmission time. When a node that wants to join the ring, such as G, hears a SOLICIT\_SUCCESSOR token, it picks a random slot and transmits a SET\_SUCCESSOR token. When the response window passes, the host node, B can decide among the slot winners. Suppose that G wins the contention, then the host node passes the SET\_PREDECESSOR token to G, and G sends the SET\_PREDECESSOR to node C, the successor of the host node B. The joining process concludes.

As shown in Figure ??, suppose station C wants to leave the ring. First, C waits for the right to transmit. Upon receipt of the right to transmit, C sends the SET\_SUCCESSOR packet to its predecessor B with the MAC address of its successor, D. If B can hear D, B tries to connect with D by sending a SET\_PREDECESSOR token. If B cannot hear D, B will find the next connected node, in the transmission order, and send it the SET\_PREDECESSOR token.

The ring address of a ring is the address of any one of the stations in the ring. The station is called the owner of the ring. In the example, the owner of ring A is station A. Because we assume that the MAC address of each station is unique the ring address is also unique. The uniqueness of the address is important, since it allows the stations to distinguish between messages coming from different rings.

To ensure that the ring owner is present in the ring, when the ring owner leaves the ring, the successor of the owner claims the ring address and becomes the ring owner. The protocol deals with the case where the ring owner leaves the ring without notifying the rest of the stations in the ring as follows. The ring owner updates the generation sequence number of the token every time it receives a valid token. If a station receives a token without its generation sequence number updated, it assumes that the ring owner is unreachable and it elects itself to be the ring owner.

### 3. PROOF OF TOKEN RING PROTOCOL

A more formal proof can be found in [?].

#### Assumptions

1. No transmission error occurs after  $t$ .
2. Topology remains constant after  $t$ .
3. Stations do not voluntarily go out of a ring (into OFFRING state).
4. If station  $x$  is in transmission range of station  $y$ , then  $y$  is also in the transmission range of  $x$ .
5. If station  $x$  is in reception range of station  $y$ , then  $y$  is also in the reception range of  $x$ .

#### Constants

1. MTRT: Maximum Token Rotation Time. In this proof, MTRT is the maximum time it takes for a token to visit a station twice.
2. IDLE\_TIME: amount of time that a station waits before regenerating a token when the medium is quiescent.  $IDLE\_TIME \geq MTRT$
3. INRING\_TIME: amount of time that a station waits when the station is not receiving any acceptable token, before going into the OFFRING state.  $INRING\_TIME \geq IDLE\_TIME \geq MTRT$   
 $INRING\_TIME < 2 IDLE\_TIME$

#### Definitions

1.  $r_i(t)$ : a set of nodes in ring  $i$ .
2.  $t_i(t)$ : a set of tokens in  $r_i(t)$ .
3. Token  $p$  and token  $q$  are said to be equivalent when their ring addresses are the same.
4.  $x.NS(t)$ : MAC address of the station to which station  $x$  forwards tokens.
5.  $x.PS(t)$ : MAC address of the station from which station  $x$  accepts tokens.
6.  $|V(t)|$ : number of the stations in the graph at time  $t$ .

#### Network

##### 1. Definitions

(a) For stations  $x$  and  $y$ , if  $x.NS = y$ ,  $y.PS = x$ , and if  $y$  can receive and accepts NORMAL tokens from  $x$  then, we say that  $x$  has the bijection with  $y$ .

(b)  $S = \{ \langle x.y \rangle \mid x \text{ has the bijection with } y \}$

(c) A set of stations,  $r_i(t)$ , is called a ring if for all  $x \in r_i(t)$ ,  $x$  has the bijection with its successor,  $y$ .

#### Summary

We first prove that all equivalent tokens are deleted by multiple token resolution protocol in finite time. Then we prove that the bijection that represents correct relationship between predecessor and successor for the exact definition) increases monotonically to the number of nodes in the graph in finite time. When the number of bijection converges, then we say that all rings are correct since rings are defined by predecessor and successor relationship.

*Proof*

*Lemma 1:* Choose any token  $p$  at time  $t = t_0$ , and build an ordered list of paths taken by  $p$ , say  $\langle (x_0, t_0), (x_1, t_1), (x_2, t_2), \dots, (x_m, t_m) \rangle$ , where  $t_n$  is the time that token  $p$  visits the station  $x_n$ , and  $t_{i+1} > t_i$ . If there exists a station  $x_i = x_j$  in the pair list such that  $0 \leq i < j \leq m$  and  $t_j - t_i < MTRT$ , then there must be a  $k$  such that  $i \leq k \leq j$ , and  $x_k$  owns  $p$ .

Suppose we find  $x_i = x_j$  such that  $0 \leq i, j \leq m$  and  $t_j - t_i < MTRT$ , but we cannot find the owner of token  $p$ ,  $x_k$ , such that  $i \leq k \leq j$ . This means that the generation sequence numbers of the token when it arrives at  $x_i$  and the generation sequence number when it arrives at  $x_j$  is the same, because no station other than the owner of the token modifies the generation sequence number.

$x_j$  could not have been in the OFFRING state at any time since  $t_i$ , because  $x_i$  could not have been able to rejoin another ring after exiting a ring for one MTRT, by implementation. Because a station is not allowed to receive a token when it is in the OFFRING state, it could not have received  $p$  before time  $t_i + MTRT$ . Thus,  $x_i$  could not have been in the OFFRING state since time  $t_i$ . By implementation, the priority of a station can only increase while it is not in the OFFRING state and thus, it does not make sense that token  $p$  could have survived station  $x_j$ . ■

*Lemma 2:* If no multiple equivalent tokens exist at time  $t$ , then no multiple equivalent tokens exist at time  $s > t$ .

Suppose that there exist multiple equivalent tokens at time  $s$  when there were no multiple equivalent tokens at time  $t$ , such that  $s > t$ . Because of our assumption, it is impossible for a station to generate multiple equivalent tokens from transmission errors. Then a station must have generated a token when a token that it has previously generated is still in the graph. But the IDLE\_TIME is greater than MTRT. And we know that a token dies when it doesn't see its owner for MTRT. Thus the station could not have generated the equivalent token when the token that it has previously generated is still in the graph. ■

*Lemma 3:* No multiple equivalent tokens exist at time  $t + 2MTRT$ .

All surviving equivalent tokens will go through the owner of the token in one MTRT by definition. After one MTRT, the owner will remember the highest priority token among them. Within the next MTRT, all or all but one equivalent token will be deleted, because the owner will not pass any token that has a lower priority than the highest priority token that it received.

If the owner of the token leaves the ring at any time, all tokens will be deleted since the owner is not able to come back to a ring in less than one MTRT. ■

*Lemma 4:* There exist a time  $s$ , such that  $s < t + 2MTRT$  and no multiple equivalent tokens exist any time  $u > s$ .

Follows from ?? and ??. ■

*Lemma 5:* If station  $x$  has the bijection with its successor  $y$ , then the INRING\_TIMER of  $x$  goes off before  $y$ .

The fact that  $x$  has the bijection with  $y$  shows that the last token  $y$  accepted was from  $x$ . Then  $x$  must have reset its INRING\_TIMER before  $y$  had. Thus, INRING\_TIMER of  $y$  cannot go off before  $x$ . ■

*Lemma 6:* When a station goes out of ring (into the OFFRING state),  $|S|$ , the number of the bijections, does not decrease.

Let us say the predecessor of  $y$  is  $x$ , and the successor of  $y$  is  $z$ . When station  $y$  goes into the OFFRING state, it forms a ring of its own by the following assignment:  $y.NS = y$  and  $y.PS = y$ . We distinguish the two cases where a station can be kicked out. The first case is when the INRING\_TIMER expires. In this case, from the lemma ??,  $x$  could not have the bijection with  $y$ , because if it did, the INRING\_TIMER of  $x$  would have gone off before  $y$ . In this case, regardless of whether  $y$  had bijection with  $z$  or not,  $|S|$  will not decrease, because in the worst cases  $|S|$  stays the same if  $y$  had the bijection with  $z$ . The second case is when  $y$  is kicked out because it is not successful in finding a successor. Again regardless of whether  $x$  had the bijection with  $y$  or not,  $|S|$  will not decrease, because in the worst case we lose the bijection from  $x$  to  $y$ , but gains a self-bijection of  $y$ . ■

*Lemma 7:* When a station accepts a token some time after  $t$ ,  $|S|$ , the number of the bijections, does not decrease.

After accepting a token, station  $y$  goes into the OFFRING state, attempts to pass the token to its successor, or sends the SOLICIT\_SUCCESSOR token. According to lemma ??,  $|S|$  is not decreasing in the first case when the station involuntarily goes into the OFFRING state. So we are left to prove that  $|S|$  is not decreasing in the last two cases where station  $y$  attempts to pass the token to its successor, or sends the SOLICIT\_SUCCESSOR token.

Let us see what happens if  $y$  decides to pass the token to its successor,  $z$ . If  $y$  has the bijection with  $z$ , then it will pass the token successfully and there will be no change in the network. In the case that  $y$  does not have the bijection with  $z$ ,  $y$  will try to find a station to form the bijection with. If  $y$  is not successful within a certain window of time, it will go into the OFFRING state. And we have already shown in the lemma ??, that  $|S|$  does not decrease. Now Let us consider the case where  $y$  successfully finds a station to form the bijection with.  $U$  is the station that  $y$  finally forms the bijection with.  $W$  is the predecessor of  $u$ , before  $y$  became its predecessor. Suppose  $w$  had the bijection with  $u$ , before

y came along. Then  $|S|$  is the same as before because we gained one bijection from  $x$  to  $u$ , but lost one from  $w$  and  $u$ . If there was no bijection from  $w$  to  $u$  to begin with, we would have gained on  $|S|$  by one.

Now Let us see what happens if  $y$  decides to send a SOLICIT\_SUCCESSOR token. If no station wins the contention, then  $y$  will proceed to pass the token to its successor. And we have already shown in the previous paragraph that  $|S|$  does not decrease. If station  $z$  wins the contention and successfully sends the SET\_SUCCESSOR token, then  $y$  now has the bijection with  $z$ . Station  $z$  inherits the generation sequence number, the ring address, and the PS pointer from  $y$ , allowing successful establishment of the bijection with  $w$  the successor of  $y$ . If  $y$  did not have the bijection with  $w$ , then at worst case,  $|S|$  will stay the same. ■

*Lemma 8:* A ring will not break after time  $t + 2MTRT + INRING\_TIME$ , and the number of stations in the ring will not decrease.

A station updates its NS pointer when it is unable to pass the token to its successor, leaves the ring, or receives SET\_SUCCESSOR. Because all stations in a ring have the bijections with its successor, it does not make sense that it is unable to pass the token to its successor. Thus, no station will be kicked out of the ring. Also, according to our assumption, a station will not leave the ring voluntarily. When a station receives the SET\_SUCCESSOR, the NS pointer of the station will change. However, the ring will still not break since all contending stations must have a connection with the successor of the soliciting station.

A station updates its PS pointer when it accepts a token from a station different from its predecessor. A station may accept the SET\_PRED token from another station, if the station has the same ring address, causing the ring to break. However, a station in the ring could not possibly have received a token from a station in the same ring that is not its predecessor since all stations in the ring has the bijection with its successor and could not have failed to pass a token to its successor. Moreover, from our assumption, a station is not allowed to leave the ring voluntarily and induce its predecessor to generate SET\_PRED.

Now we will show after  $t + MTRT + INRING\_TIME$ , a station outside a ring cannot possibly send a SET\_PRED token to a station inside the ring with the same ring address. Let us suppose that station  $y$ , with the ring address  $B$ , receives a token from station  $w$ , outside of the ring. Let us label the predecessor of station  $y$ ,  $x$ . From time  $t + MTRT$  and on, no more multiple equivalent tokens exist according to lemma ???. A station cannot possibly remember about a token that did not exist at time  $t + MTRT$ , and at time  $t + MTRT + INRING\_TIME$ , because a station must have accepted a token during  $[t + MTRT, t + MTRT + INRING\_TIME]$  or have formed a self-ring. Thus, by  $t + MTRT + INRING\_TIME$ ,

if a station remembers anything about the token with a particular ring address,  $B$ , they are remembering the same token. If a station receives a token from a station outside the ring, then the token must have made a loop from station  $y$  to  $w$  and back to  $y$ . This means that there was a breach in the ring that  $x$  and  $y$  belongs to, because when a token travels it makes bijection between the sender and the acceptor of the token. For station  $x$  and  $y$  to be in the same ring at time  $t + MTRT + INRING\_TIME$ , a new token must have been regenerated by a station in the loop after the token with ring address  $B$  has passed them by. But this does not make sense because  $y$  must have received the token with the ring address  $B$  within  $MTRT$  since the last time it saw it, and all stations in the loop have seen the token after station  $y$  accepted it. ■

*Lemma 9:*  $|S|$  monotonically increases and will converge to  $|V|$  in a finite time.

From lemma ?? and lemma ??,  $|S|$  is not decreasing. According to lemma ?? that a ring will not break nor decrease in size after time  $t + MTRT + INRING\_TIME$ . In addition a station will not solicit another station to join unless it sees two successful token rotations. If there are no topological change, a station will not join a node unless it is part of a ring. This means that the number of stations in a ring does not decrease.

If  $|S| \neq |V|$  at some time  $s$  such that  $s > t + 2MTRT + INRING\_TIME$ , then there exists station  $y$  that will not receive a NORMAL token from its predecessor at time  $t$ . If station  $y$  does not accept a token within the  $INRING\_TIME$  since the last time it accepted a token, it will form a self-ring. In this case, we gained a station that belongs to a ring. If the station  $y$  accepts a token, allowing another station to form the bijection with  $y$ , then we gain in the size of  $|S|$ . This is because the former predecessor did not have the bijection with  $y$ , and the new predecessor did not have the bijection with its former successor since it could not have generated SET\_PRED token to form the bijection with  $y$ . Since within every  $INRING\_TIME$ , the number of bijection that belongs to a ring, or the number of bijections increase,  $|S|$  will converge to  $|V|$  in finite time.

When  $|S|$  finally converges to  $|V|$  at time  $u > s$ , using the multiple token resolution lemma ??, we know that there exists one and only one token in all rings within  $u + IDLE\_TIME + INRING\_TIME$ . ■

*Lemma 10:* If  $|S| = |V|$  at time  $s > t$ , then within  $s + IDLE\_TIME + 3MTRT$ , there exists one and only one token in any ring.

There could be multiple tokens in a ring at time  $s$ . Either one and only one of these tokens will survive or none will survive by time  $s + 2MTRT$ . Station  $y$  in a ring will only accept a token if it has higher priority than the last token

User Interface	Simulation FrontEnd	Operating System IP
WTRP Core	WTRP Core	WTRP Core
Select Based Scheduler	Simulator Scheduler	OS Scheduler
Operating System UDP	Channel Model	IEEE802.11 DCF

**Figure 4.** Implementations

that it accepted. This means that after one revolution, the priority of all existing tokens must be increasing in terms of its order of visits to station  $y$ . All of these tokens must visit station  $y$  within another MTRT, and will be deleted but one token.

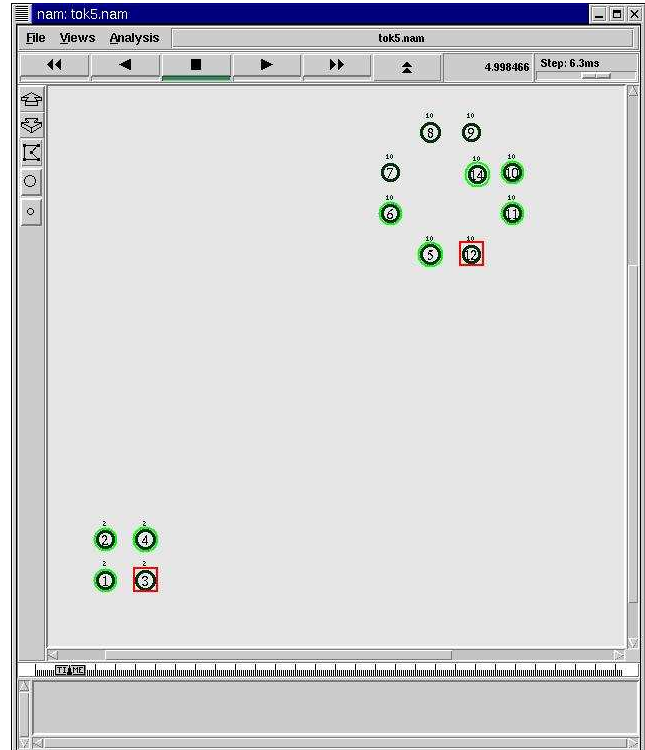
Even if all tokens get deleted at time  $u > s$ , within  $u + \text{IDLE\_TIME}$ , there exists at least a one token in the ring. From the bijection, we know that if  $x$  has the bijection with  $y$ ,  $y$  must accept tokens from  $x$ . This means that the station that holds the token has the higher priority than its successor. The only station that is an exception to this rule is the owner of the token, because the owner increments the generation sequence number by one when it passes the token. The only generation sequence number assignment that will satisfy these constraints is the following. The generation sequence number of the stations from the successor of the owner to the station to the station with the token has the same generation sequence number as the token. The generation sequence number from the successor of the station with the token to the owner is one less than that of the token. This means that when one or more stations regenerate token during  $[u, u + \text{IDLE\_TIME}]$ , the generation sequence number of these tokens will be higher than that of any stations at time  $s$ . Because these tokens will be passed around as a NORMAL token, only the highest priority token will survive within one MTRT, and lower priority tokens will be deleted. ■

#### 4. IMPLEMENTATIONS

We implemented the WTRP on top of 2 Mbps WaveLAN[?] PCMCIA card that implements the IEEE802.11[?] protocol. WTRP uses DCF (Distributed Coordinated Function) mode of IEEE802.11. This is done by prepending the token ring header to the IP packet and broadcasting all packets in peer-to-peer mode.

We have three different implementations of WTRP as shown in Figure ???. All three implementations use the same protocol core. This sharing of codes is made possible by the implementation of Linux kernel libraries for packet manipulations and event scheduling in user space.

The importance of the UDP implementation (Figure ??) is that it is platform independent. In a managed network, applications can reduce the frequency of packet collisions by sending the UDP packet using the UDP interface. In this



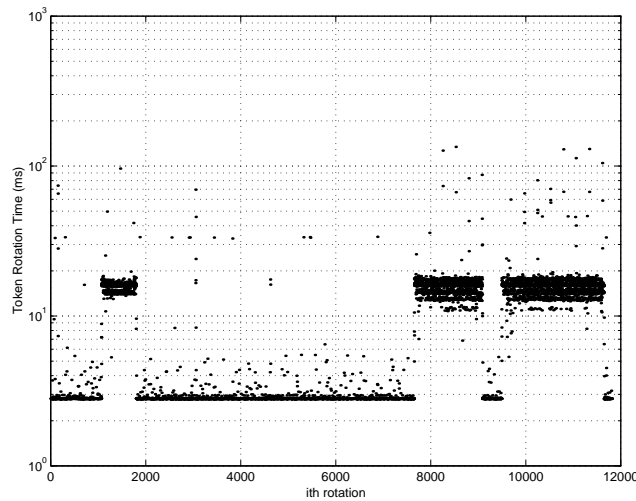
**Figure 5.** Visualization of the Simulator

case, WTRP is used as a transmission scheduling method rather than a medium access control. We have shown the efficiency of the implementation by streaming without jitter 1 Mbps video without buffering on the WaveLAN cards.

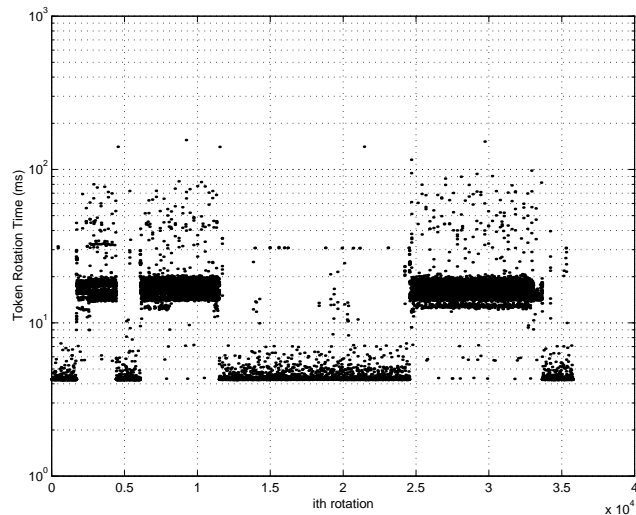
The simulator (Figure ??) allows testing and simulation of the same WTRP core shared by all implementations. This design of simulator is advantageous since it nullifies the need to develop separately for simulation and for deployment. The separated development is not only expensive in terms of added developmental costs, but also increases the chance of failure to identify potential problems of the protocol since the protocol that is tested in simulations and deployed is often different.

In terms of mobility model of the simulator, a node can be assigned one of implemented random movement patterns or follow a predefined path. The mobility is visualized in the Network Animator[?] by generating a log that Network Animator understands. A screen shot of the visualization is shown in Figure ???. In terms of the channel model, the simulation models a simple Direct Sequence Spread Spectrum (DSSS) capturing effect based on transmission to interference ratio, free space wave propagation, and underlying CSMA radio. Multipath or fading affect is not modeled.

The Linux kernel module is developed for kernel version 2.2.19, and it is inserted between the IP and WaveLAN libraries. The next section (Section ??) shows the performance analysis conducted on the Linux kernel module.



**Figure 6.** Token Rotation Time during FTP (Two Nodes, 2048Hz)

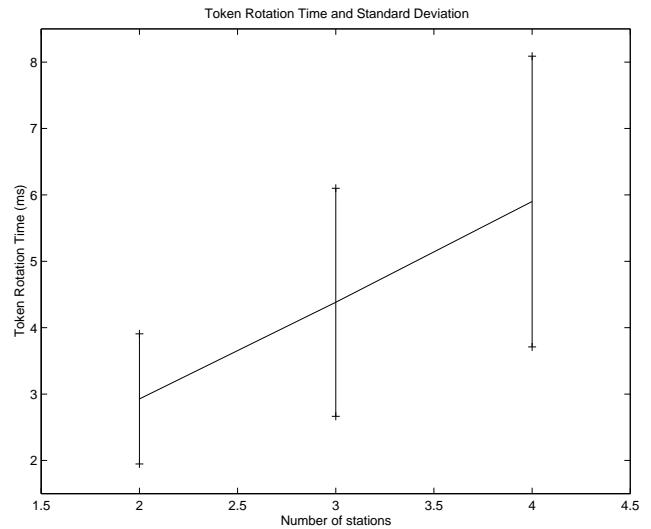


**Figure 7.** Token Rotation Time during FTP (Three Nodes, 2048Hz)

5. PERFORMANCE

In the trial shown in Figure ??, a large file size 5830080 Bytes was transferred using FTP between two nodes running the token ring medium access protocol. Figure ?? shows the trial done with three nodes. A file of size 7403520 bytes was transferred. The LINUX clock resolution was set at 2048Hz. The token rotation time was measured at the FTP server for each token rotation. The file transfer of the same file was done three times with a single transfer, 2 concurrent transfers, and 3 concurrent transfers. The three peaks in Figure ?? and Figure ?? correspond to the three FTP transmission periods.

From the figure, one can see that the token rotation time did not increase in spite of an increased number of simultaneous transmissions. Rather, the transfer took longer to complete in direct proportion to the number of concurrent FTP



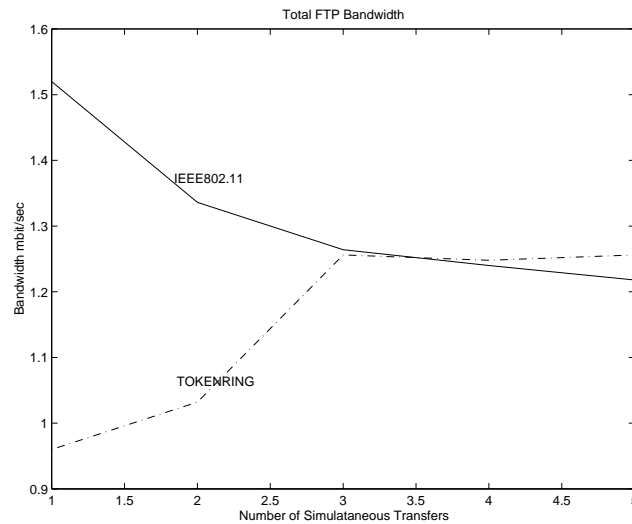
**Figure 8.** Mean Token Rotation Time

transfers. This bound on latency is the most salient property of the token ring protocol. It enables real time application support.

The test results seem to indicate that WTRP is scalable with the number of nodes. The following trials are done to see this. The scalability of the system response can be measured by observing the increase in token rotation time as the number of the nodes in the ring is increased. As expected, the mean token rotation time increases linearly with the number of nodes in the ring as shown in Figure ?. The standard deviation of token rotation time is also indicated in Figure ?. At least with small number of nodes, the standard deviation increased linearly. In addition, token rotation time is well contained. Less than 0.01% of the time the rotation time took longer than 40ms during the trial (0.0017% for Figure ?? and 0.0064856% for Figure ??).

6. COMPARISON WITH THE IEEE802.11 IN DCF MODE

The token ring protocol in its current implementation is disadvantaged relative to the original IEEE802.11 driver because the token ring protocol is implemented on top of IEEE802.11 in DCF mode, incurring all the overhead that is associated with IEEE802.11 plus the overhead from the token ring. The overhead is the increased computation time and packet header size. In spite of these disadvantages, we find that under heavy load, the token ring implementation performs better than the IEEE802.11 in DCF mode. This is shown in Figure ?. In the figure, the aggregate FTP bandwidth is plotted against the number of simultaneous FTP transfers. Both cases involved five nodes. Regardless of the number of simultaneous transmissions, the ring was formed with five nodes. The FTP was done as follows. For the case of two simultaneous transfers, one transfers went from station 1 to station 2 another from station 2 to station 3. For the case of three simultaneous transfers, 1 to 2, 2 to 3, and



**Figure 9.** FTP performance (IEEE802.11 in DCF mode vs. Token ring)

3 to 1.

In Figure ??, the solid line represents the IEEE802.11 in DCF mode and the dotted line represents the token ring protocol. At least with the number of nodes that were involved, we observed a decrease. The decrease in the throughput is expected since the number of collisions increase in a CSMA medium access control.

The performance surprisingly improves in the Wireless Token ring case when going from 1 to 3 simultaneous transfers. This can be explained as follows. Since for all trials in Figure ??, the ring size remained constant regardless of the number of simultaneous transfers, the number of token transmissions per token rotation remained constant in all trials. However, on increasing the number of simultaneous transfers, the number of data transmissions per token rotation is increased. This increases the ratio of data to token transmissions. This decreases the overhead per data bit. The need for retransmission of the data due to collisions is eliminated since there is no collision in the token ring implementation. We found that the token ring actually performs better when there are more than three simultaneous transfers.

The Centralized approach (802.11 PCF mode) uses the star topology, meaning that all slave nodes need to have a connection with the master. It is then easier to manage the network since all management information can be stored at the master. However, the approach is vulnerable to a single node failure at the master.

## 7. CONCLUSION

Even though the token ring protocol was implemented on top of the IEEE802.11 in DCF mode, we found that the to-

ken ring performs well or even better under heavy load. We have designed a protocol that is fast in terms of recovery (since there were several tokens lost during the tests) and efficient in terms of header size. One reason for the fast recovery is the use of a connectivity cache in each station. The performance results also show that we have a software implementation that is useful under a controlled application environment when utilized on top of an arbitrary network interface card.

The consistency of the token rotation time, regardless of the number of simultaneous transmissions is key to bounding the medium access latency. This perhaps is the most valuable feature of the wireless token ring protocol, since this is necessary in real time applications. In fact, the development of IEEE802.4 was initiated by people from General Motors and other companies interested in factory automation[?]. It has been established as a “mandatory protocol within the General Motor’s manufacturing automation protocol (MAP)”[?]. Features such as bounded latency and robustness against multiple node failures are some of the reasons for this choice. Our design bring the same bounded latency and robustness features to the wireless medium. Moreover, WTRP accomodates ad-hoc topologies.

## REFERENCES

- [1] *Draft International Standard ISO IEC 8802-11* — IEEE P802.11/D10, 14 January 1999
- [2] *High Performance Radio Local Area Network (HIPER-LAN), Type 1; Functional specification*, ETS 300 652, Radio Equipment and systems (RES) October 1996
- [3] <http://www.bluetooth.com>
- [4] Akyildiz, I.F.; McNair, J.; Martorell, L.C.; Puigjaner, R.; Yesha, Y., *Medium access control protocols for multimedia traffic in wireless networks*, IEEE Network, vol.13, (no.4), IEEE, July-Aug. 1999. p.39-47.
- [5] Hannikainen, M.; Knuutila, J.; Letonsaari, A.; Hamalainen, T.; Jokela, J.; Ala-Laurila, J.; Saarinen, J., *TUTMAC: a medium access control protocol for a new multimedia wireless local area network.*, Ninth IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, New York, NY, USA: IEEE, 1998. p.592-6 vol.2. 3 vol. 1574 pp.
- [6] P.Varaiya., *Smart Cars on Smart Roads: Problems of Control.*, IEEE Transactions on Automatic Control, 38(2):195-207, February 1993.
- [7] H. Shim, T. J. Koo, F. Hoffmann, S. Sastry, *A Comprehensive Study on Control Design of Autonomous Helicopter*, In Proceedings of IEEE Conference on Decision and Control, Florida, December 1998
- [8] *International Standard ISO IEC8802-4:1990* — ANSI/IEEE Std. 802.4-1990
- [9] Duke Lee, *Wireless Token Ring Protocol Master’s Thesis* at UC Berkeley, 2001

- [10] <http://www.wavelan.com>
- [11] <http://www.isi.edu/nsnam>
- [12] A. Tanenbaum, *Computer Networks* Prentice Hall, New Jersey, 1988
- [13] W. McCoy, *RFC1008: Implementation Guide for the ISO Transport Protocol* 1987

**Duke Lee is a graduate student at UC Berkeley, has been involved in UC Berkeley PATH Project since 1997.**

**Roberto Attias received his BS in Computer Science at the university of Milan, Italy in 1995. He held a research position at PATH in 1999 and is currently with Teja Technologies, where he works on the development of software platforms for network processors.**

**Anuj Puri received his PhD from the University of California, Berkeley in 1995. He was with Bell Labs of Lucent Technologies after that. Since January, 1999 he has been with Department of EECS at the University of California, Berkeley. His research interests are in wireless networks and embedded systems.**

**Stavros Tripakis received his PhD in 1998 from the Verimag laboratory in Grenoble, France. He is currently working at Verimag as a CNRS researcher. His interests include formal methods, networks and embedded systems.**

**Pravin Varaiya is Nortel Networks Distinguished Professor at UC Berkeley. He is a Fellow of the IEEE and a member of the National Academy of Engineering.**