Internet Architecture Papers

- End to end principle: put it at the ends since application knows best. Can make an exception for performance
- Clark 85 paper: Internet was designed for a set of military requirements
  - Design reflects that – examples?
- Clark 2002 paper: Tussles in cyberspace – implications for designer?

Application Model – Bulk Data Transfer

- Properties
  - Lots of data
  - Many receivers of different types
  - Real time?
- Example applications
  - Software distribution/updates
  - Pushing data to web caches
  - Internet radio stations/TV stations?
- Why not ARQ?

Administrivia/Projects

- Project
  - My daily plea to start ASAP
  - I may give an extension, but homework and critique will be assigned next week
  - Help session Friday 10am in room J-15 (Engineering building, first floor)
Two families of Erasure Codes Studied

- **Reed Solomon Codes**
  - Optimal encoding efficiency
  - But difficult to decode (quadratic in k)

- Tornado codes (invented by the authors in late 1990s)
  - Suboptimal (but pretty good) efficiency
  - Easier to decode (linear in k)

- What is efficiency?

ARQ not suitable

- Many receivers
  - Ack/Nack explosion
  - Sender must track lots of state

- Heterogeneous receivers
  - Sender has to buffer to accomodate the slowest receiver
  - Different types of receivers – loss rates high on low BW channels

- Hard to join/leave

- If delay sensitive, ARQ has large variability in delay

Solution

- Use erasure codes (type of FEC) to redundantly encode k source packets into n packets; n = k + l
  - multicast n packets
  - receiver reconstructs packets from any k received packets
  - Is this better?
Tornado Coding

- Code packets are xor of a small random set of the original packets
  - How the random graph connectivity is chosen is not in this paper; heavy tail distributed

- Some additional code packets can be generated from xor of the first level code packets

- \( y_1 = x_1 \oplus x_2 \oplus x_3 \). If \( x_1, x_2 \) and \( y_1 \) are received, can recover \( x_3 \)

- Tornado name origin from cascading decoding when a missing piece is received

Decoding/Encoding Performance

<table>
<thead>
<tr>
<th></th>
<th>Tornado</th>
<th>Reed-Solomon</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decoding inefficiency</td>
<td>( 1 + \epsilon ) required</td>
<td>1</td>
</tr>
<tr>
<td>Decoding times</td>
<td>((k+O(\ln(1/\epsilon)))P)</td>
<td>(kP)</td>
</tr>
<tr>
<td>Basic operation</td>
<td>XOR</td>
<td>Field operations</td>
</tr>
</tbody>
</table>

Table 1: Properties of Tornado vs. Reed-Solomon codes

- Very expensive to do RS decoding on large files!
- Discuss also an interleaving technique to reduce the cost of RS
- Tornado codes outperform both

Performance

<table>
<thead>
<tr>
<th>Encoding Benchmarks</th>
<th>Tornado Codes</th>
<th>Reed-Solomon Codes</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIZE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>256 MB</td>
<td>4.5 seconds</td>
<td>0.11 seconds</td>
</tr>
<tr>
<td>512 MB</td>
<td>9.1 seconds</td>
<td>0.11 seconds</td>
</tr>
<tr>
<td>1 MB</td>
<td>9.1 seconds</td>
<td>0.29 seconds</td>
</tr>
<tr>
<td>2 MB</td>
<td>44.2 seconds</td>
<td>0.67 seconds</td>
</tr>
<tr>
<td>4 MB</td>
<td>112.4 seconds</td>
<td>1.01 seconds</td>
</tr>
<tr>
<td>8 MB</td>
<td>699.4 seconds</td>
<td>1.69 seconds</td>
</tr>
<tr>
<td>16 MB</td>
<td>3988.2 seconds</td>
<td>3.53 seconds</td>
</tr>
</tbody>
</table>

Table 2: Comparison of encoding times.

<table>
<thead>
<tr>
<th>Decoding Benchmarks</th>
<th>Tornado Codes</th>
<th>Reed-Solomon Codes</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIZE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>256 MB</td>
<td>4.5 seconds</td>
<td>0.11 seconds</td>
</tr>
<tr>
<td>512 MB</td>
<td>9.1 seconds</td>
<td>0.11 seconds</td>
</tr>
<tr>
<td>1 MB</td>
<td>9.1 seconds</td>
<td>0.29 seconds</td>
</tr>
<tr>
<td>2 MB</td>
<td>44.2 seconds</td>
<td>0.67 seconds</td>
</tr>
<tr>
<td>4 MB</td>
<td>112.4 seconds</td>
<td>1.01 seconds</td>
</tr>
<tr>
<td>8 MB</td>
<td>699.4 seconds</td>
<td>1.69 seconds</td>
</tr>
<tr>
<td>16 MB</td>
<td>3988.2 seconds</td>
<td>3.53 seconds</td>
</tr>
</tbody>
</table>

Table 3: Comparison of decoding times.
Features of the Solution

- Compare with a theoretical ARQ based solution
- Low server overhead (overhead is not a function of number of receivers)
- Two way communication not needed
- Tolerates loss well (is it always more efficient than ARQ?)
- Good for network as well

Discussion?

- I skipped some interesting details; anything you want to bring up?
- Digital fountain idea is not the contribution of this paper; this paper argues for using Tornado codes and shows they are more efficient than RS or block interleaved RS

Code Efficiency

- 99%+ less than 8% inefficiency

Interleaved RS

- Reduce cost by doing RS on per block
  - Cost moves towards linear from quadratic
  - Still much slower than Tornado

Features of the Solution

- Low server overhead (overhead is not a function of number of receivers)
- Two way communication not needed
- Tolerates loss well (is it always more efficient than ARQ?)
- Good for network as well

Discussion?

- I skipped some interesting details; anything you want to bring up?
- Digital fountain idea is not the contribution of this paper; this paper argues for using Tornado codes and shows they are more efficient than RS or block interleaved RS
Automatic Repeat ReQuest (ARQ)

- Note:
  - Potential of duplication of packets
  - How to discriminate packet from duplicate?
  - Delayed packets?
  - Efficiency (sliding window size)

Sequence Numbers

- SeqNum cannot grow arbitrarily; will the protocol still work?
- Stop-and-wait is sliding window with window size one; true or false?
  - What was the equivalent of the sequence number there?
  - How big should the sequence number field be in general?
- What if MaxSeqNum is equal to the window size; will this work?
  - For a window size 8, 3 bit seqNum field (0..7)
  - Sender transmits frames 0..6, they arrive successfully but ACK's are lost
  - Sender retransmits 0..6, receiver expecting 7, 0, .. 5; what happens?

Sequence Number (cont’d)

- \( SWS < \frac{MaxSeqNum + 1}{2} \)
  - Intuitively, SeqNum slides between two halves of sequence number space
- What if packets are delayed? could still run into problems?
Dealing with Delayed Packets

- Is this an issue with link layer?
- Never reuse sequence number?
- Require in-order delivery?
- Other?

Place limit on Packet delivery time

- Time To Live: IP packets hop count is decremented at every hop; discard if the packet is too old
- Sequence numbers unique within the time-to-live period
- TCP standard defines a maximum segment lifetime
  - 120 seconds recommended

Medium Access Control (MAC)

- What are the possible ways to organize access to the shared medium?
- FDM? TDM? Traffic is bursty, utilization is poor
- Problem definition
  - N independent stations sharing a single channel
  - If two frames are transmitted simultaneously, they overlap on the channel and both are lost
  - Carrier sense or not?
  - Frames all the same size for now
  - Continuous or discrete time (can the stations agree on slots?)

Multiple receivers (e.g., multicast)

Delay sensitive applications (e.g., real time traffic)

High loss high delay channels (e.g., satellite)

One way channels (e.g., TV)

Digital Fountain paper example of alternative (basically FEC)

When is ARQ not suitable/efficient?
Carri er Sense Not Used – ALOHA

• Due to Abramson’s group at U. Hawaii
• Wireless network used to connect the Hawaiian islands
• Simple algorithm:
  – Transmit when you have data to send
  – Listen at the same time, if you receive what you transmit no collision
  – otherwise backoff and retry
• Random traffic (and a lot of random phenomena) are often modeled as a Poisson distribution
• Probability of generating k frames in a given time frame is:
  \[ P(k \text{ frames generated}) = \frac{G^k e^{-G}}{k!} \]
• What is the probability of successfully transmitting a frame in ALOHA?

ALOHA Success Probability

• Network traffic often modeled as a Poisson distribution – probability of generating k frames in a given time frame is:
  \[ P(k \text{ frames generated}) = \frac{G^k e^{-G}}{k!} \]
• A frame is transmitted correctly only if no other frames are transmitted at the same time
  \[ P(0) = e^{-G} \]
• Average number of frames transmitted successfully in a time frame is average number transmitted (G) times the probability that no other frames are generated (P(0))
  \[ S = G P(0) = G e^{-G} \] — is this the throughput?
  \[ \text{How big is the vulnerable period for a frame?} \]
• Success rate = \( G e^{-2G} \)
  What is the maximum success rate?

Carrier Sense Multiple Access – CSMA

• If you listen before you speak, should result in less confusion
  – 1-persistent: as soon as quiet, start transmitting
  – p-persistent: as soon as quiet, start transmitting with probability p; otherwise, wait for next slot and start transmitting with probability p if idle
  – Non-persistent: wait a random time if busy
• Which is best? How do they compare with slotted Aloha?
  – Consider different load conditions
  – What type is ethernet?
• Can anything else be done to improve performance?
Collision Detection

- Stop speaking as soon as you hear collision CSMA/CD; ethernet is CSMA/CD 1-persistent
- What happens if the propagation delay is long?
- Can collisions be avoided altogether?
- Reservation Protocols

Ethernet Physical Layer

- Some confusion due to Ethernet (industry) standard vs. IEEE standard. Different requirements for different physical cables
  - 5-4-3 is IEEE 802.3 rule; no more than 2 repeaters is “Ethernet” rule
  - 1024 hosts is the maximum number of hosts using 10BaseT or 10BaseF cables. These are star topologies (through hubs) and do not have the minimum separation between hosts requirement
- Ethernet FAQ (http://netman.cit.buffalo.edu/FAQs/ethernet.faq)

Ethernet – A Contention Based Protocol

- Developed by Xerox PARC in the mid seventies
- Seminal paper by Metcalf and Boggs – reading (origins), class homepage
- Influenced by the Aloha packet-radio network (important project at university of Hawaii to connect the Hawaiian islands)
- Standardized by Xerox, DEC and Intel in 1978
- Similar standard by IEEE: IEEE 802.3
- The 802 family of standards define link layer operations for shared media
  - Original standard 10Mbps; updated to 100Mbps and 1Gbps
- For “shared” links such as ethernet, need a distributed algorithm that provides fair access to the shared medium
Frame Format

<table>
<thead>
<tr>
<th></th>
<th>Preamble</th>
<th>Dest</th>
<th>Src</th>
<th>Type</th>
<th>Body</th>
<th>CRC</th>
</tr>
</thead>
<tbody>
<tr>
<td>64</td>
<td></td>
<td>48</td>
<td>48</td>
<td>16</td>
<td></td>
<td>32</td>
</tr>
</tbody>
</table>

- Preamble: alternating 0's and 1's for synchronization
  - IEEE 803.2 uses count (in bytes) instead of type field
- Addresses are globally unique 6 bytes placed in firmware by the manufacturer; Example: 5:8b:40:2:cf:4a
- Multicast address starts with 1
- Broadcast address all 1's
  - How does a sender know the receiver's address? (ARP; later)
- Promiscuous mode – listen to all packets

Transmission Algorithm

- If line is idle:
  - Send immediately; upper bound message is 1500 bytes, lower bound is 46 bytes (why do we need a lower bound?)
  - If a collision is detected (by one of the senders), it transmits a jamming sequence (32-bits)
  - Must wait 51.2 µ-sec between back to back frames; why?
- If line is busy
  - Wait until idle and transmit immediately
- This is an example of a medium access algorithm (what we called a fair distributed algorithm for sharing the medium)

**Collisions**

(a)

(b)

(c)

(d)

- What is the maximum time that a node has to wait before it detects a collision?
- What happens on a collision?
  - Node waits up to 51.2 µ-sec, senses the line, then transmits if idle
  - If another collision, node waits up to 102.4 µ-sec, then tries again
  - Another collision? Wait up to 204.8 µ-sec
  - **Exponential back off**; gives up after 16 retries, timeout capped at 10

Contention Discussion

- Contention based protocols are good under some circumstances
  - Low probability of collisions (low load)
  - Low cost of collision (low ?)
  - Contention can be bad
  - Too much time wasted contending for the medium
  - Unpredictable performance
  - Role of backoff
Reservation Based Protocols

- Try to eliminate collisions completely
- Reservation based protocols - collaborate on reserving the use of the medium
  - Advantage: eliminate collisions, more predictable performance
  - Disadvantage: reservation overhead even when not necessary, can be high
- Characteristics of the network: most notably the delay (τ) affects the choice of the MAC protocol
- Characteristics of the traffic also affect the choice

Satellite Networks

- Ground station transmits via uplink to satellite; satellite broadcasts down (downlink) to all stations
- Different types of satellite “constellations”
  - Geostationary Earth Orbit (GEOs) are very high (> 22,000 miles). Fixed orbit relative to earth
  - Low Earth Orbit (LEOs) are low orbit (few hundred miles), but move relative to earth. Latency much lower
  - MEO’s in the middle
- “Long fat pipes”: delay is a few hundred m-sec (270 msec typically used); bandwidth can be high

Satellite Networks

- Carrier sense not feasible:
  - Can’t sense uplink, it is a point to point connection
  - Sensing downlink results in sensing what was transmitted 270 msec ago
  - For same reason collision detection not useful (what is the equivalent of ethternets 46 bytes minimum packet length?)
- What are our options? Given that CSMA and CD dont work:
  - Either Use ALOHA
  - Or, need some reservation based protocol
  - Claim: Static allocation (FDMA, TDMA, CDMA) methods are ultimate reservation protocol