Token Relay with Optimistic Joining

Eric E. Johnson*, Zibin Tang*, and Manikanden Balakrishnan*
Klipsch School of Electrical and Computer Engineering
New Mexico State University

ABSTRACT

Token passing has emerged as an attractive channel access control mechanism for wireless networks with long link turnaround times, such as the HF Wide Area Network (HFWAN) network used in naval battle groups. However, the use of token passing over unreliable links requires sometimes complex recovery protocols, which can reduce network performance both during network formation and in steady-state operation. In this paper we examine the ability of token relaying to add robustness to ring formation as well as during link outages of varying durations.

1. INTRODUCTION

Token passing has been found to be surprisingly well suited to channel access control in some wireless networks applications [1, 2]. One potential drawback to token passing is the need to maintain a connected ring of stations for passing the token. Topology changes can invoke recovery mechanisms whose overhead traffic might reduce the expected throughput advantage of the contention-free token protocol.

Approaches to handling topology changes have evolved over the short history of wireless token protocol development:

- The Wireless Token Ring Protocol (WTRP) [1] uses timeouts to detect breaks in the ring, and recovers by dropping a node from the ring if the ring can be patched by excluding that node, or by dissolving and rebuilding the ring.
- The HF Token Protocol (HFTP) [2] also uses timeouts to detect lost links, but will attempt to route around a link outage by relaying the token. This avoids dropping a node if it is still reachable by any node in the ring. When relaying fails, HFTP falls back to the WTRP recovery approach.

In some pathological topologies, however, a ring cannot form using the usual "solicit successor" mechanism (described in section 3). In this paper, we document this problem, and introduce a solution that increases the robustness and efficiency of token passing for wireless network channel access control.

2. TOKEN RELAY IN HFTP

Token passing is a contention-free channel access (a.k.a. Medium Access Control or MAC) protocol in which permission to transmit is circulated among network nodes via a conceptual "token." In normal operation (Figure 1), the nodes in a wireless network employing a token passing MAC protocol are organized into a ring. Each node has a successor to which it passes the token, and a predecessor from which it receives the token.

The ring topology is extracted from the (more densely connected) graph of nodes and their interconnections during ring formation. Depending on the density of interconnections, nodes may be able to overhear many other nodes in the network, and each node can build a list of many of the stations in the ring, in successor order, from overheard traffic. A counter in the token informs each node of the size of the ring, so each node can insert the appropriate number of blank positions in this Connectivity Table.

After a node holding the token has finished its transmissions for the current token tenure, it sends ("passes") the token to its successor. Success in passing the token may be detected from an explicit acknowledgement from the successor or implicitly by overhearing transmission(s) from that successor. If a node fails in its first attempt to pass the token to the successor node, it will make several additional attempts to pass the token, but if these also fail, a link outage will be declared.

In HFTP, recovery from link outage begins with a Solicit Relay packet, sent by the node trying to pass the token (node A in Figure 2). The Solicit Relay packet specifies the successor node (B) that the soliciting node needs to reach. Responses to the Solicit Relay packet are sent in slots corresponding to the token passing order; this maintains the contention-free nature of the token-passing MAC protocol.

* This work supported by the US Navy Space and Naval Warfare Systems Center under contract N66001-03-D-0042

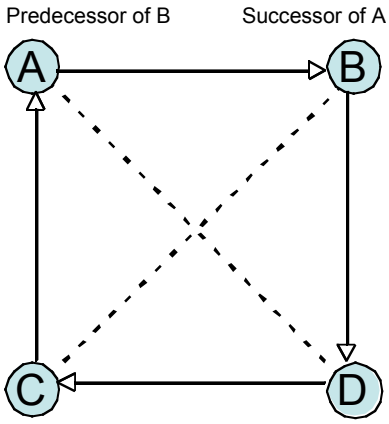

Figure 1. Token Passing

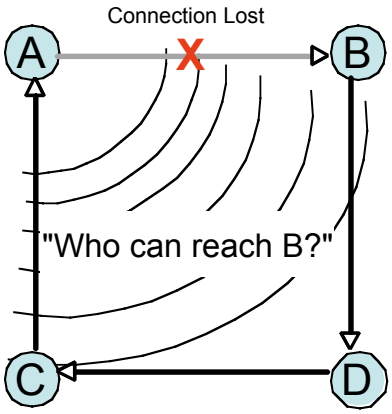

Figure 2. Soliciting a Relay

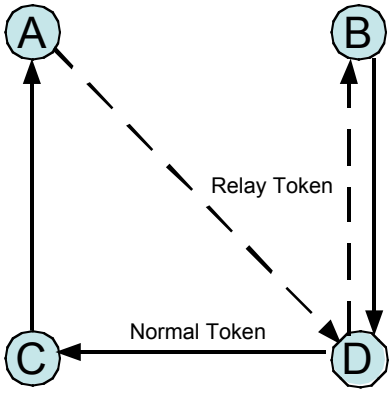

Figure 3. Token Relay

If there is at least one positive response, the soliciting node (node A in Figure 3) will select a relay node for the token, and send a special Relay Token to that node (D). The Relay Token specifies the successor node to which the token is to be sent (B).

Processing by the relay node is simple: it forwards the token to the desired destination and listens for an acknowledgement. It will retry as usual if there is no acknowledgement, but if the retries fail, it will keep the token and inform the soliciting node that the lost node is truly lost; the relay node then becomes the new successor to the soliciting node. Normally, however, the Relay Token will reach its destination, and that node will convert the Relay Token into a normal token and use it. Token passing then continues as before, except that the soliciting node (A) will relay the token at each rotation until it again overhears its successor (B). This recovery mechanism quickly returns the ring to full operation, but network overhead is increased by one token transmission time per token rotation until the link is restored.

3. RING FORMATION IN HFTP

Ring formation (and adding nodes to an existing ring) in WTRP and HFTP employs a similar solicitation mechanism. When token-passing nodes power-up, they enter a "Self-Ring" state: a ring of one. If such a node overhears an operating token ring, it changes state to Floating and waits to be invited to join that ring. However, if it doesn't detect another ring, it periodically broadcasts a Solicit Successor packet, inviting any nodes within range to join its ring.

Responses to the Solicit Successor packet are sent in randomly selected time slots to minimize collisions. Slots must be selected randomly because, unlike the Solicit Relay responses, the responders are not yet organized into a network with a successor ordering that could be used to assign slots in collision-free fashion.

The Solicit Successor packet names the current successor of the soliciting node. Responders must be able to insert themselves cleanly into the ring, which requires that the joining node be able to reach both its new predecessor (the soliciting node) and its new successor (the successor of the soliciting node). A node in the Self Ring state names itself as its successor (Figure 4).

In Figure 5, A and B have formed a ring of two nodes, while C and D have entered the Floating state. After some number of token rotations, B solicits a successor, naming A as its current successor. Both C and D are eligible to respond because both can reach both B and A.

In Figure 6, we see that D was chosen as the successor to B. Node A is again soliciting, and C will join the ring as successor to A. The token flow in this case will follow a figure-eight pattern: A-C-B-D.

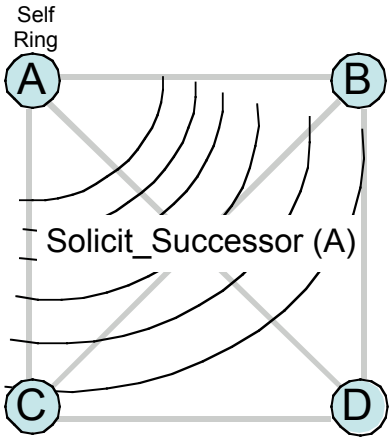

Figure 4. Solicit First Successor

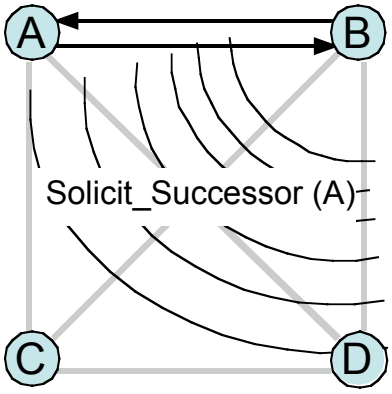

Figure 5. Adding Third Node

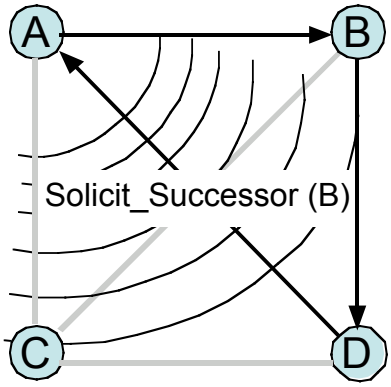

Figure 6. Adding Fourth Node

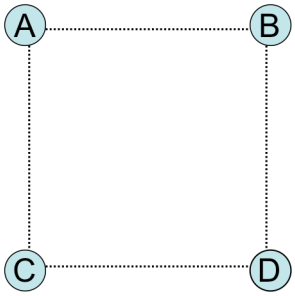

Figure 7a. Square Topology

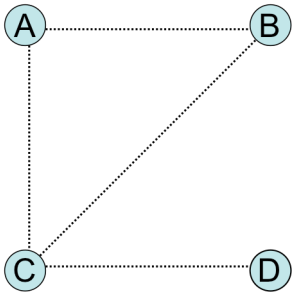

Figure 7b. Whisker Topology

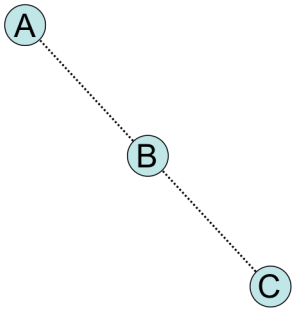

Figure 7c. Line Topology

4. PATHOLOGICAL TOPOLOGIES

Some sparsely connected network topologies do not support ring formation using the "solicit successor" mechanism just described. Three such "pathological" topologies are shown in Figure 7. In each case, communication is possible only along the links shown in dashed lines.

The Square Topology will support a ring (A-B-D-C), but the "solicit successor" mechanism cannot form it. For example, assume that A solicits first, and that B responds and is selected as the successor to A. If B then solicits, D is the only potential successor, but D cannot reach A and therefore cannot respond. Likewise, when A next solicits a successor, C cannot respond because C cannot reach A's current successor, B. From the symmetry of the topology, this argument holds for any pair of stations that first form a two-node ring.

In the Whisker Topology, the A-B-C ring will be formed by the "solicit successor" mechanism, but D cannot join the ring because D can only reach one station, C.

The Line Topology (the classic "hidden node" topology) cannot even support a ring, and so presents an especially difficult case for a ring-based protocol.

Other pathological topologies arise from combinations or generalizations of these three. For example, larger rings with no cross-connections are similar to the Square Topology. Rings with arbitrarily long "whiskers" are possible, as are lines of arbitrary length as well as stars and trees (which, like the Line Topology, have no connected cycles that would support a ring).

5. TOKEN RELAY WITH OPTIMISTIC JOINING

Supporting token passing in arbitrary sparsely connected topologies requires that we recognize that the ring around which the token is passed need not correspond with the physical topology of the network. That is, it is not necessary that the successor node in the token ring be directly connected to its predecessor. This notion is already recog-

nized in the Token Relay mechanism of HFTP, so it is only necessary to generalize this to the ring formation mechanism.

The current HFTP specification requires that a node responding to a Solicit Successor packet must be able to reach the current successor named in that packet. We propose allowing a node that wishes to join the ring to lie about its connectivity: a node may respond "optimistically" even if connectivity to the current successor is unknown. If the optimistic node is accepted as successor and it discovers that it cannot, in fact, reach its new successor, it must immediately request relaying of its tokens, probably via its new predecessor.

5.1 Protocol Extensions

The enhancements of HFTP to support ring formation in the challenging network topologies described above fall in two areas that we term "Optimistic Joining" and an extended version of Token Relay. The details of these enhancements are specified below.

5.1.1 Optimistic Joining

1. A node in the Floating (or Self Ring) state always responds to Solicit Successor; it doesn't check connectivity to that successor.
2. If it receives the token in response, it checks its Connectivity Table:
   a. If the new successor is directly reachable, it sends the token directly. If it fails to pass the token, it begins the Solicit Relay process.
   b. Otherwise, it begins the Solicit Relay process immediately.
3. If Solicit Relay fails while joining, it sends a RelayToken to its new predecessor.
4. If sending a RelayToken to its predecessor fails (should not happen; may mean the node's own radio is down), it discards the token and goes to the Idle state. The network will resume operation upon expiry of a lost token timeout.

5.1.2 Relay Tokens

All tokens, including the RelayToken, are MAC-layer packets. The modified RelayToken contains address fields for the usual MAC addresses, Immediate Sender and Immediate Destination, plus two new fields: Ultimate Destination and Original Sender. The Ultimate Destination is the successor to the Original Sender. These new fields support relaying the token through multiple intermediate nodes.

5.1.3 Processing of Relay Token

A node that receives a RelayToken processes it as follows:

1. If it is the Ultimate Destination, it converts the RelayToken to a real token and uses it. The Original Sender of the RelayToken is recorded as its new predecessor;
2. else if the node has already seen this RelayToken (a loop has developed), the RelayToken is discarded and the node goes to Idle; the network will resume operation upon expiry of a lost token timeout;
3. else if the destination is directly reachable, the node sends the RelayToken to its destination (if no passive ack is received, fall through to the next step);
4. else forward the RelayToken to the node's predecessor.
5. If sending a RelayToken to the predecessor fails, the RelayToken is converted to a normal token and used. Then a normal Token is forwarded to the successor.

Flowcharts for processing Normal and Relay Tokens are shown in the Appendix.

5.2 Examples

In this section, we illustrate how Token Relay with Optimistic Joining addresses the pathological topologies described earlier. In Figure 8, we see how a ring is formed in the square topology. In Figure 8a, a two-node ring is formed as usual. In Figure 8b, node D has responded to a Solicit Successor from B and receives the token from B. Since D cannot reach its new successor, A, it relays the token via its new predecessor, B. In Figure 8c, node C has joined the ring as the successor to D, and it can directly reach its new successor A, so no Relay Tokens are now necessary. This is because, as noted earlier, the topology can support a ring in steady state; it just couldn't support formation of a ring without the new feature.

Figure 9 illustrates the formation of a virtual ring in a linear topology. As usual, a two-node ring forms first (Figure 9a). To add a node on either end of the line requires Optimistic Joining (a.k.a. lying). In Figure 9b, node C joins as successor to B, and relays the token the length of the line to its new successor. This procedure is applicable to linear topologies of arbitrary length.

Finally, in Figure 10 we see a ring operating with a "whisker" node D, which can only reach node C. The token circulates normally from A to B to C to D. Then D relays the token via C back to A.

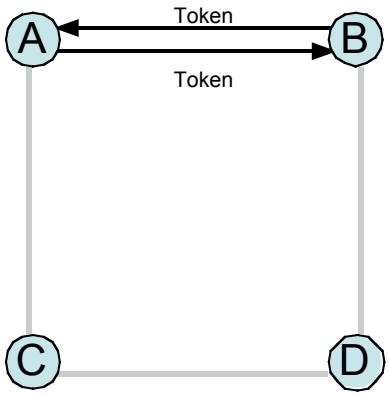

Figure 8a. Square: 2 Nodes

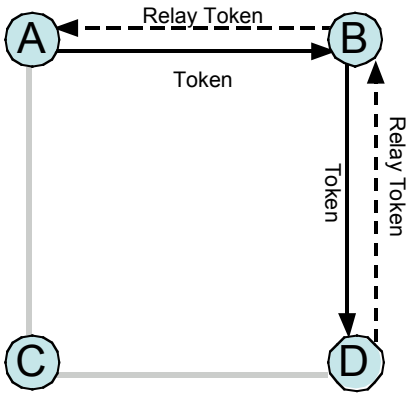

Figure 8b. Square: 3 Nodes

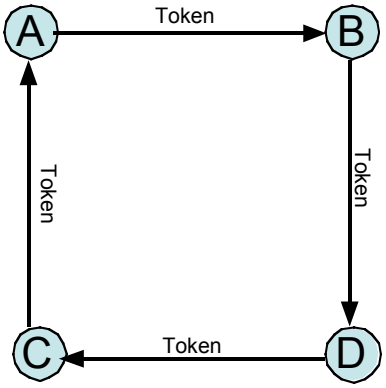

Figure 8c. Square: 4 Nodes

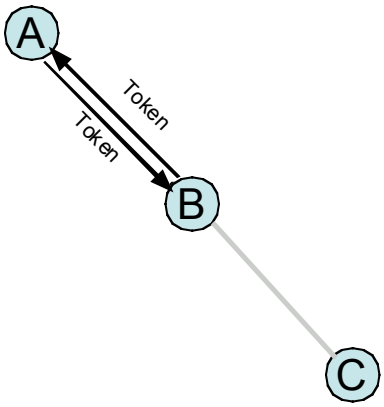

Figure 9a. Line: 2 Nodes

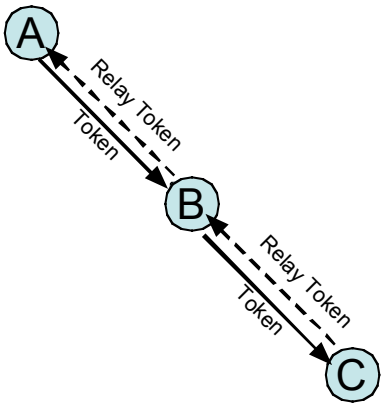

Figure 9b. Line: 3 Nodes

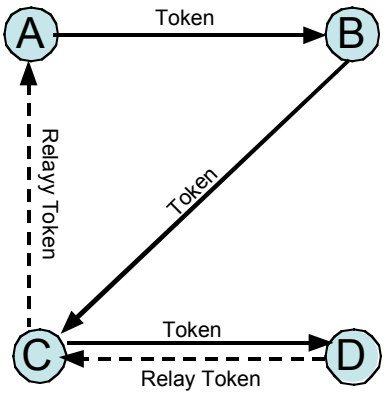

Figure 10. Whisker

5.3 Analysis

An important concern is whether this algorithm always results in a normal token circulating in the network within a finite number of steps. This can be proved as follows:

1. The fields used to detect a looping RelayToken — namely the Ultimate Destination, Original Sender, and Generation Sequence Number — are unchanged as the RelayToken is propagated.
2. With a finite number of nodes, a RelayToken cannot circulate indefinitely within the network. Within a finite number of steps (see flowcharts in the Appendix), one of the following will occur:
    1. It will reach its Ultimate Destination.
    2. It will be converted to a normal token due to delivery failure.
    3. It will revisit a node and be discarded.
    4. It will be lost.

In the first two cases, the Relay Token is converted to a Normal Token within a finite number of steps. In the latter two cases, a new Normal Token will be created by some node when its Idle timeout expires.

Thus, a Relay Token will always become a normal token after a finite number of steps (possibly including a timeout).

5.4 Performance

Token Relay with Optimistic Joining has been implemented in the NetSim simulator at NMSU and initial performance experiments show that it works as expected. Furthermore, it forms rings in about the same time as HFTP in densely connected topologies.

Figure 11 below compares the time to form a ring in a fully connected 4-node network to our three pathological topologies. The ring formation time is shown in the following Table. The means of 6 simulation runs are shown, along with 95% confidence intervals for the mean.

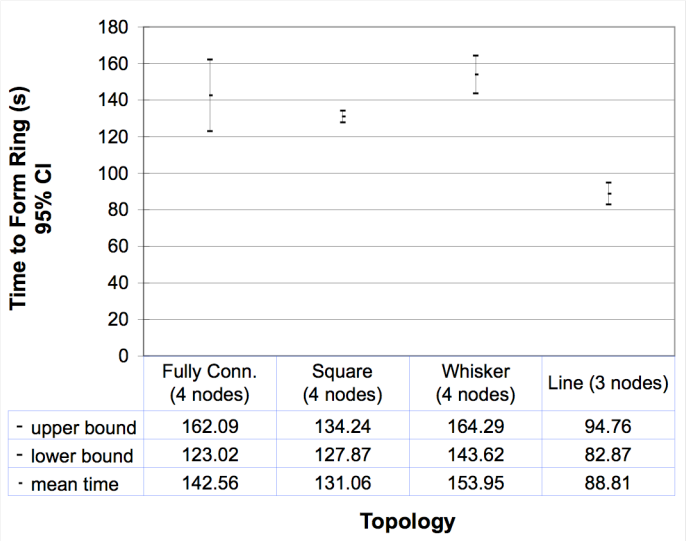

Figure 11. Time to Form Ring (Pathological Cases)

6. CONCLUSIONS AND FUTURE WORK

In this paper, we have proposed minor extensions to the HF Token Protocol that enable the formation of token rings in pathological topologies that do not support the formation of a token-passing ring in current wireless token-passing protocols (WTRP and HFTP). We are currently engaged in simulation studies of these extensions, termed Token Relay with Optimistic Joining, and anticipate future trials of the enhanced HF Token Protocol in the Allied HF WAN software stack.

7. REFERENCES

1. M. Ergen, *et al*., "Wireless Token Ring Protocol," *Proceedings of SCI 2002*.
2. E.E. Johnson, *et al*., "Robust Token Management for Unreliable Networks," *Proceedings of MILCOM 2003*.
3. STANAG 5066, *Profile for HF Data Communications*, Annex L–High-Frequency Wireless Token Ring Protocol (HFTP) Requirements, Edition 2, Draft 1, NATO C3 Agency, June 2004.

Appendix

The following flowcharts specify the processing of Normal and Relay Tokens when Token Relay is added to HFTP. C[X] refers to the entry in the local Connectivity Table for remote node with role X (e.g., successor or predecessor).

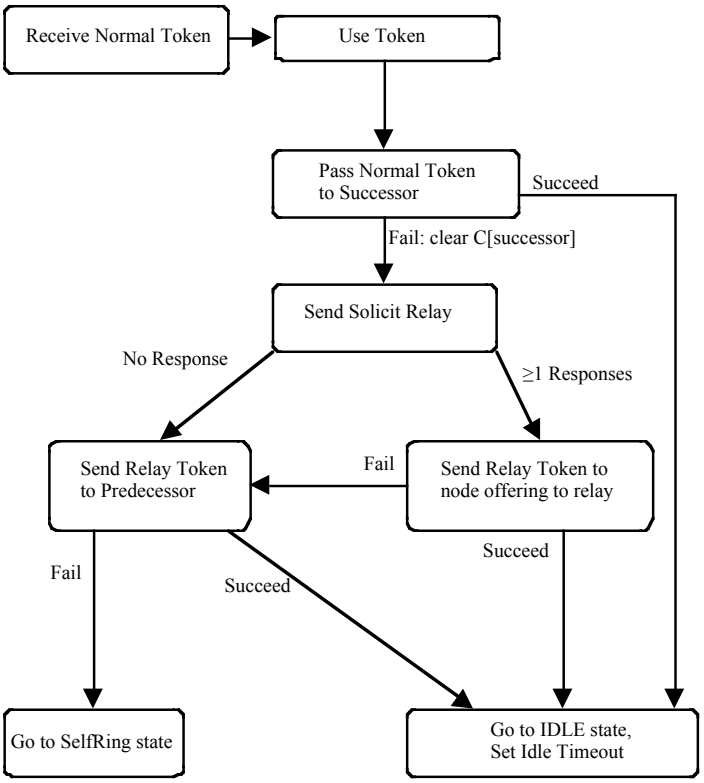

Figure A-1. Processing of a Normal Token

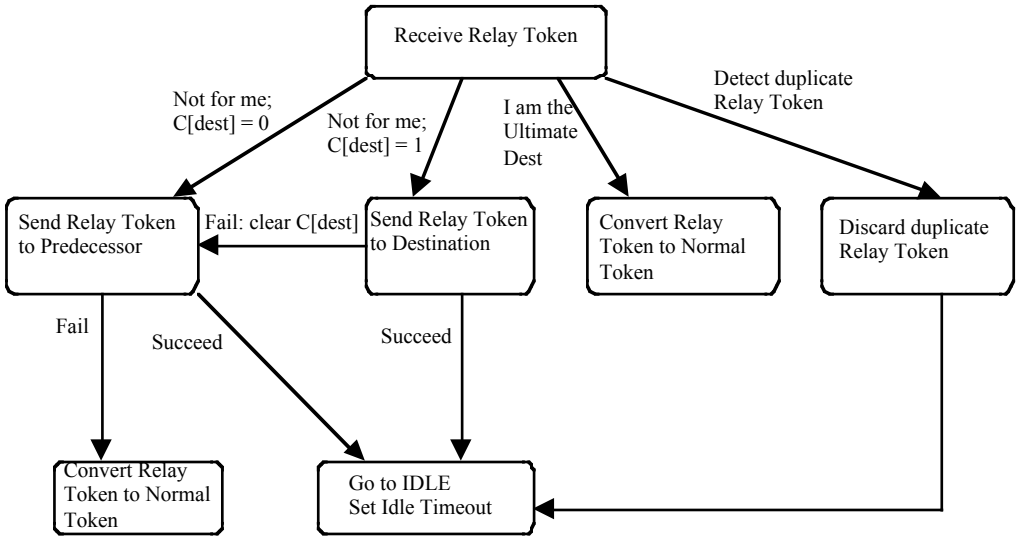

Figure A-2. Processing of a Relay Token