Error control for Continuous Media and Multipoint Applications

PROPOSAL DEFENSE

Christos Papadopoulos

<christos@dworkin.wustl.edu>

July 28, 1995

This work is sponsored by:

- ARPA
- National Science Foundation
- Ascom Nexion
- Bell Northern Research
- NTT
- NEC America
- Southwestern Bell
- Bay Networks
- Tektronix
- Samsung
- NIH

Motivation and Perspective

Multimedia Multipoint applications pose new challenges:

Application Requirement	Options				
connection configuration	one-to-one	one-to-many	many-to-many		
connection size	small (2-10).....	medium (10-100)......	large (100-1000)		
quality of service	none	loss	delay	jitter	media synchronization
error control	none	application-specified	100%		
flow control	none	variable/policy-based	fully variable		
latency	100 µSec..........	33 mSec.......	200mSec.........	1Sec........	

Too many challenges...

We concentrate on just a few dimensions of the problem:

1. Error control for point-to-point continuous media applications
2. Implosion control for error control in multipoint applications with large number of participants

WASHINGTON UNIVERSITY
ONE BROOKINGS DRIVE
ST. LOUIS MISSOURI

1: Error Control for Continuous Media Applications

Motivation

Continuous media applications require high throughput and low latency: reliability may suffer!

CM applications can trade some reliability for lower latency:

- CM data has limited lifetime
- CM data contains high redundancy

Options for CM transmission

- Ensure loss-free transmission (peak-bandwidth allocation)
- Concealment
- Forward Error Correction (FEC)
- Retransmission

We believe that retransmission can be successfully used in CM

- Low cost (compared to FEC)
- Reduces BW requirements (compared to FEC and peak-bw allocation)
- Will help concealment

WASHINGTON UNIVERSITY
ONE BROOKINGS DRIVE
ST. LOUIS MISSOURI

Retransmission-Based Error Control

Questions:

- Is recovery possible with continuous media applications?
- How much does it cost (bandwidth, complexity, latency)?
- How does it affect concealment?

Selected methodology:

- Design an actual mechanism
- Implement the mechanism in a protocol
- Experiment

Related work:

Dempsey, Libeherr, Weaver

- audio

IBM European Networking Center, Heidelberg

- similar to our work
- implementation status unknown

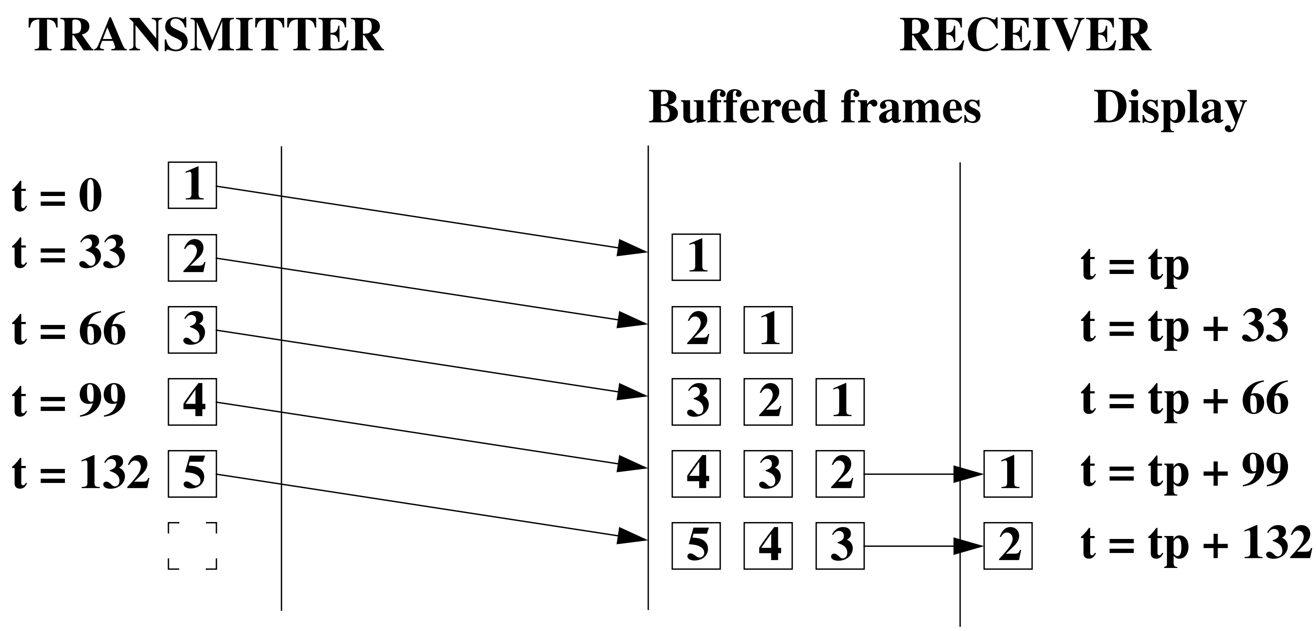
Features

- Play-out buffering
- Conditional retransmission
- Gap-based loss detection
- Implicit expiration of sender retransmission buffers
- Delivery of data integrity information to the application

Mechanism has already been designed and implemented

Playout Buffering

Users can tolerate 200 mS delay for interactive applications



tp: propagation delay (typically 1 - 50 mS)

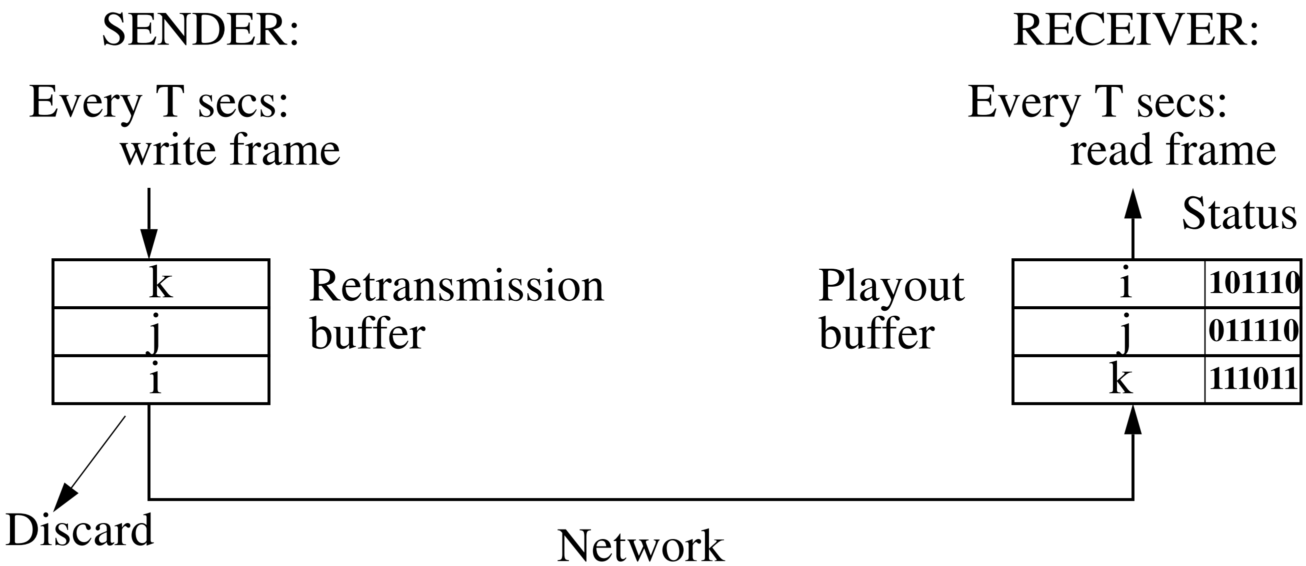
Playout buffering allows:

- additional time to retransmit lost packets
- jitter absorption

Trade-offs:

- trade latency, buffering and complexity
- gain lower error rate to application and less stringent requirements from the network

Send and Receive Actions



SENDER

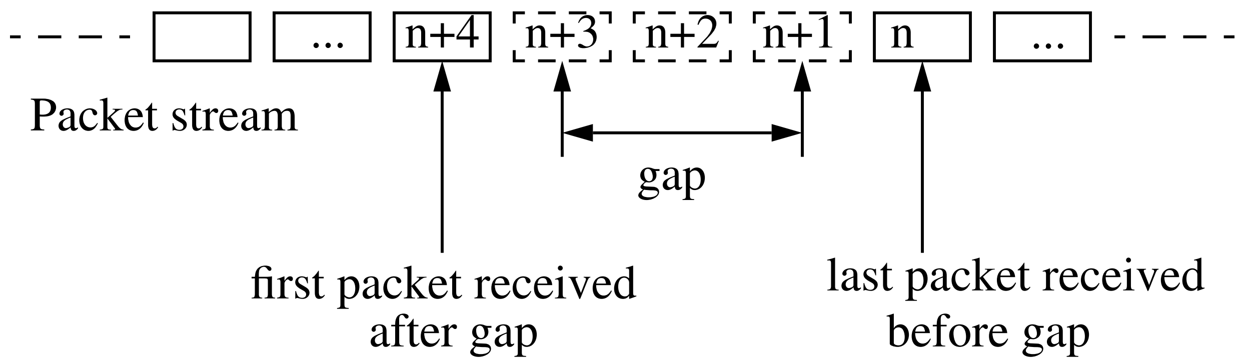
- Application writes frames with period T (e.g. 33 mS)
- Protocol transmits new frame immediately and stores copy in retransmission buffer
- Each new frame ejects oldest frame from retransmission buffer

RECEIVER

- Application reads frame+status with period T
- If new frame finds the playout buffer full, a frame is dropped. Possible candidates: (a) oldest frame, (b) frame with most errors, (c) least important frame (e.g. not I-frame)
- If application attempts to read when playout buffer is empty, application is put to sleep until buffer fills up

Gap-based Loss Detection

Error detection done with sequence numbers:



Gap Detection!

- First packet after gap triggers gap-detection
- No need for per-packet retransmission timers
- Gap-detection is faster than timeout-detection for CM
- Gap-detection avoids retransmissions during network loss periods (unless gateways do random discard)

Conditional retransmission:

As soon as a gap is detected, calculate T_{left} = time left before display.
$T_{left} = n * T$, where:

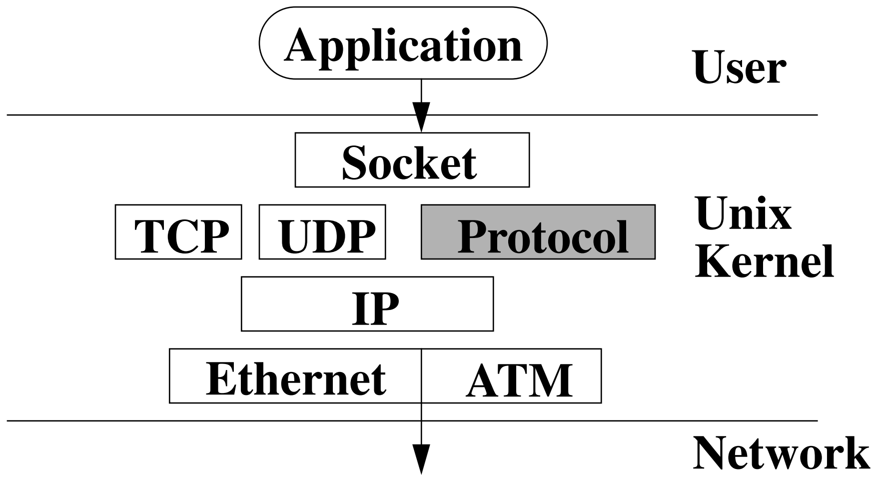
n: frames in playout buffer

T: rate application reads frames

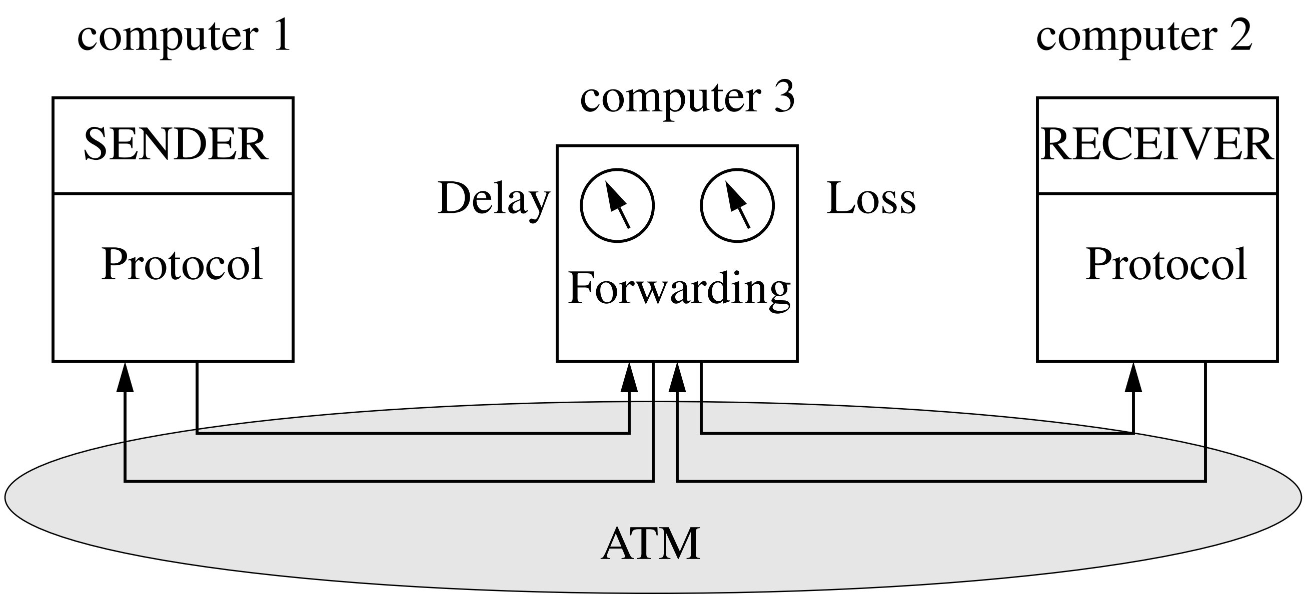
If T_{left} > RTD, then send retransmission request

WASHINGTON UNIVERSITY
ONE BROOKINGS DRIVE
ST. LOUIS MISSOURI

Implementation



Experimental Setup:



Implemented!

- RTD granularity: 2 mSecs
- experimentation ready to begin

WASHINGTON UNIVERSITY
ONE BROOKINGS DRIVE
ST. LOUIS MISSOURI

Evaluation Plan

1. Run experiments varying RTD, playout buffer and loss model
2. Instrument the protocol and investigate trade-offs between
   - playout buffer size
   - RTD
   - error recovery success
3. Run experiments with MPEG streams (with help from NEC):
   - transmit MPEG stream
   - save frame status for each frame to a file
   - do off-line concealment
   - display the resulting MPEG stream
4. Consider slice per packet - may help MPEG concealment

WASHINGTON UNIVERSITY
ONE BROOKINGS DRIVE
ST. LOUIS MISSOURI

2: Multipoint Protocols: Perspective

Previous work on Reliable Multicast for Distributed Computing Applications:

- Chang & Maxemchuk (1984)
- Isis group (Birman & Joseph)
- Crowcroft & Paliwoda (1988)
- Tanenbaum et. al. (1989)
- Armstrong, Freier, Marzullo (RFC 1301, 1992)
- Braudes, Zabele (RFC 1458, 1993)
- RMP (1994)
- ... others

Main characteristics:

- 100% reliability
- strict or causal message ordering constraints
- poor scalability
- poor throughput performance
- implemented mostly on datagram networks

Error Control for Multipoint Applications

Motivation

Multicast connections in the future:

- Hundreds or more participants (e.g. news distribution, stock-market information update)
- Senders still need feedback from receivers for error and flow control
- Application latency requirements critical (e.g. CM)

WASHINGTON UNIVERSITY
ONE BROOKINGS DRIVE
ST. LOUIS MISSOURI

Current Related Work

Main focus shifted to implosion control:

- XTP (1992)
- Bolot, Turletti, Wakeman (1994)
- Paul, Sabnani (1994)
- Pingali, Kurose (1994)
- Van Jacobson et. al. (1995)
- Cheriton et. al. (1995)
- Yavatkar et.al. (1995)

Issues being addressed:

- scalable information solicitation (flow control)
- support for Distributed Interactive Simulation (DIS)
- hierarchical implosion control (pre-configured, shortest-path topology)

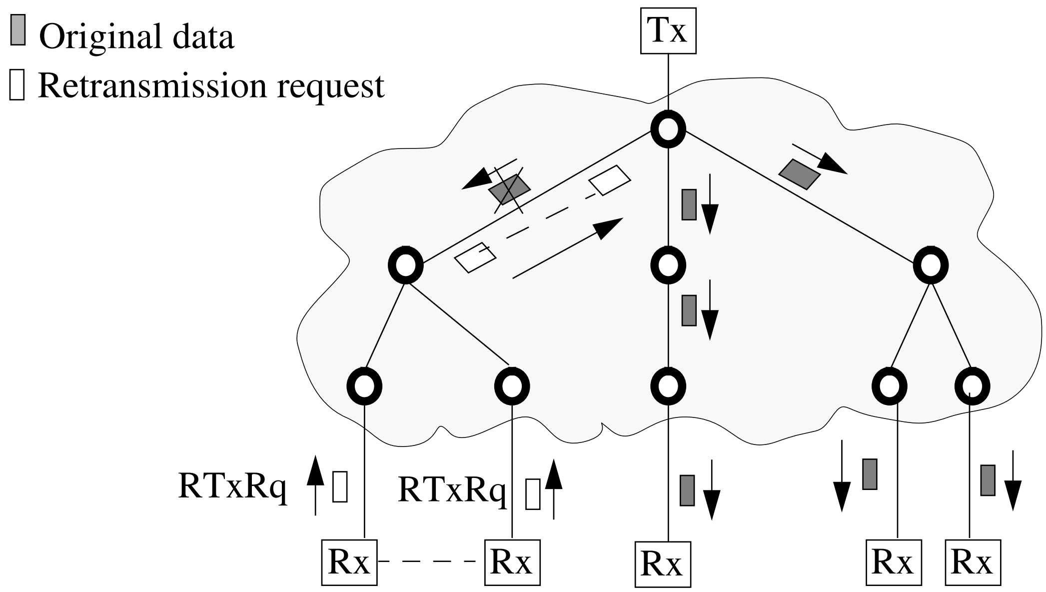
My work: (1994 - 1995)

- 2 IPP progress reviews
- 2 graduate student research days
- various postings on end2end mailing list

WASHINGTON UNIVERSITY
ONE BROOKINGS DRIVE
ST. LOUIS MISSOURI

Implosion in Error Control

Example:



Possible consequences:

- The sender is swamped with messages and loses replies
- Severe latency is introduced at the sender due to processing of retransmission requests
- Network bandwidth requirements hard to predict

Estimating implosion is very complex:

- loss probability, topology, location of loss
- using a simple model: 3 orders of magnitude increase in packets compared to point-to-point

Optimal Implosion Control Criterion

Optimality criterion:

optimal solution should resemble point-to-point recovery both in terms of messages and latency

Messages:

1. Recovery-related messages reach *only* endpoints experiencing loss and the retransmitting entity
2. Only one retransmission request is generated per subtree
3. Only one retransmission reaches each endpoint that experienced loss

Latency:

4. A retransmission request is sent immediately by the endpoint that discovers the loss first
5. A retransmission is sent immediately by endpoint closest to the requesting endpoint

Additional condition:

Participation in error recovery is optional. Recovery should be completely transparent to endpoints that do not participate

Implementation of Implosion Control

Attention must be paid to:

1. Implosion-control processing:
   - negligible overhead during normal protocol operation
   - not more than a O(100) of instructions per packet
   - optimized critical path

2. Distribution of processing:
   - avoid bottlenecks, improve scalability

3. State information: resist temptation to increase state

4. Timers: expensive - they involve interrupts + OS overhead
   - avoid per-packet timers
   - minimize or eliminate per-endpoint timers

5. Minimize number of additional control connections and minimize control bandwidth

WASHINGTON UNIVERSITY
ONE BROOKINGS DRIVE
ST. LOUIS MISSOURI

Implosion Control Design Taxonomy

Sender-controlled implosion

- All responses from receivers go to sender only
- Sender tells the receivers what to do
- Centralized control
- Requires communication between sender and each receiver

Receiver-controlled implosion

- All responses from receivers go to all other receivers and sender
- Implosion control decisions are taken by receivers; sender may or may not participate
- Distributed control
- Requires broadcast communication between receivers and sender

Hierarchical implosion control

- Receivers are structured in a tree hierarchy, where children can send responses only to their parent
- Implosion control decisions are taken by receivers; sender may or may not participate
- Distributed control
- Requires communication between each child and its parent

Note: Network participation not assumed

WASHINGTON UNIVERSITY
ONE BROOKINGS DRIVE
ST. LOUIS MISSOURI

Receiver-Controlled Implosion

Favored by the Internet community

Basic design first appeared in the XTP protocol

Current enhanced design implemented in wb - paper will appear in Sigcomm '95

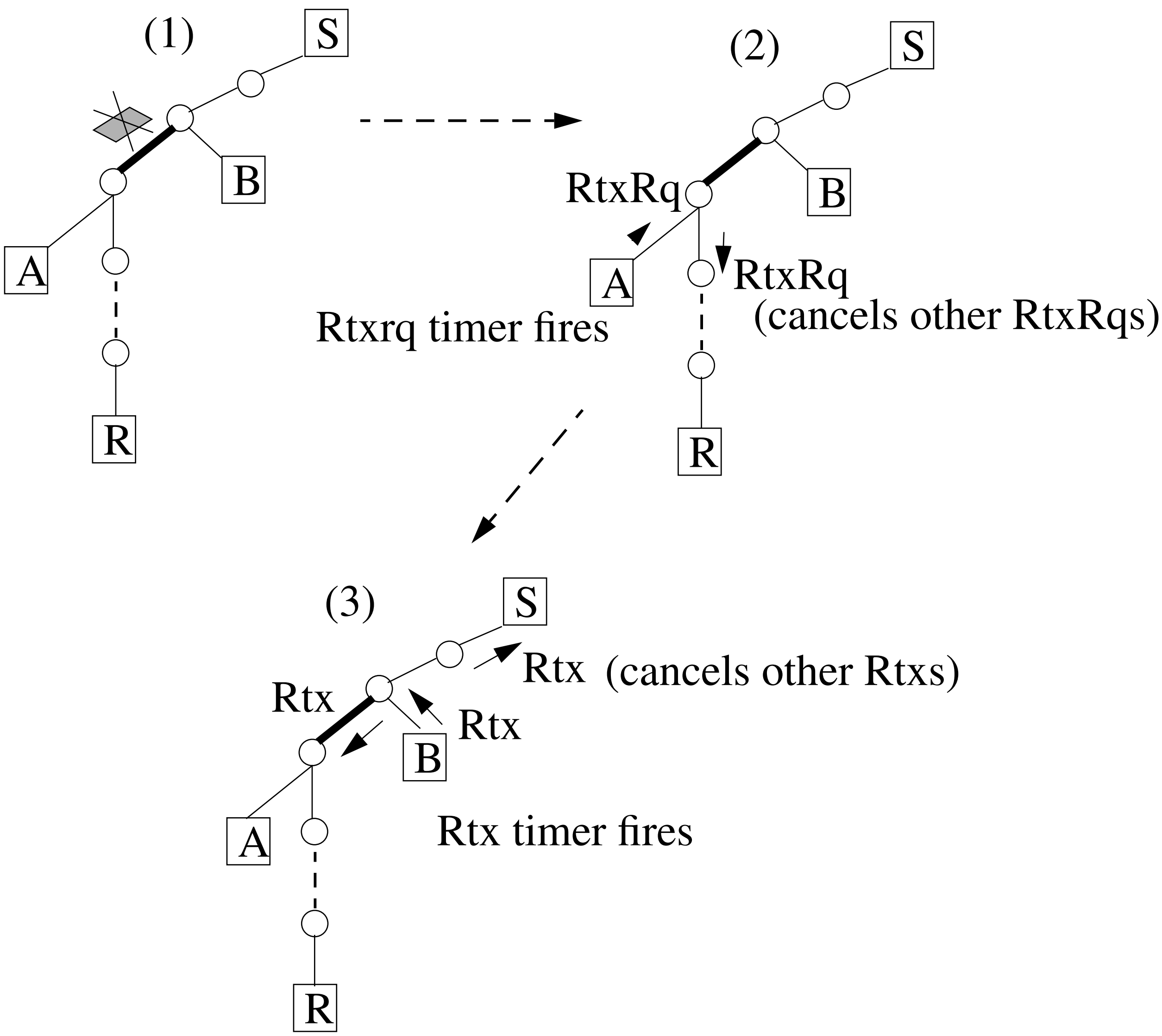
Requirements:

- inter-receiver communication (many-to-many connections)
- receivers able to buffer and retransmit lost data to others

Error Recovery

Two timers on each endpoint:

- REQUEST TIMER = $U[C_{1*}d_{SA}, (C_1 + C_2)*d_{SA}]$ secs
- REPAIR TIMER = $U[D_{1*}d_{AB}, (D_1 + D_2)*d_{AB}]$ secs



WASHINGTON UNIVERSITY
ONE BROOKINGS DRIVE
ST. LOUIS MISSOURI

Features:

- moderate latency (2 - 3 times RTD to original source)
- even processing distribution
- transparent adding/deleting receivers
- endpoints receive all transmitted messages
- high receiver complexity
- non-zero probability of implosion

My plan:

- Investigate mechanism in context of hierarchical scheme
- Integrate in my simulation/implementation environment

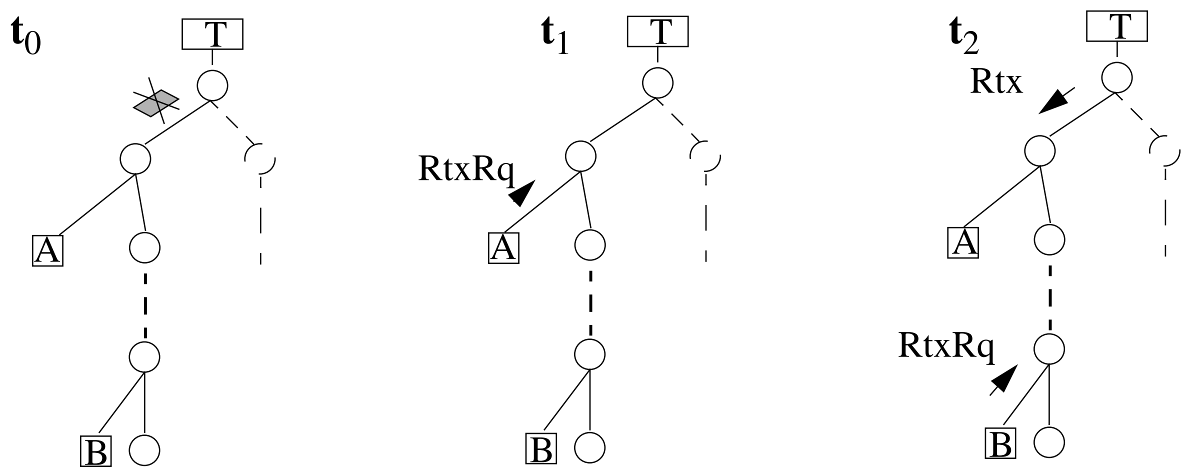
Sender-Controlled Implosion

Applications:

- One-to-many connections over connection-oriented networks
- Receivers cannot buffer or retransmit (low-power receivers, mobile communication)
- Receiver privacy or isolation is required

So far ignored by the Internet community

Basic idea:



Note: loss detection based on seq num gaps

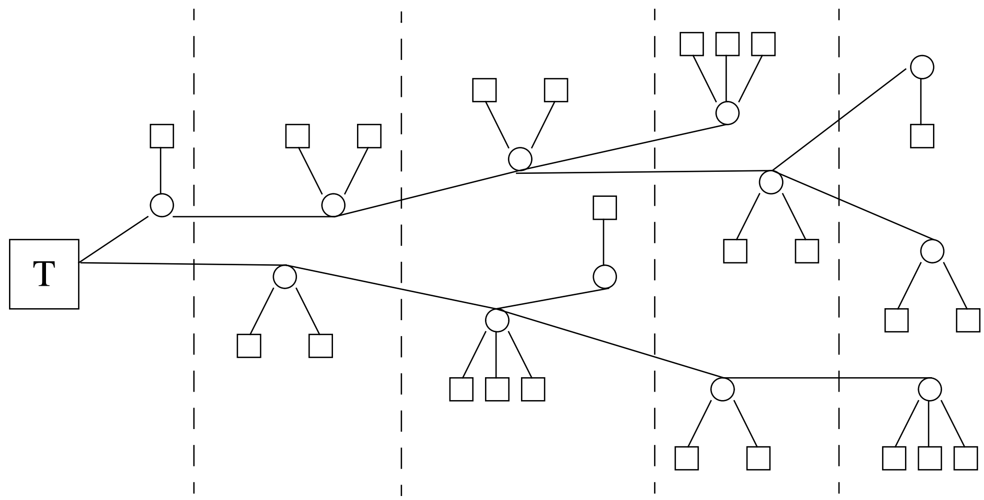
Key observations:

- Receivers closer to sender discover loss sooner
- Each receiver should back off for interval proportional to RTD from sender
- Receivers discover other receivers' actions only through sender

WASHINGTON UNIVERSITY
ONE BROOKINGS DRIVE
ST. LOUIS MISSOURI

Example

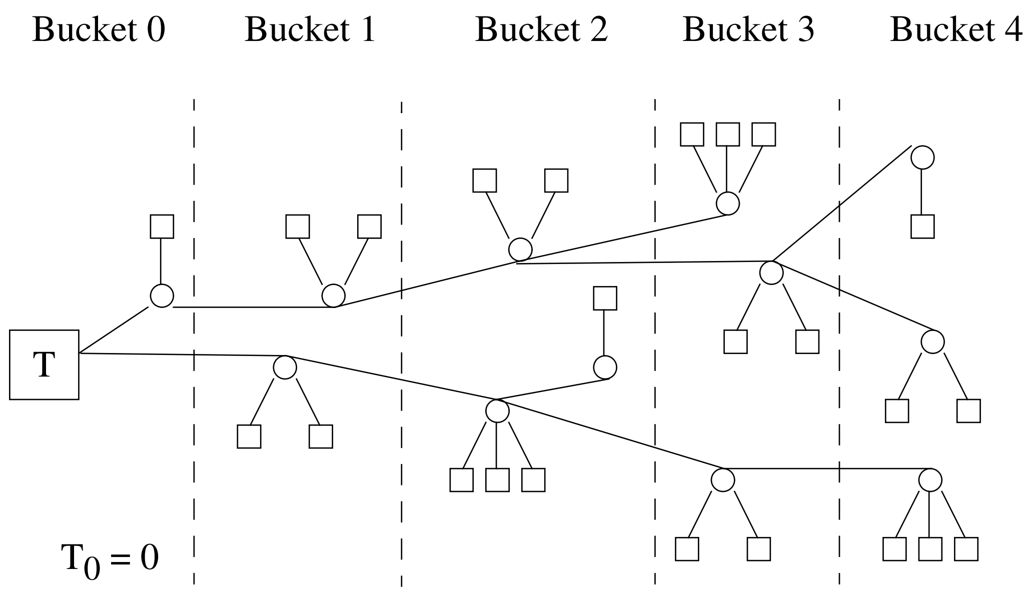
One-to-many connection



- Receivers divided into groups
- On detecting a loss, each group assumes that the previous group has detected the loss
- Assume the previous group has initiated a request and wait for a retransmission
- If none arrives, initiate a retransmission request

WASHINGTON UNIVERSITY
ONE BROOKINGS DRIVE
ST. LOUIS MISSOURI

Heuristic Algorithm



$T_0 = 0$

$T_1 = RTD_0$

$T_2 = RTD_0 + RTD_1$

$T_3 = RTD_0 + RTD_1 + RTD_2$

$T_4 = \ldots$

... continued...

For each bucket b:

- RTD_b = MAX (RTD of any receiver in bucket)
- back-off timer: $T_b = \sum_{k=0}^{b-1} RTD_k$

Sender state:

- T_b and bucket ID for each bucket
- histogram of receivers according to their RTD values

Receiver state (relayed by sender):

- Bucket ID
- T_b
- Max T_b

WASHINGTON UNIVERSITY
ONE BROOKINGS DRIVE
ST. LOUIS MISSOURI

Algorithm Steps

Receiver:

- On loss start T_b. If retransmission is received before T_b expires, cancel request
- If T_b expires send request and set T_b = Max T_b
- If no retransmission arrives send another request and double T_b. Assume failure after N iterations

Sender:

- On reception of first request, send retransmission immediately
- Ignore other requests from same bucket for interval T_b

Implosion: O (bucket size)

Latency: O (number of buckets)

Enhancements:

- Distribute receivers from same subtree to different buckets
- bring lossy receivers to nearer buckets
- Allow arbitrary bucket distribution of receivers by assigning uniform RTD - allows bucket rotation
- Allow select receivers to respond immediately with probability p

Features/Trade-offs

Features:

- Sender has direct control on implosion
- Minimal implosion-related processing at receivers
- Sender may become a bottleneck
- Potentially higher message latency than receiver-controlled implosion

Trade-offs:

- Implosion reduces as number of buckets increases
- Latency increases with number of buckets
- Sender selects desirable implosion/latency compromise based on # of receivers, loss rate, RTD

WASHINGTON UNIVERSITY
ONE BROOKINGS DRIVE
ST. LOUIS MISSOURI

Evaluation Plan

1. Create a simulation environment to evaluate performance

   Parameters:

   - number of endpoints
   - number of buckets
   - topology and bucket distribution
   - loss rate

   Measure:

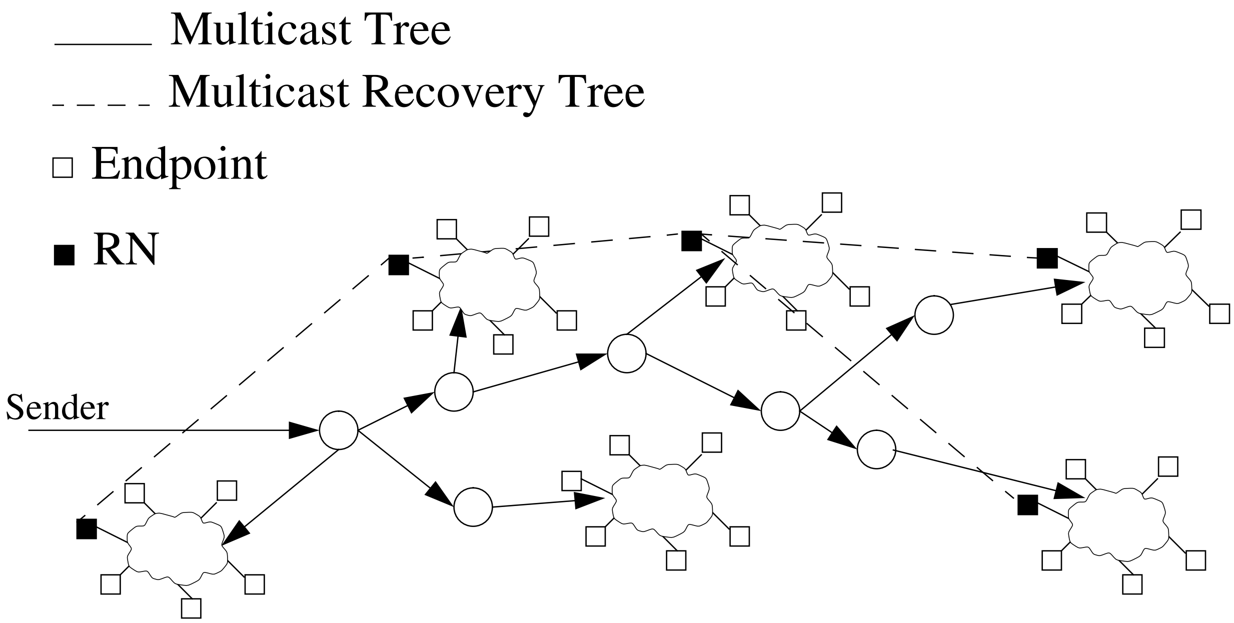
   - latency (compare with optimal and with other solutions)
   - implosion at sender
   - network bandwidth requirements

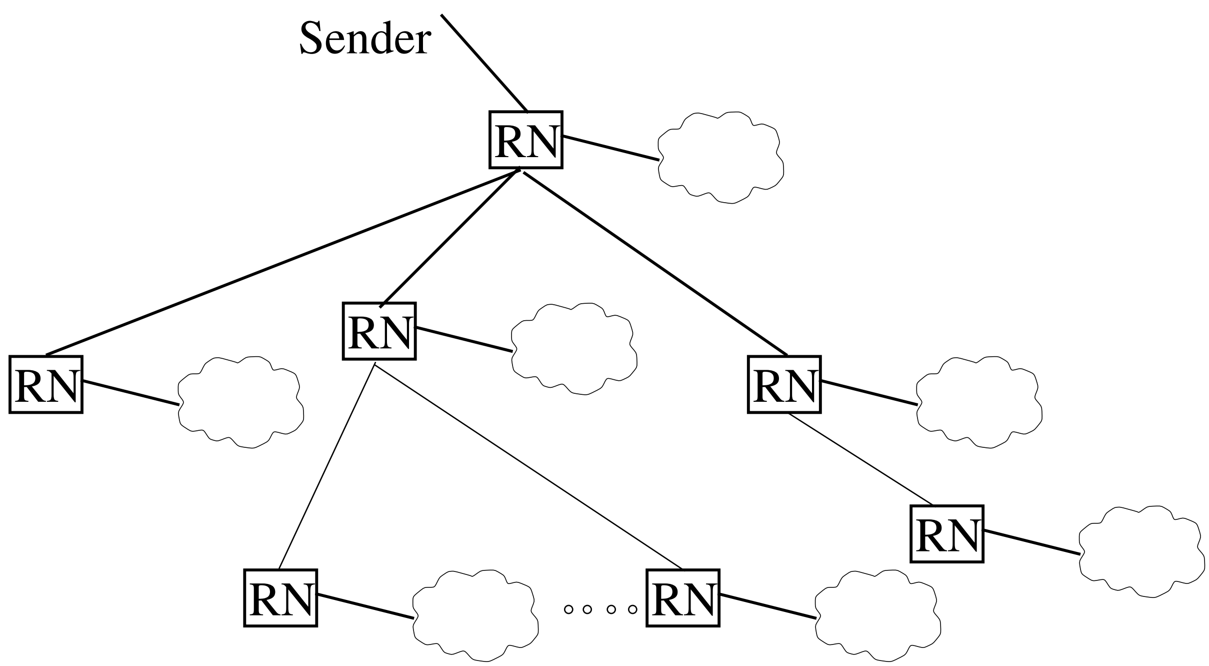
2. Protocol specification
   - packet formats
   - state machines

3. Implementation in SunOS or NetBSD over local ATM testbed
   - processing requirements (instruction count)
   - compare results with simulation

WASHINGTON UNIVERSITY
ONE BROOKINGS DRIVE
ST. LOUIS MISSOURI

Hierarchical Implosion Control





Multicast Recovery Trees (MRTs)

Basic ideas:

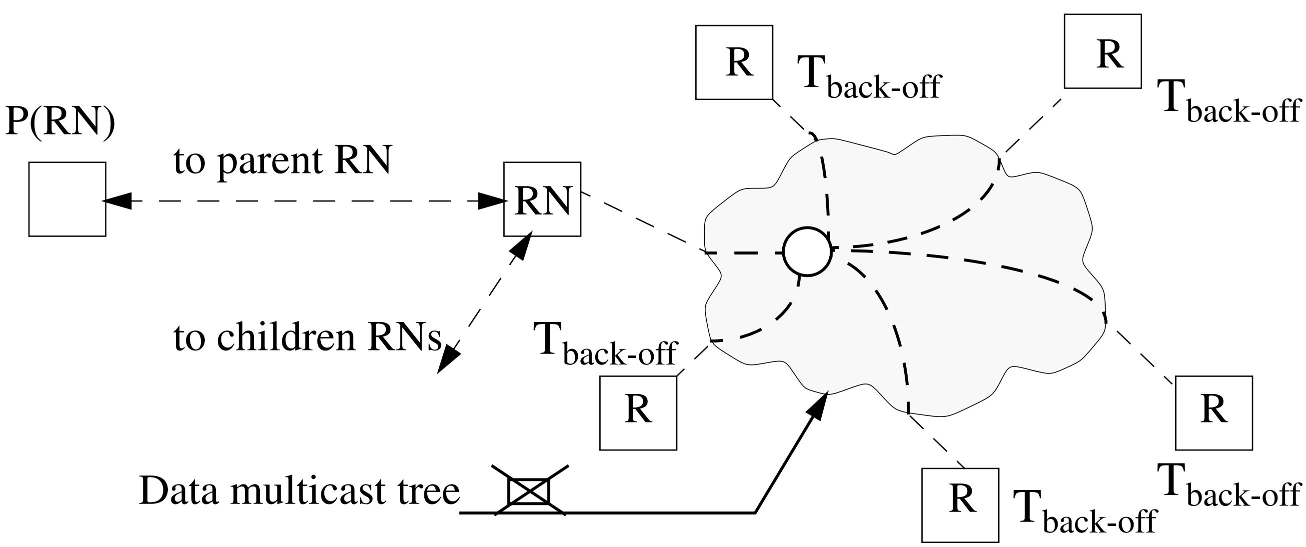
1. Sender establishes a unidirectional multicast tree (MT)
2. Receivers form a Multicast Recovery Tree (MRT) as follows:
   - Receivers group themselves in local groups
   - Each local group selects a relay node (RN)
   - A bidirectional connection between receivers and RN is created (each group is free to choose the connection type according to requirements and the local underlying network)
   - RNs connect to parent RNs via point-to-point connections
   - local groups may use sender or receiver-controlled implosion

Key observation: MRTs need not follow same topology as MTs

Heuristic Algorithm

A. Whole group detects loss

1. Each R starts local back-off timer $T_{backoff} >= T_{prop}(R, RN)$
2. RN immediately sends retransmission request to parent P(RN) and to all R in local group
3. If request from RN arrives before $T_{backoff}$ expires, R keeps quiet
4. If P(RN) has data, it sends copy to RN.
5. If P(RN) does not have data, it marks RN as "expecting" data
6. When retransmission arrives, RN sends copies to local group and any marked children



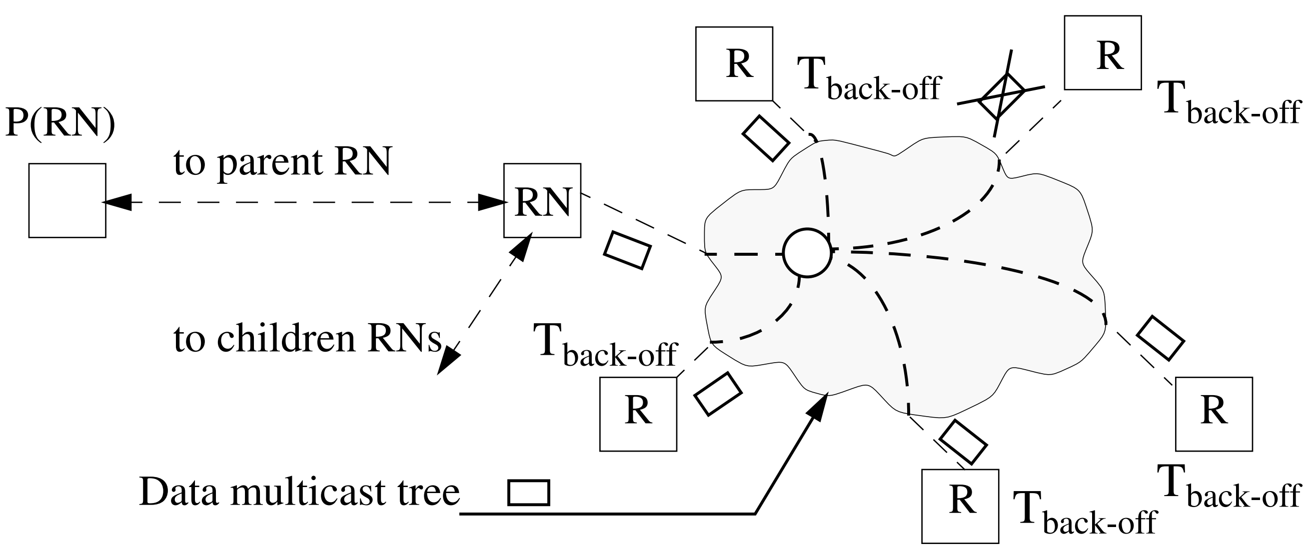
... continued

B. Some R detects loss, but RN has data

1. Each R sets $T_{backoff}$ as before
2. $T_{backoff}$ expires before notification from RN arrives
3. A request is send to RN
4. RN transmits missing data

C. RN detects loss, but some R has data

1. Notification from RN triggers a retransmission from R of missing data to RN
2. Or: P(RN) retransmits to RN



WASHINGTON UNIVERSITY
ONE BROOKINGS DRIVE
ST. LOUIS MISSOURI

Features/trade-offs

- Near-optimal performance: meets most optimality criteria
- Low latency
- Distributed processing
- Excellent scalability
- Higher complexity than sender/receiver controlled implosion due to formation and maintenance of hierarchical structure

WASHINGTON UNIVERSITY
ONE BROOKINGS DRIVE
ST. LOUIS MISSOURI

Multicast Recovery Tree Protocols

1. Local group formation/maintenance protocol
   - group membership (addition/deletion)
   - group scope
   - RN creation/discovery

2. MRT formation/maintenance protocol
   - topology (parent discovery)
   - membership operations (addition/deletion)

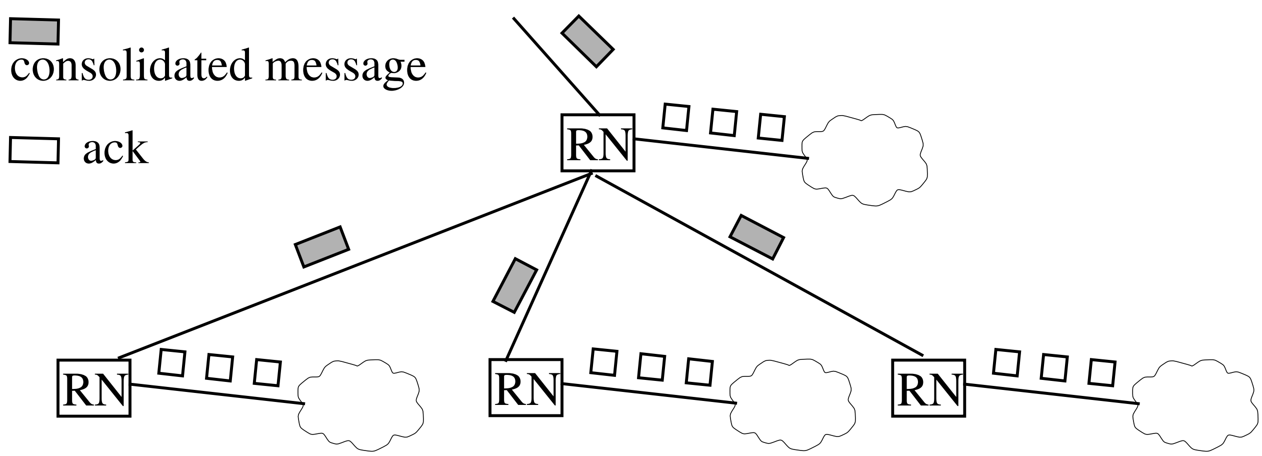
WASHINGTON UNIVERSITY
ONE BROOKINGS DRIVE
ST. LOUIS MISSOURI

Consolidation

Why do we need consolidation?

Consolidation is critical for reliable multicast protocols with large number of endpoints

- reliable protocols need to send positive acks
- positive acks are not redundant - must be delivered to sender
- positive acks pose a bigger implosion threat



WASHINGTON UNIVERSITY
ONE BROOKINGS DRIVE
ST. LOUIS MISSOURI

Consolidation Messages and Operations

Collection method	Explanation
Periodic	wait for time T, consolidate all collected messages and forward up the tree
Message per receiver	collect one message from every receiver
Sender initiated	sender requests message collection explicitly, specifying method

Operation	Example
MIN $(x_1, x_2,...x_n)$	highest packet sequence number received by all receivers
MAX $(x_1, x_2,...x_n)$	maximum window size advertized by any receiver
OR $(b_1, b_2,...b_n)$	combine error bitmaps from all receivers to determine packets lost by all receivers
AND $(b_1, b_2,...b_n)$	combine error bitmaps to determine which packets have been received by all receivers

WASHINGTON UNIVERSITY
ONE BROOKINGS DRIVE
ST. LOUIS MISSOURI

Plan of Action

1. Examine algorithms for MRT formation, maintenance

2. Examine location and interaction of MRT formation protocols
   - Coexist with, or be part of transport protocol
   - Part of network (IPv6), or reservation protocol (RSVP)?
   - Network daemons (user space)

3. Clearly define consolidation operations and requirements

4. Create a simulation environment and measure:
   - latency (compare sender/receiver controlled implosion)
   - implosion in terms of messages
   - network bandwidth requirements

5. Create an implementation in Unix over local ATM testbed
   - complexity and processing requirements

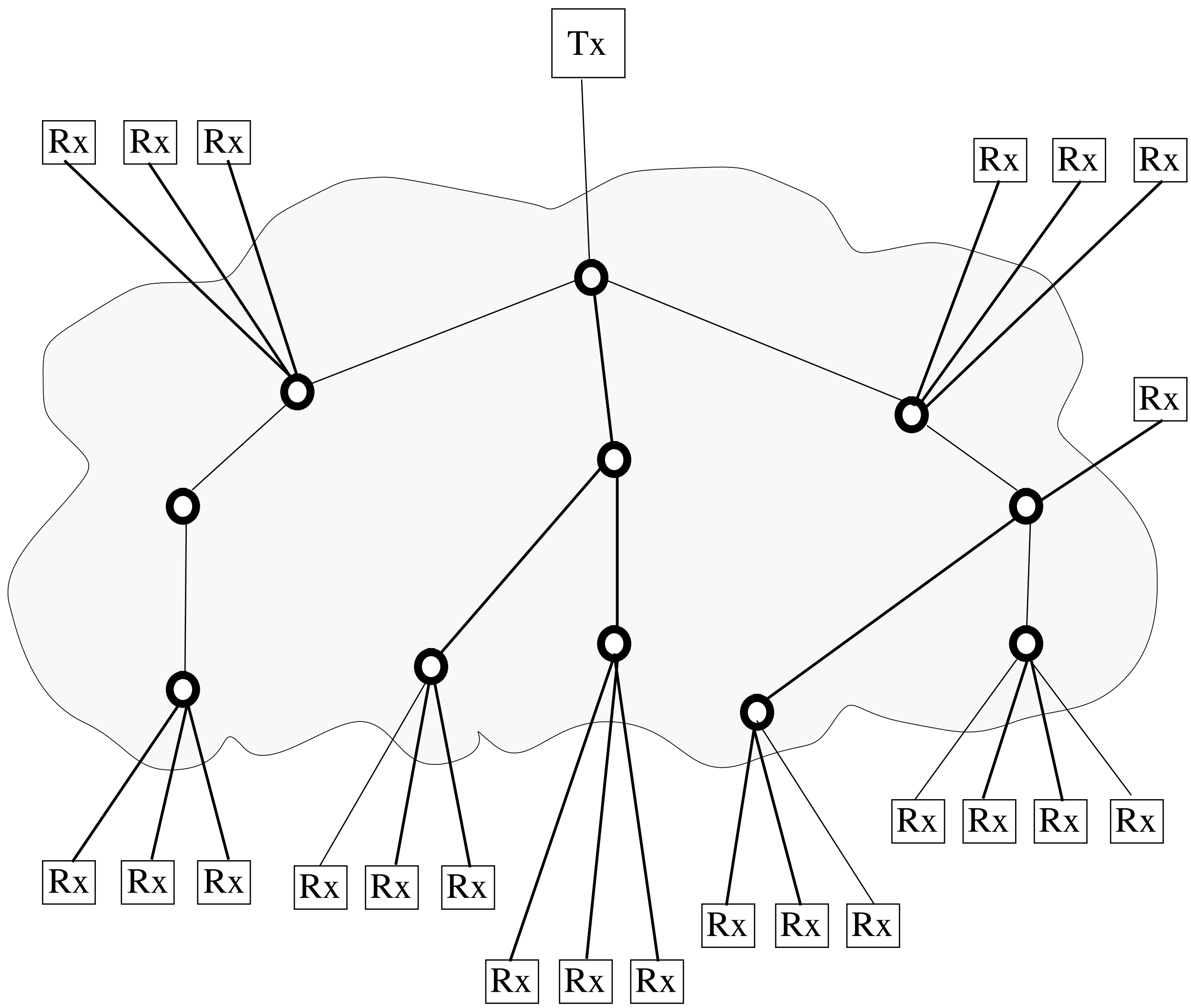
6. Understand impact of using sender or receiver controlled implosion in MRT groups

Summary

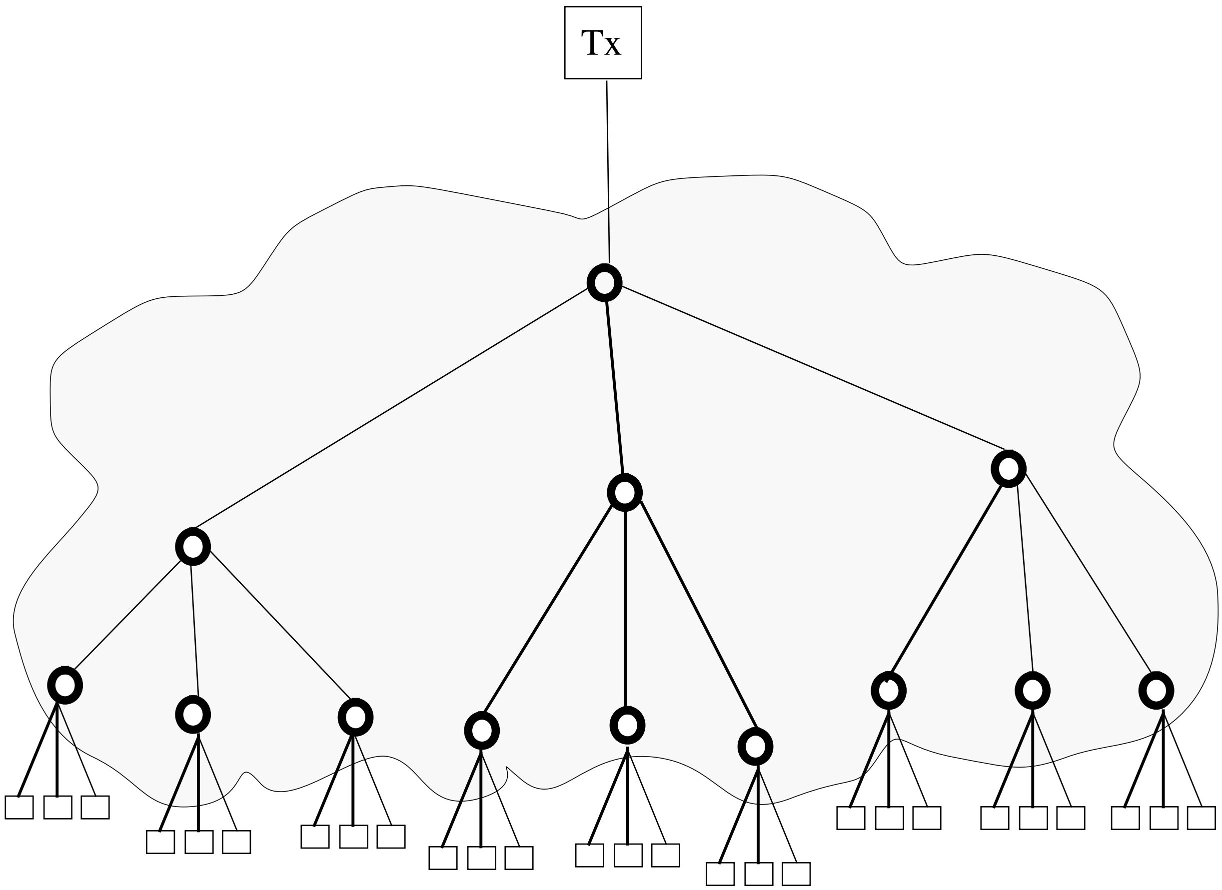
1. Implosion is a serious problem which must be addressed

2. Optimal solutions are hard - however, sub-optimal solutions look promising:
   - sender-controlled implosion
   - receiver-controlled implosion
   - hierarchical implosion control

3. Trade-offs vary widely - systematic evaluation needed via both simulation and implementation



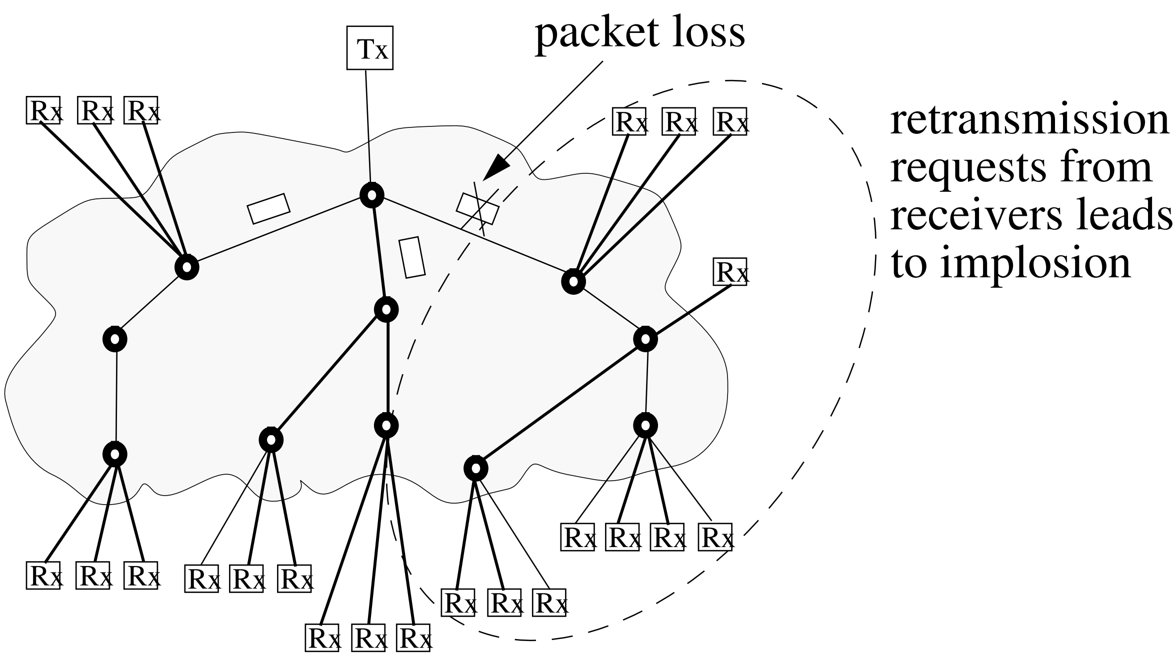
WASHINGTON UNIVERSITY
ONE BROOKINGS DRIVE
ST. LOUIS MISSOURI



k = 3
h = 3
27 endpoints

WASHINGTON UNIVERSITY
ONE BROOKINGS DRIVE
ST. LOUIS MISSOURI

Implosion control in multipoint transport protocols



Sender-controlled implosion

sender dictates implosion control actions

requires one-to-many bidirectional connection

Receiver-controlled implosion

receivers collaborate to control implosion

requires many-to-many connection

Hierarchical implosion control

receivers are organized in hierarchical groups

WASHINGTON UNIVERSITY
ONE BROOKINGS DRIVE
ST. LOUIS MISSOURI

requires point-to-point and one-to-many, or many-to-many connections among receiver groups

WASHINGTON UNIVERSITY
ONE BROOKINGS DRIVE
ST. LOUIS MISSOURI