Chapter 8: Multicast Protocols

1. The Steiner Tree Problem

Reference 1, sections 3-3.2.1.

Steiner Tree Problem: Find the least cost multicast network to a subset of the nodes in a graph

The minimum spanning tree finds the least cost network to ALL of the nodes in the graph

- Broadcast rather than multicast.
- The best multicast tree is NOT necessarily a subgraph of the best broadcast tree.

The Steiner Tree

Exact solution:

Create a set of subgraphs

Each sub graph contains all of the nodes that must be in the Steiner tree

The set of subgraphs contain all possible combinations of the other nodes.

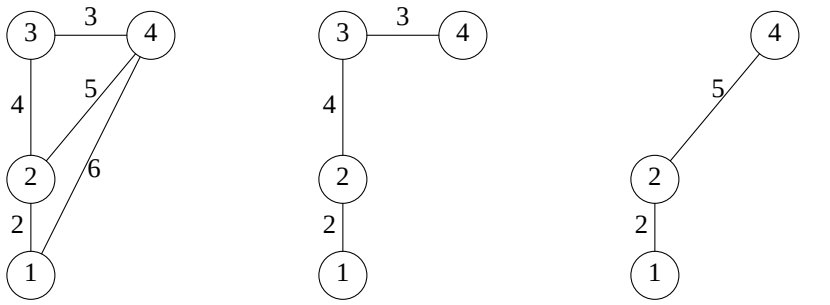
Find the minimum spanning tree of each subgraph.

The minimum weight tree of all of the sub graphs is the minimum weight Steiner tree.

Example:

Find the Steiner Tree for nodes 1, 2, and 4.

Graph MST with node 3 MST without node 3



There are 2 subgraphs one contains node 3 and the other does not.

The weight of the MST of the subgraph with node 3 is 2+4+3=9

The MST of the subgraph without node 3 is 2+5=7, and is the Steiner tree.

Note: If link 2-3 had weight 1, then the Steiner tree would have included node 3.

When the number of nodes in the network is large and there are a small number of nodes in the Steiner tree, the number of subgraphs can be large.

If there are N nodes in the graph and K nodes in the Steiner tree

there is 1 subgraph with exactly K nodes.

there are $\binom{N-K}{1} = N-K$ subgraphs with K+1 nodes

there are $\binom{N-K}{2} = \frac{(N-K)(N-K-1)}{2}$ subgraphs with K+2 nodes

. . .

and, there is $\binom{N-K}{N-K} = 1$ subgraph with N nodes.

The total number of subgraphs is $\sum_{i=0}^{N-K} \binom{N-K}{i} = 2^{N-K}$

In a network with 1000 nodes and 100 nodes in the multicast tree, there are $2^{900} \approx 10^{270}$ subgraphs.

Therefore, in large networks we must resort to heuristic techniques, rather than considering all sub-graphs.

One of many heuristics is the Takahashi-Matsuyama heuristic.

At each step, the receiver that is added to the tree has the shortest path between itself and the currently existing tree.

This heuristic combines Dijkstra's algorithm for calculating the the minimum spanning tree and Dijkstra's algorithm for calculating the minimum depth tree algorithms.

The nodes that are permanently connected to the tree are assigned depth 0 as in the minimum spanning tree.

Initially only the source node is connected.

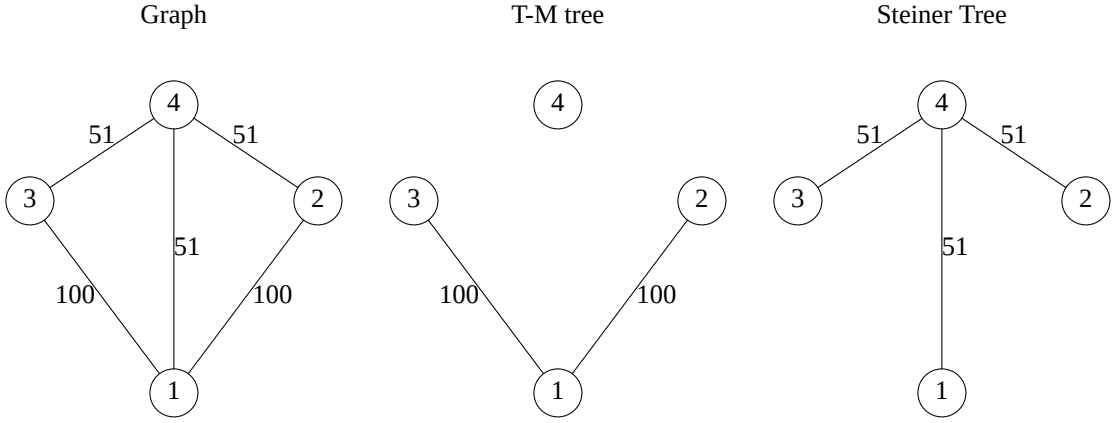
At each step the minimum depth algorithm is applied to the connected tree, and nodes are temporarily connected to the tree, until a receiver is temporarily connected.

This is the receiver that is at the minimum distance from the connected tree.

This receiver, and all of the nodes on the path to this receiver are permanently connected to the tree and assigned depths zero.

All of the remaining temporarily connected nodes are removed from the tree and the algorithm is repeated until all of the receivers are connected.

The heuristic does not always give the minimum Steiner tree. Steiner tree with nodes 1, 2, 3.



2. A Family of Reliable Multicast Protocols

1. **RBP/RMP:** The Reliable Broadcast/Multicast Protocol
   Application: A distributed/replicated database on an Ethernet
   Guaranteed message ordering at all of the receivers

2. **T-RMP:** The Timed Reliable Multicast Protocol
   Application: A fair distributed stock market
   Delay guarantees

3. **M-RMP:** The Mobile Reliable Multicast Protocol
   Application: Multicast on 802.11 LAN's with mobile users
   Receivers enter and leave multicast group frequently

4. **Neighborcast**
   By overlapping M-RBP groups, each vehicle participates in at most two groups, but communicates reliably with all of the vehicles within a specified distance of itself.
   Application: Nearby communications for each car in a very large string of cars on a highway

5. **Fail Safe Broadcast**
   Error on the side of safety. If there is any chance that any vehicle in a group has not received a message, none of the members use the message and all of the members follow the same exit strategy.
   Application: Safe communications in collaborative driving applications.

Common Characteristics

1. **Objective:** Reliable message delivery
   M receivers acquire all of the messages that are transmitted by N sources.

   • RBP - up to an earlier acked message

   • T-RMP - by a deadline

   • M-RMP - limited to a subset of persistent receivers

2. Additional Guarantees

   A. All receivers place all of the messages in the same order

      • This analogous to in order delivery in the ARQ protocols

      • The order of messages from different sources is the order in which they are acknowledged, which, because of lost receptions, may differ from the order that they are transmitted.

      • This characteristic is important for the consistent operation of a distributed database

   B. All Recievers know when all other receivers have a message

      • A receiver can make a decision based upon the source messages, and be confident that the other receivers can make the same decision.

      • Stock market - irrefutable trades

3. Efficient

   • RMP - 1 ack per source message
     Independent of number of receivers

   • T-RMP, M-RMP - 1 ack per period
     Independent of number of receivers or source messages

Operation:

1. All of the protocols use a token passing strategy to acknowledge messages.
   One site ack's source message then passes the right.

   • RMP - event driven by source messages

   • T-RMP - time driven

   • M-RMP - implicit token passing

2. We combine simple ARQ protocols to create more complicated protocols that have interesting new characteristics and are efficient, in terms of the number of control messages per broadcast message.

3. In these protocols we use two additional types of acknowledgements

   A. Negative acknowledgements, NAK's.

   — Instead of sending an ACK when a message is received, a NAK is transmitted when a message is not received.

   — There are fewer NAK's than ACK's because most messages are received correctly.

   — NAK's cannot be used in simple ARQ protocols, because a receiver does not know when it misses a message.

   — If source messages are numbered, receiving a higher number than expected can be used to trigger a NAK for a missing source message in the go-back-N and selective repeat protocol.
   This strategy only works when there is another message to send.

   B. Implicit acknowledgements.

   — ACK's that are not explicitly transmitted

   — The ACK is implied by receiving a later message

   — In the go-back-n protocol, acknowledging a higher message number, implies that the lower message numbers were received.
   When an acknowledgement is lost, a later acknowledgement indicates that it was transmitted.

4. In addition, in a system with scheduled messages, we can get information through reliably by not transmitting.
   We cannot receive a message that was not transmitted.

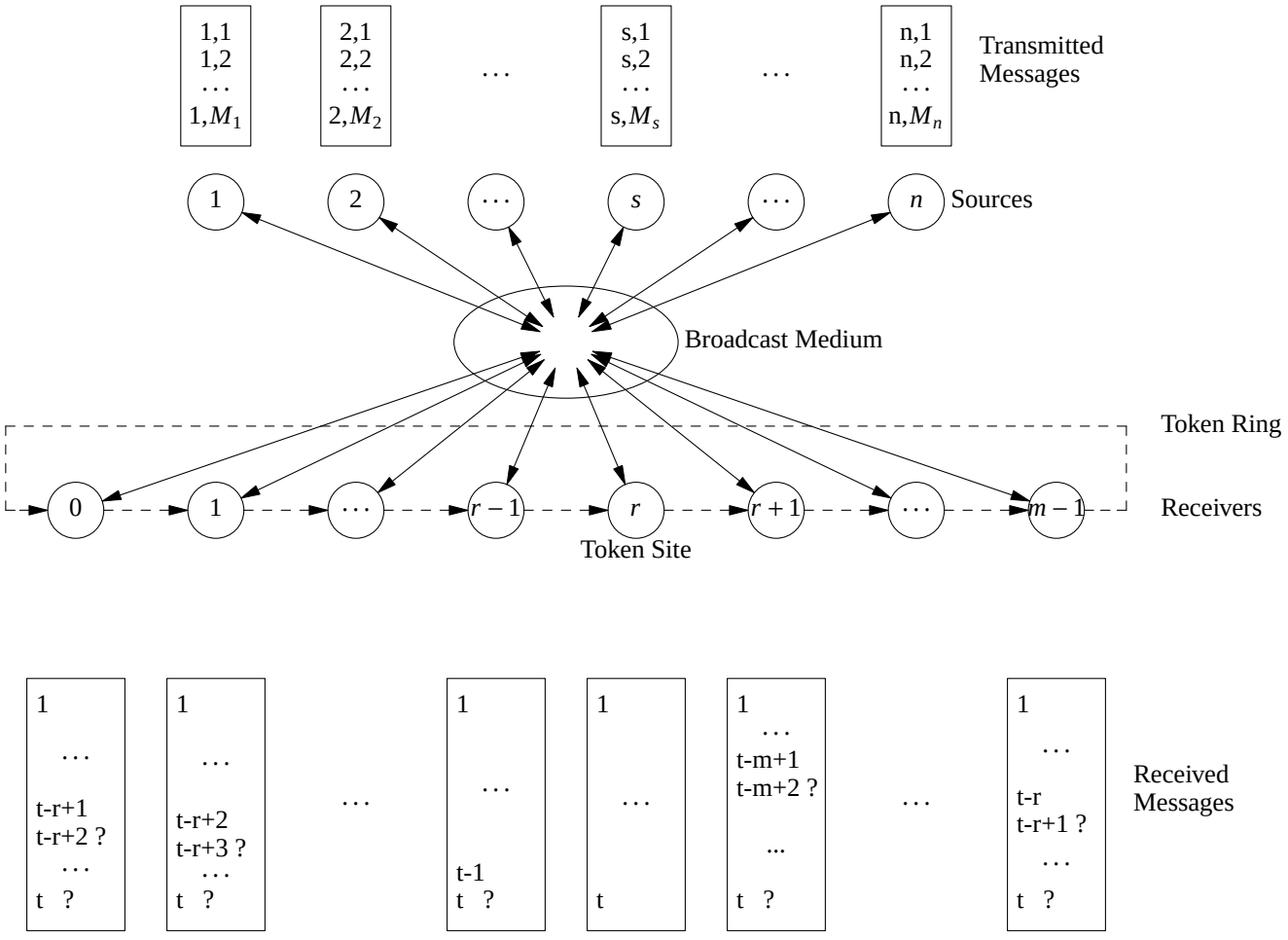
2.1 RBP - reliable broadcast protocol [2]

The RBP protocol is a pseudo-stationary protocol. It operates with a fixed group of receivers. If a receiver enters or leaves the group, the protcol stops, forms a new group, then operates with the new group.

It has two parts:

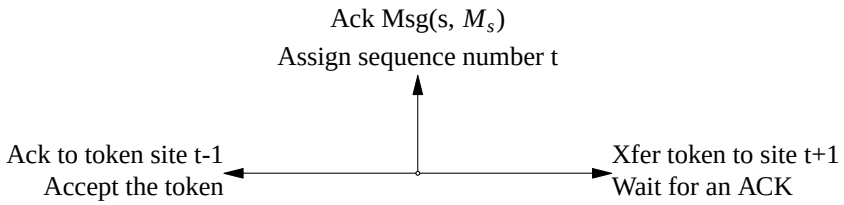
1. During normal operation it guarantees the delivery and ordering of the messages from the sources. There are a fixed group of receivers in the multicast group.
2. After receivers fail or recover from failures there is a reformation protocol that reorganizes the multicast group and guarantees the consistency of message sequences at the receivers in the new group.
    - The original reformation protocol is a 3-phase commit protocol:
        - — It is completely distributed
        - — It operates by electing a leader, then forming a list of operable receivers
        - — It guarantees a single group by requiring that at least half of the receivers from the previous group are in the new group.
    - In most practical implementations, the distributed protocol has not been implemented. Instead, a single control site, which may be backed up, forms the list
    - We will not discuss the distributed reformation protocol.

The RBP - Architecture



- All Source messages and ACK's are broadcast
- Each source numbers its messages
- One receiver - the token site
  Ack's the message and assigns a unique sequence number
- All other receivers use negative acknowledgements to recover missing source messages and ACK's
- The token rotates

The Token Site Message



Three Functions at token site r

1. **Ack message (s, M_s), assign it sequence number t**

The source uses a positive ack protocol
It retransmits (s, M_s) until it receives an ACK

If the source retransmits (s, M_s) after ACK(t) is transmitted
The source did not receive ACK(t)
ACK(t) is retransmitted

2. **Acknowledge receiving token t-1**

Token site r-1 uses a positive ack protocol to transfer the token
It retransmits ACK(t-1) until it receives ACK(t)
If token site r does not have a source message to acknowledge, it sends a confirmation message to token site r-1

If token site r retransmits token ACK(t-1) after ACK(t), or a confirmation is transmitted
Site r-1 did not receive ACK(t), or the confirmation
Retransmit ACK(t), or the confirmation

3. **Transfer the token to token site r+1**

Token site r uses a positive acknowledgement protocol to transfer the token to token site r+1
Token site r retransmits ACK(t) until it receives ACK(t+1), or a confirmation

TOKEN PASSING GUARANTEES

RULE:

Accept the token only after acquiring all acknowledged source messages

Token site services retransmission requests for source messages and acknowledgements until the next token site acknowledges receiving the token
One site retransmits messages - usually
The sources may leave the system

When token site r acknowledges message t

1. Site r has all messages up to t

2. Site $(r - k)$ mod m has all messages up to $t - k$

All sites have all messages up to $t - m + 1$

1. Site r knows that all sites have all messages up to $t - m + 1$

2. Site $(r - k)$ mod m knows that all sites have all messages up to $t - k - m + 1$

All sites know that all sites have all messages up to $t - 2(m - 1)$

2.1.1 E-FSM of a receiver

Figure 2 is an extended finite state machine, E-FSM, representation of the actions that a receiver takes when an acknowledgement is processed.

— The states indicate tests that are performed or situations where the receiver waits for an external stimuli, such as a message or a timeout.

— The transitions between states are labeled with the event that caused the transition, followed by a "*"'ed list of actions that occur during the transition.

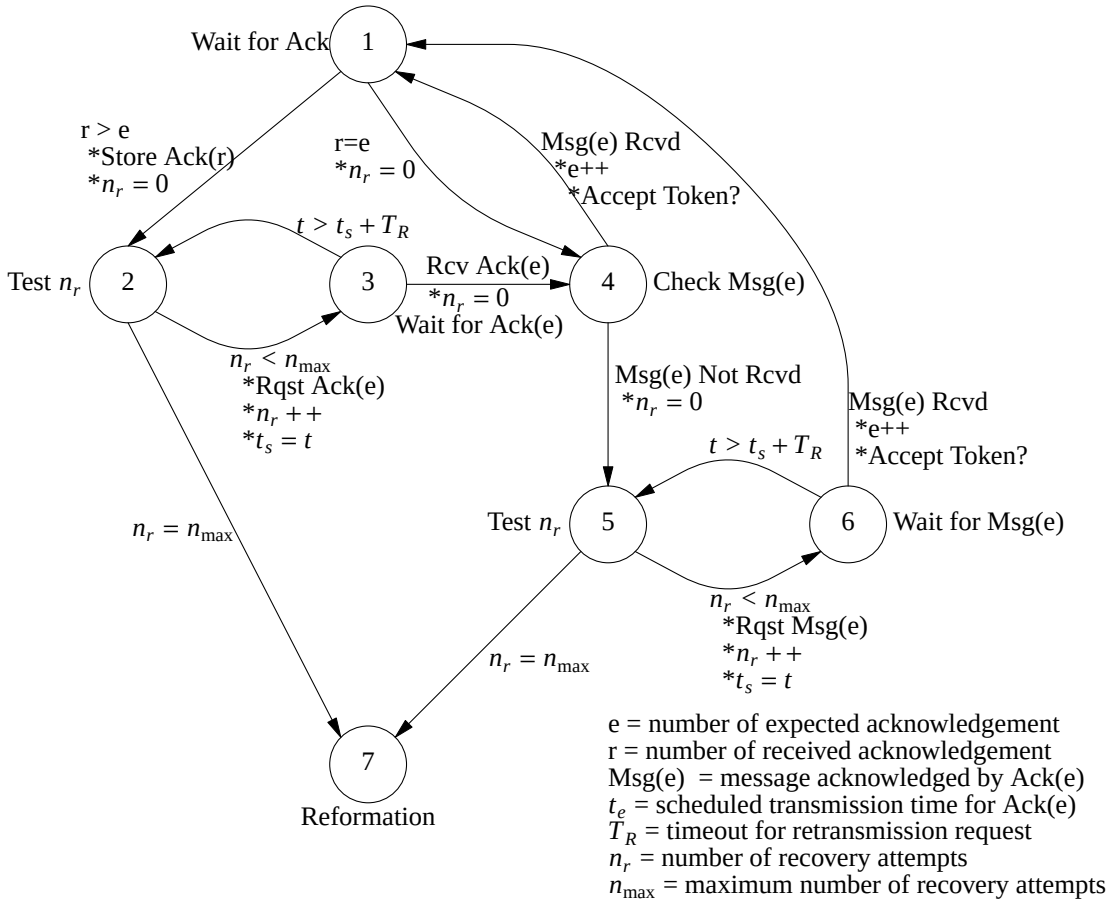


Figure 1. An E-FSM representation of acknowledgement processing at a receiver in the RBP protocol"

2.2 The Timed Reliable Multicast Protocol -- TRMP [3]

Basic Change: Event Driven → Time Driven Protocol
Periodically transfer the token each τ_t, instead of waiting for a source message

1. **Bulk acknowledgements:** Acknowledge all source messages which have not been acknowledged. Arrive during τ_t, or were not received by a previous token site

2. **If the acknowledgement is scheduled at** t_t

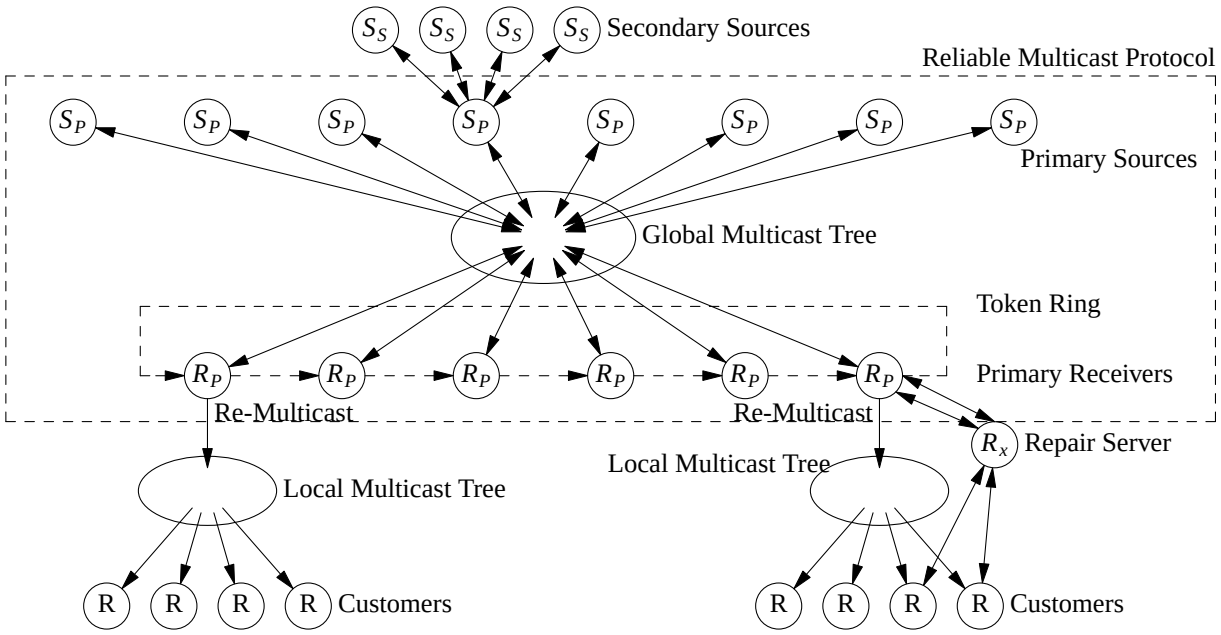
   — Each receiver starts recovery at $t_t + \Delta_N$ *(the nominal network delay)* and, Rexmits requests each 2 * Δ_n

— **Weak guarantee:** By $t_t + (4n_r + 1)\Delta_N$, every operable receiver has the ACK'ed source message, or the system is in reformation
- n_r is the number of retry attempts before declaring a failure
- Recover missing Ack, then recover missing source messages

— **Strong guarantee:** After a token cycle, every receiver is operable and has the message

3. When $\tau_t \geq (4n_r + 1)\Delta_N$ all receivers, including the next token site, are ready to receive when the ack is xmitted

Stock Market Architecture



- A small core of stable, trusted, network based components
- Provide Server to recover missed messages on the final leg
  2 grades of service
- **Common Ticker** - Simultaneous Delivery - Controlled by R_P
- **Submitting Orders**
  The token site is the portal into the list of acknowledged messages.
  Moving the token gives each source fair access into the list.
- **Distributed Trading Floor**
  Know when everyone has seen the trade.

2.3 The Mobile Reliable Broadcast Protocol, M-RBP [4, 5]

RBP and T-RMP are quasi-stationary; they operate on a fixed group of receivers, when the group changes, the protocol stops, forms a new group, and distributes a token list to the group members. M-RBP continues to operate when the underlying receiver group changes by using an aggressive token passing strategy that doesn't require receiving a token before using it. Inconsistencies that occur because of this strategy are resolved by a distributed voting procedure. The voting procedure also identifies and removes receivers that have left the group. New receivers enter the group by sending a message to the group and waiting until the vote to accept that message is complete.

M-RBP uses a token ring of receivers, as in RBP. The m receivers take turns as the acknowledging site by passing a token every Δ_T seconds. Each token has a unique number, t, and contains an acknowledgment for all received source messages that have not been previously acknowledged. When a source transmits a message it gives it a unique identifier, (s, M_s). The sequence number assigned to a source message, (s, M_s), is the composition of the control message sequence number, t, and its position in the control message list. All receivers recover missing control messages, by sending NACK's for the sequence numbers t missing from their list. They then send NACK's to recover missing source messages that were acknowledged by the control messages. When the receivers recover the missing control messages, they place the source messages in the same order.

In M-RBP a receiver transmits the token at its scheduled time, whether or not it receives, or recovers, the control message from the previous token sites. When a token site does not recover a missing control message before transmitting its own control message, it may acknowledge source messages that were previously acknowledged, and source messages may receive several sequence numbers. Receivers recover all control messages that are received by a majority of the group, or leave the group. When multiple control messages sequence the same source message, the lower-numbered token t takes precedence at each receiver, and unique sequencing is preserved.

Aggressive token passing, rather than waiting to receive a token, allows M-RBP to continue to operate when receivers leave the group. The receivers use a distributed voting procedure to determine when a receiver that was scheduled to send the control message has left the group. When the vote is complete, if a majority of receivers vote that a receiver failed to transmit a control message, all of the receivers remove that receiver from the token list and the protocol continues to operate.

The e^{th} control message is scheduled to be transmitted at time t_e. The other receivers begin a recovery process if they do not receive the message after the maximum propagation time, which is very small in a local broadcast group. The maximum allowed recovery time for the control message, is $T_A = (n_{\max} + 1/2)T_R$, where T_R is the time between recovery attempts, and $n_{\max}$ is the maximum number of recovery attempts before giving up. The control messages that are transferred after $t_e + T_A$ include a vote on whether or not the control message at t_e was transmitted. If there are m receivers in the group, at $t_e + 2T_A + m\Delta_T$, all of the receivers have voted and the control messages with the votes have been recovered by all of the receivers that can recover the control messages. The vote is then tallied at each of the receivers. A similar vote is started for the messages acknowledged by the e^{th} control message to decide which messages are included in the final sequence. That vote is started at time $t_e + (2n_{\max} + 1/2)T_R$, which is the maximum time to recover the acknowledged messages after recovering the acknowledgment.

When the vote for the e^{th} control message is tallied:

1. There are $A(e)$ receivers that have received the e^{th} control message,

2. There are $B(e)$ receivers that have not received the e^{th} control message,

3. And, there are $C(e)$ receivers that have left the group, and have not voted.
$C(e) = m_e - (A(e) + B(e))$, where m_e is the number of receivers in the group when the vote starts. If $A(e) < m_e/2$, the receiver that was scheduled to transmit the control message is voted out of the group and it is not counted in m_e in future votes.

Each receiver may not receive all of the votes. At receiver r_j, $A_j(e) \le A(e)$, and $B_j(e) \le B(e)$. We require r_j to make the correct decision or leave the group itself. At r_j :

- If $A_j(e) \ge m_e/2$, then $A(e) \ge m_e/2$, and r_j leaves the receiver that transmitted the e^{th} control message in the group.
- If $B_j(e) > m_e/2$, then $B(e) > m_e/2$, $A(e) < m_e/2$, and r_j removes the receiver that transmitted e^{th} control message from the group.
- If $B_j(e) \le m_e/2$, and $A_j(e) < m_e/2$, r_j is uncertain whether or not $A(e) < m_e/2$, and leaves the group itself.
- If $A_j(e) \ge m_e/2$, but r_j has not recovered the e^{the} control message, then r_j leaves the group.

One token round later, the receivers know which receivers have received the e^{th} control message and have remained in the group. If more than half of the receivers leave the group on a particular token passing round, the group dissolves and is reformed by individual receivers joining a group with nearby receivers.

2.4 Reliable Neighborcast Protocol - RNP [6]

Our objective is to have each vehicle on a very long highway, with many vehicles, communicate reliably with its neighbors. The neighborhood for each vehicle is different, but the neighborhoods of nearby vehicles will overlap. As shown in the fogure.

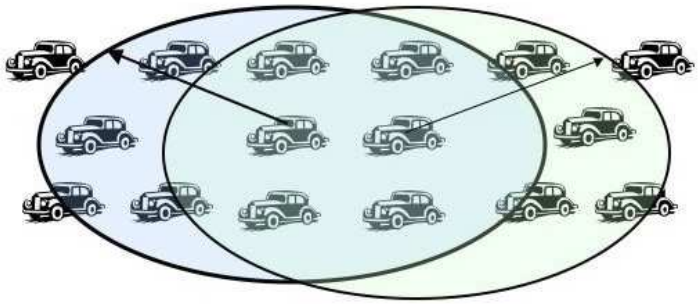


Figure 2. Neighborhoods

We define neighborcast as communications with all of the vehicles in your neighborhood. Neighborcast is different from broadcast. Broadcast provides communications between all of the nodes in a group. The group of vehicles that communicate in the figure is the union of all overlapping neighborhoods, which may include all of the vehicles on the highway. Neighborcast only requires that each vehicle communicate with its neighbors, which may be a small subset of the broadcast group.

RNP is a reliable neighborcast protocol. It is implemented as an overlay on any reliable broadcast protocol with small, overlapping groups. It transfers the guarantees that the reliable broadcast protocol provides to the small groups of receivers to each neighborhood.

We will build RNP on top of MRBP. This implementation of RNP provides

1. guaranteed delivery,

2. bounded delay,

3. a new type of relative sequencing between neighborhoods,

4. knowledge of which vehicles are currently in a neighborhood,

5. knowledge of which vehicles have received which messages, and

6. knowledge of which vehicles know what information a specific vehicle has acquired.

RNP is implemented on top of overlapping reliable broadcast protocols that cover regions of a highway. A vehicle participates in all of the reliable broadcast groups that cover its location.

There is a separate algorithm that moves the broacast group at the average speed of the vehicles in the group, so that vehicles do not change groups frequently, and also splits and joins groups to reflect changes in the relative location of vehicles. This algorithm is not described here.

A highway is modeled as a one-dimensional network of vehicles that can move with respect to one another. The size and overlap of the MRBP groups is selected so that all of a vehicle's neighbors are in at least one of the MRBP groups that cover the vehicle's location. This allows a vehicle to communicate with its neighbors without forwarding messages between broadcast groups.

Consider the simple example in the figure below. In this example we show a one-dimensional system with three MRBP groups and 21 evenly spaced vehicles. A vehicle's neighborhood is the vehicles three in front and three behind the vehicle. Eleven of the vehicles transmit and receive messages from two broadcast groups, and the remaining vehicles transmit and receive in one broadcast group. With this overlap, every vehicle belongs to at least one broadcast group with each of its neighbors. For instance, vehicle 5 has all of its neighbors, vehicles 2, 3, 4 and 6, 7, 8, in group 1, while vehicle 8 has its neighbors 5, 6, 7 and 9 in group 1 and its neighbors 7, 9, 10, and 11 in group 2. Note that its neighbors 7 and 9 are in both broadcast groups.

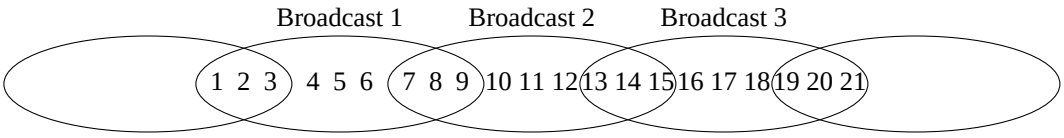


Figure 3. Overlapping broadcast groups, with evenly spaced vehicles, in a one dimensional network

From the example it is clear that there are many arrangements of broadcast groups that satisfy the constraints on neighborhoods. On one extreme there can be a single broadcast group that covers the entire highway. The disadvantages with this configuration are that 1) many of the guarantees are not provided until the token has cycled through all of the vehicles, 2) vehicles that participates in the group must receive or recover many messages which are not used, 3) powerful transmitters are required to have single hop transmissions over the entire highway and 4) there are a very large number of sources sharing the channel. The broadcast groups should be small.

At another extreme, each vehicle can have a broadcast group that covers its neighborhood. When a vehicle has N_n neighbors, it belongs to $N_n + 1$ broadcast groups. The disadvantage with this configuration is that each source message must be acknowledged in $N_n + 1$ separate groups, and the source message sequences in the different groups must be coordinated. Messages should be transmitted in a small number of broadcast groups.

In order to quantify the differences between the configurations, we define the overlay efficiency, η_{ov}, as the number of new messages received from neighbors over the total number of messages received. Assuming that each vehicle transmits about the same number of messages, $\eta_{ov} = \underset{v}{\text{Average}}\left[\frac{N_n(v)}{\sum_{v \in B_i}(N_{B_i} - 1)}\right]$, where v is a vehicle, B_i are the broadcast groups that cover v, and N_{B_i} is the number of vehicles in B_i. In a configuration with a single broadcast group, the number of vehicles on the highway is $N_H \gg N_n$, and $\eta_{ov} = \frac{N_n}{N_H} \to 0$. In the configuration with one broadcast group per vehicle, each vehicle receives messages, many of which are duplicates, from N_n vehicles in $N_n + 1$ broadcast groups, $\eta_{ov} = \frac{N_n}{N_n(N_n + 1)} = \frac{1}{N_n + 1}$.

In the example in figure *F_overlap*, with one broadcast group per vehicle, $N_n = 6$ and $\eta_{ov} = .14$. In the configuration in that example, $N_B = 9$, 1/3 of the vehicles belong to one group and 2/3 of the vehicles belong to two groups, $\eta_{ov} = \frac{1}{3}\frac{6}{8} + \frac{2}{3}\frac{6}{16} = .5$. If the group size in the figure is reduced to 6, with the same overlap, the efficiency increases to .6 .

In general, vehicles and broadcast groups are not evenly spaced, a vehicle's neighborhood may extend further in one direction than another, and each of the groups and neighborhoods may have a different size, as shown in the figure below.

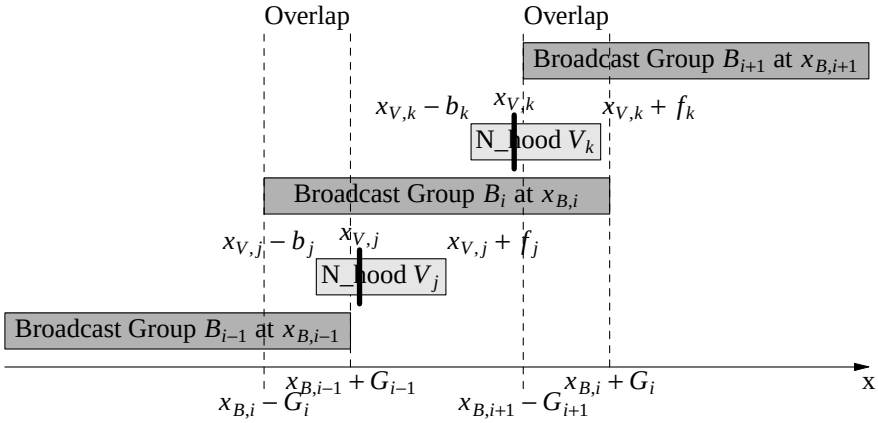


Figure 4. A general one-dimensional network

In [6] we prove that we can pick the overlap in a one-dimensional network so that every vehicle can communicate with its neighbor by participating in at most two broadcast groups. We also establish the following relationships, which are the rules we use in our mobility management protocols:

1. Each vehicle participates in at most 2 groups when the maximum distance from the center to the edge of a broadcast group is less than the minimum distance between the centers of broadcast groups.

2. Any vehicle can communicate with all of its neighbors without forwarding between groups, when the smallest overlap between groups is greater than the greatest distance between a vehicle and its neighbor.

In order to guarantee that each vehicle belongs to at most two groups we place an upper bound on the overlap. In order to guarantee that every vehicle can communicate with its neighbors in one of its groups we place a lower bound on the overlap. In order to simplify the lower bound we make every neighborhood symmetric with the distance in each direction equal to the greatest distance from a vehicle to one of its neighbors. When vehicles can communicate with vehicles in this extended neighborhood, they can definitely communicate with vehicles in their actual neighborhoods.

An underlying MRBP group includes all of the vehicles in an area. A vehicle participates in all of the MRBP groups that cover its location. The vehicle transmits it messages in all of these groups. A separate RNP operates in each vehicle, and joins the messages from each of its MRBP groups. The RNP overlay does not communicate with it's peer RNP overlays. Instead, it processes the messages from one or more broadcast groups and passes those messages to the application at a particular vehicle.

RNP filters the messages from the broadcast groups and provides the protocol guarantees to its neighborhood. The broadcast groups include messages that are from vehicles that are not in this vehicle's neighborhood, these messages are filtered out. Some messages are received from more than one of the broadcast groups, the duplicate messages are removed. Finally, when two messages appear in more than one broadcast group the messages may be in a different order in the two groups. This may occur when the first message is retransmitted in one of the groups.

Complete sequencing over the entire network is an unnecessary burden, since any vehicle does not receive

most of the messages. We define partial sequencing in the neighborhoods as two messages having the same order at any vehicles that must receive both. We can resolve differences in sequences by using accurate clocks and time-stamping the source messages the first time that they are transmitted. associates time-stamps, rather than sequence numbers with acknowledgments.

2.5 Fail Safe Operation

Application: Safe Driving Systems.

Objective: To guarantee that either every participant in a collaborative operation uses data or a command at the same time, or none of the participants use the data or command.

MRBP has characteristics that are useful:

1. The scheduled token passing protocol guarantees that every operable station has a message, or knows that it has failed to recover a message by a deadline.
2. Successive token rotations inform each station which other stations have received a message, voted to include the message in its list of committed messages, know which other stations have committed to include the message, $\cdots$

Problem: With each token rotation of the token, the information concerning a message converges, but the type of information that each reciver is known to have changes as it passes the token, so there isn't any time when every receiver is known to have the same information, and when we can use that information at the same time everywhere.

Choices and modifications in MRBP:

1. The number of participants in the group is small, so that the token rotation time is smaller than the response time for driving maneuvers - on the order of
2. Transmit any data or command message when the vehicle has the token:
   - The token site both transmits and acknowledges its own message.
   - The maximum delay for a message to enter the system is a token rotation time.
   - When the protocol operates over an 802.11 network, there are no collisions because there is only one source transmitting at a time.
     This type of operation is common in token passing on loop and broadcast networks.
3. Require a unanimous vote for all messages and acknowledgements, and leave the group when the vote isn't unanimous.
   - If you cannot recover any message the vote will not be unanimous.
     The probability that a message cannot be recovered when devices are within communication range and are actively participating in the group is small, since multiple recovery attemps are made, and inability to recover a message requires multiple message losses.
   - Stop transmitting when you cannot recover the message.
     The vote would not have been unanimous.
   - If a site does not transmit, no site can recover the message, and the other sites will stop transmitting.
   - One token rotation period + the maximum message recovery period after a message is transmitted, either every site has the message, or every site has left the group. It is safe to use the message.

In fail safe operation, when there is one participant that becomes vulnerable when a command or data is used, all of the participants that are not vulnerable commit to their operations when they recieve the message, and the vulnerable participant performs his operation after the token circulation plus the recovery

period, knowing that the other participants have committed to their participation.

This is the basis for the lock protocol. The master station asks for a lock until a deadline. The master proceeds with the lock after all other stations have granted the lock, and will maintain that lock until a deadline.

In the driver assisted merge protocol, the vehicle that will merge between two other vehicles is vulnerable if the other vehicles do not maintain the gap. The merging vehicle requests a lock, the other two vehicles commit to the lock as soon as they receive the request, but the merging vehicle does not act upon the lock until after he receives an acknowledgement for the other two vehicles. There is no danger if either or both of the first two vehicles hold onto a lock, but the merging vehicle doesn't use it. But a dangerous situation could orrur if the merging vehicle thought that the other two vehicles are cooperating, but they are not.

In a shared memory system:

— Each participant can transmit an update to the memory when it has the token, at its scheduled token transmission time, t_x.

— Vehicles that do not receive the message start a recovery immediately, with a deadline for recovery at $t_x + T_R$, where T_R is the time for the maximum number of recoverys.

— Any station that does not recover the message stops transmitting its token after $t_x + T_R$, and every station knows that it has stopped within T_R of its scheduled transmission.

— If any station has not received message t_x, every station knows that it has missed the message by $t_x + 2 * T_R + T_T$, where T_T is the time for the token to cycle around the group of receiviers.

— When there is one receiver that becomes vulnerable by using the information in a message, if the other receivers do not use the message,

  • the stations that are not vulnerable commit to use the message before $t_x + T_R$, if they receive the message.
    They commit to use this message, whether or not they remain in the broadcast group.

  • the station that becomes vulnerable uses the information by $t_x + 2 * T_R + T_T$, if it receives the message from every other station.

— If there is more than one sation that becomes vulnerable if any other station uses the information, we cannot determine when to use the information.

Protocols should be designed so that only one participant is vulnerable at a time. Since the token circulates rapidly, each station can be made vulnerable in turn.

REFERENCES

[1] N. F. Maxemchuk, "Video Distribution on Multicast Networks," IEEE JSAC, April 1997, vol.15, no.3, pp. 357-372.

[2] J-M. Chang, N. F. Maxemchuk, "Reliable Broadcast Protocols," ACM Transactions on Computer Systems, Vol. 2, No. 3, Aug. '84, pp. 251-273.

[3] N. F. Maxemchuk, D. Shur, "An Internet Multicast System for the Stock Market," ACM TOCS, Aug. 2001.

[4] T. Willke, N. F. Maxemchuk, "Reliable Collaborative Decision Making in Mobile Ad Hoc Networks," Management of Multimedia Networks and Services: 7th IFIP/IEEE International Conference, MMNS 2004, San Diego, CA, USA, October 3-6, 2004. Proceedings, pp. 88-101.

[5] T. L. Willke, N. F. Maxemchuk, "Coordinated Interaction Using Reliable Broadcast in Mobile Wireless Networks," Networking 2005, May 2 - 6, 2005, University of Waterloo, Waterloo Ontario Canada.

[6] N. F. Maxemchuk, P. Tientrakool, T. Willke,, "Reliable Neighborcast," IEEE Trans. on Vehicular Technology, vol. 56, iss. 6, Part 1, Nov. 2007, pp. 3278-3288.