Outline

- P2P video systems
  - What and why
- Video management
  - Caching Collaboration
  - Cache Allocation
- Performance Evaluation
- Conclusions
P2P Video Systems

**Definition**
- A P2P video system is a logical system consisting of a number of hosts collaborating for the purpose of video services.

**Features**
- Resource sharing
  - A host can be both a service requester and a service provider.
- There is no central server and no host guarantees the existence of any video
  - A host can introduce a video to the system and then delete it quickly.
- The participation is fully volunteer
  - A seeding host might quit the service at a later time.
Why this becomes a problem

- Videos are large
  - Statistics show that a majority of clients contribute less than 1 GB of disk space

- Streaming out of a video takes a substantial communication bandwidth
  - This seems to be more concerned by users

- If we allow a client to cache a video in its all or not at all, we may end up only a few clients can contribute
  - This will again create the problem of server bottleneck
Challenges of Caching Partial Videos

- **Caching collaboration**
  - Given a fixed amount of disk space, what part of a video should a client cache?
  - How can we find a set of video segments that are cached by many different hosts?

- **Cache allocation**
  - Should a client cache a newly downloaded video?
  - If yes, how much cache space should be allocated for this video?
  - If the cache is full, what existing data should be expunged to make room?
Caching Collaboration: Cell

Cell concept

- A cell is a cluster of hosts that together can supply a complete video
- Each host in a cell caches a number of video segments
- A CacheTable is used to track who caches what
Cell Advantages

- **Reducing cache redundancy**
  - Caching and deletion of file segments can be coordinated at cell level
  - The cost of such caching collaboration can be minimal since each cell typically contain only a small number of member
    - A cell is split whenever possible

- **Reducing search cost**
  - Looking for a set of video segments now becomes looking for any single segment
    - Search scope is dramatically reduced: determined by the distance to the *nearest* caching host instead of the *farthest* caching host
Cache Operation

Cache procedure

- Retrieve CacheTable from a cell that provides service
- Cache the least redundant segments in CacheTable
- Modify CacheTable and update other cell members
Split Operation

Why split

- Keep the cell size small and reduce maintenance overhead

Exhaustive search is impractical

- To split $k$ hosts into two separate groups, there are $O(2^k)$ different combinations

Proposed split algorithm

- Can split a cell whenever possible
- Takes only $O(k^2)$ computation
Split Operation

Split procedure
- Put all members in OldSet and create an empty NewSet
- Find a host in OldSet and move it to NewSet which satisfies two conditions:
  - Without this host OldSet can still supply a complete file
  - This host can provide maximum number of segments that are currently not in NewSet
- Repeat this procedure until no such host can be found
- If NewSet can supply a complete file then the original cell can be split into two cells
Split Example
Delete Operation

- Deletion happens when a host needs to recycle its cache space
- Delete procedure
  - Delete most redundant segments in CacheTable
  - Modify CacheTable and update other cell members
Merge Operation

- Merge happens when a cell is broken
  - Cell members delete non-redundant segments
  - Cell members found off-line

- Merge procedure
  - Find a list of cells
  - For each member in the broken cell, select a cell from the list and add it into this cell
  - Split the resulting cell if possible

- Select procedure
  - A host should join a cell whose least redundant segments can be maximally reinforced
Implementation Issues

- CacheTable consistency
  - Using mutual exclusion mechanism such as majority quorum algorithm to avoid concurrent updates on CacheTable
  - Using optimistic concurrency control mechanism. When concurrent Delete operations happen and a cell is broken, Merge operation will be invoked to reorganize the cell members

- Cell data redundancy control
  - Increasing the data redundancy level in a cell to avoid frequent Split and Merge operations
Cache Allocation

Given a newly downloaded video, should it be cached?

If yes, three questions remain:

- How much cache space should be allocated for the new file
- Which part of the file should be cached
  - This is solved by Cell technique
- If necessary, what data in the cache should be expunged to make space for the new file
Traditional Schemes

Least Recently Used (LRU) and Least Frequently Used (LFU)
- Keep popular data items in the cache to reduce the service cost of popular items
- Unpopular files might be quickly deleted
- Suitable for systems with a central server but might cause unpopular files to vanish in a decentralized P2P system
Uniform and Proportional Allocation

Uniform allocation
- Maintains equal number of copies in the system for all files
- Protects unpopular files from being extinct but suffers from load balancing problem

Proportional allocation
- Allocates cache space based on the file popularity
- Achieves optimal load balancing but unpopular files might become extinct quickly
Controlled Inverse Proportional (CIP) Cache Allocation

1. Determine if a newly downloaded file should be cached
2. Calculate the amount of cache space to be allocated for the new file
   - Try to allocate equal cache space for all videos
3. If there is no empty space for a new file, some redundant segments of cached videos are deleted
   - Look at the CacheTable of each video
4. If there are no redundant segments, a cached video will be deleted to make room for the new video segments
   - Which segments should be cached is determined by Cell
CIP-H

A heuristic solution
- Compare the popularity of the new video with the popularity of each cached video
- Among all cached videos that have higher popularity than that of the new video, choose the one with the lowest popularity to replace

Advantages
- Files with lower popularity than the new video will be retained
- Popular videos are replaced dynamically and most popular videos are also retained
A solution based on analysis model

- Define Caching Probability of video $i$ as $P_i = \left( \frac{p_{\text{min}}}{p_i} \right)^{\alpha}$
  - $p_{\text{min}}$: the popularity of the least popular video
  - $p_i$: the popularity of video $i$
  - $\alpha$: an adjustable parameter

- By adjusting $\alpha$, the caching effect of CIP-$\alpha$ can vary between uniform and proportional allocation
  - When $\alpha=0$, $P_i=1$, CIP-$\alpha$ becomes proportional allocation
  - When $\alpha=1$, $P_i=p_{\text{min}}/p_i$, CIP-$\alpha$ becomes uniform allocation
Performance Metrics

- **Accessibility**
  - Minimum number of hops that a query message needs to be delivered in order to find a complete file
  - Measures the cost of searching a large file

- **Availability**
  - Number of distinct copies of the file in the whole system
  - Measures the cache space allocated for a file in the system
Schemes for Comparison

- **Caching collaboration**
  - Cell vs. Random
    - Random: a host randomly chooses segments to cache after downloading a video

- **Cache allocation**
  - CIP-h vs. CIP-a vs. Uniform vs. Proportional
    - Uniform: all videos are allocated equal cache space in the whole system
    - Proportional: the total cache space allocated to a video in the whole system is proportional to its popularity
Simulation Setup

- Simulated in a Gnutella-like P2P network of 5000 hosts with the average distance between two hosts of 7 hops.
- The video popularity follows Zipf-like distribution with skew set to 1.
- The cache space among hosts also follows Zipf-like distribution with skew set to 1.
- 100 videos are shared. In the beginning each file has one seeding host. Then hosts begin to download files and cache files. When the network caching capacity has been sufficiently utilized we collect the performance data.
Effect of Cell on Video Accessibility

- Results in Cell and Random show some similarities
- The file accessibility in Cell is much better than that in Random
Effect of CIP on Video Availability

- CIP provides balance between Uniform and Proportional
- CIP-α provides more fine tuned control over the file availability
Effect of Cell/CIP on Video Accessibility

In all cases the accessibility in Cell/CIP is better than that in Random/Uniform and Random/Proportional.
Effect of Cell/CIP on Video Availability

- The curves of Cell/CIP fall between Random/Uniform and Random/Proportional.
- Adjusting $\alpha$ can change CIP-$\alpha$ between Uniform and Proportional.
Conclusions

- **Caching collaboration**
  - Cell caching can effectively balance the data redundancy and reduce the search cost for large files

- **Cache allocation**
  - CIP cache allocation can achieve loading balancing and maintaining unpopular files in the system