ZIGZAG : An Efficient Peer-to-Peer Scheme for Media Streaming

Presented by Chi-Hung Chao

INFOCOM 2003
Outline

- Introduction
- Proposed Solution
- Performance Evaluation
- Conclusions
Introduction

- Problem:
  Streaming live media from a single source to a large quantity of receivers on the Internet
- IP Multicast could be the best way to overcome this drawback, but......
Introduction

- Chaining
  - Building a delivery tree which is rooted at the source and including all receivers.
  - Receivers get the content from the source or from the other receivers.
Introduction

- Important issues:
  - End-to-end delay
    - Small tree height
    - Bounded node degree
  - Failure recovery
  - Control overhead at each receiver should be small
ZIGZAG:
- Address all of the previous issues
  - Organizing receivers into a hierarchy of bounded-size cluster
  - Failure recovery can be done regionally with only impact on a constant number of existing receivers and no burden on the source.
Outline

- Introduction
- **Proposed Solution**
- Performance Evaluation
- Conclusions
Proposed Solution

- Administrative Organization
- Multicast Tree
- Control Protocol
- Client Join
- Client Departure
Proposed Solution - Administrative Organization

Fig. 1. Administrative organization of peers
Proposed Solution-
Administrative Organization

- Layer 0 contains all peers
- Peers in layer \( j<H-1 \) are partitioned into clusters of sizes in \([k,3k]\). Layer H-1 has only one cluster which has a size in \([2,3k]\)
- Cluster head in layer \( j \) becomes a member of layer \( j+1 \) if \( j<H-1 \)
Proposed Solution - Administrative Organization

- Subordinate
- Foreign head
- Foreign Subordinate
- Foreign cluster
Fig. 2. The multicast tree of peers ($H = 3$, $k = 4$)
Proposed Solution - Multicast Tree

- A peer, when not at its highest layer, cannot have any link to or from any other peer.
- A peer, when at its highest layer, can only link to its foreign subordinate. (besides server)
- Non-head members get the content directly from a foreign head.
Proposed Solution - Multicast Tree

- **Theorem 1**: The worst-case node degree of multicast tree is $O(k^2)$

  Proof: A node has at most $(3k-1)$ foreign cluster, thus having at most $(3k-1)(3k-1)$ foreign subordinates. Therefore the server degree is at most $(3k-1)(3k-1)+(3k-1)=9k^2-3k$. Theorem 1 has been proved.

- **Theorem 2**: The height of the multicast tree is $O(\log_k N)$
Proposed Solution: Multicast Tree

- A peer gets the content from a foreign head, but not its head, and can only forward the content to its foreign subordinate, but not its subordinate.

- Suppose the members of a cluster always get the content from their head. A node would have larger degree if it is closer to the source.
Proposed Solution - Multicast Tree

- Using a foreign head as the parent has another nice property.
  - When the parent peer fails, the head of its children is still working, thus helping reconnect the children to a new parent quickly and easily.
Proposed Solution - Control Protocol

To maintain its position and connections in the multicast tree and the administrative organization, each node X periodically communicates with its clustermates, children and parent on the multicast tree.
Proposed Solution -

Client Join

- Some functions in the algorithm
  - \text{Reachable}(X)
  - \text{Addable}(X)
  - \text{D}(X)
  - \text{D}(X,Y)
Proposed Solution - Client Join

1. If $X$ is a leaf
2. Add $P$ to the only cluster of $X$
3. Make $P$ a new child of the parent of $X$
4. Else
5. If $Addable(X)$
6. Select a child $Y$:
   \[ Addable(Y) \text{ and } D(Y)+d(Y, P) \text{ is min} \]
7. Forward the join request to $Y$
8. Else
9. Select a child $Y$:
   \[ Reachable(Y) \text{ and } D(Y)+d(Y, P) \text{ is min} \]
10. Forward the join request to $Y$
Proposed Solution - Client Join

Theorem: The join overhead is $O(\log_k N)$ in terms of number of nodes to contact.

Proof:

height = $O(\log_k N)$

degree = $O(k^2)$

overhead = $O(k^2 \log_k N) = O(\log_k N)$
Proposed Solution - Client Join

- If the new size of the joined cluster is over 3k, the cluster has to be split so that the newly created clusters must have sizes in \([k, 3k]\)
Proposed Solution - Client Join

(a) Before Splitting  (b) After Splitting

Ex-children of Y

A set of links  A single link to a child

Split Algorithm
Proposed Solution - Client Departure

(a) Before Failure

(b) After Recovery
Outline

- Introduction
- Proposed Solution
- Performance Evaluation
- Conclusions
Performance Evaluation

- N=2000 , k=5
- 3 scenarios
  - Failure free
  - Failure possible
  - Comparing ZIGZAG to NICE
Performance Evaluation
Failure Free

ZIGZAG (avg=47.99, max=115)
Performance Evaluation
Failure Free

ZIGZAG (avg=5.13, max=136, #splits=221)
Performance Evaluation
Failure Free

ZIGZAG (max=22, std-deviation=3.1)
Performance Evaluation

Failure Free

ZIGZAG (avg=12.5745, max=48, std-deviation=6.71)
Performance Evaluation

Failure Possible

ZIGZAG (avg=0.96, max=31, std-deviation=3.49)

1800 clients remained

Failure Overhead

Failure ID
Performance Evaluation
Failure Possible

ZIGZAG (avg=11.16, max=17, std-deviation=3.16, #merge=62)
Performance Evaluation
ZIGZAG vs NICE

![Graph showing performance evaluation of ZIGZAG vs NICE]

- ZIGZAG
- NICE

Avg. Recovery Overhead vs Failure probability p
Outline

- Introduction
- Proposed Solution
- Performance Evaluation
- Conclusions
Conclusions

- The key in ZIGZAG’s design is the use of a foreign head to forward the content.

- 4 properties
  - Short end-to-end delay
  - Low control overhead
  - Efficient join and failure recovery
  - Low maintenance overhead