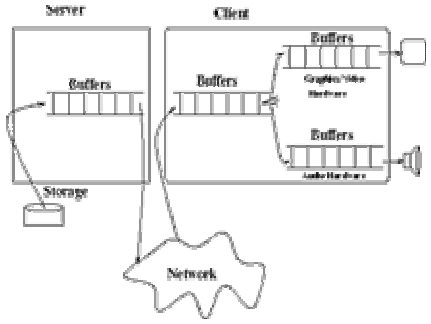
CS323 Operating Systems Multimedia III

Yuanyuan Zhou
Lecture 31
4/14/2003

Content of this lecture

- Reminder:
  - Midterm 4/21, no class in the morning
  - Conflict exam signup: 4/4 noon
    - Exam data: Wed 4/23 5-6pm, room TBA

Data Flow for a Multimedia Server

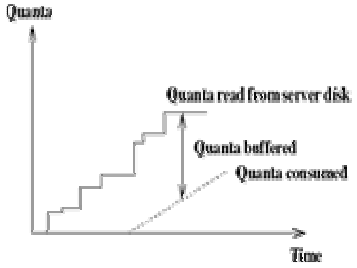


Requirements

- Real-time Storage and Retrieval CM recording devices generate continuous stream of media quanta that must be stored in real time.
- Playback is a reverse operation of recording. The media quanta must be presented using the same timing sequence with which they were captured.
- Media components can be combined in fashion requiring synchronization.
- Multimedia file servers must act like VCRs (start, stop, fast forward, rewind, pause)
- Multimedia file servers need to support streaming and push the data at the user, hence they are called push servers.
- High Data Transfer Rate and Large Storage Space Examples are:
  - HDTV quality 1280x720 pixels/frame 24 bits/pixel yield 81 Mbytes per second
  - NTSC quality 640x48 pixels/frame 24bits/pixels yield 27 MBytes per second

Continuous Media Recording and Playback

- Server's challenge is to ensure continuous retrieval of a media stream from the disk.



Playback

- **Single-stream Playback** Continuous playback can be assured simply by buffering the entire stream before initiating playback. However, this solution is **not efficient**.
- In order to efficiently service a single stream, there are three problems to solve:
  - preventing starvation
  - minimizing the buffer space requirement
  - minimizing the initiation latency
- **Multi-stream Playback** MM server must process retrieval requests for several streams simultaneously. One solution is to dedicate a disk head to each stream and treat each disk head as a single stream. This solution is not efficient. Another solution is to multiplex streams per disk.
- MM server must process multiple retrieval requests for the same stream.
- We need new **multimedia file systems**.

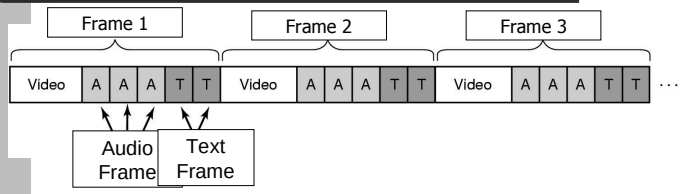
Support of Continuous Media

- Provide a constant and timely retrieval of data. Following approaches are taken:
- Proper management of multimedia disk storage.
  - Special disk scheduling algorithms

Management Multimedia Disk Storage

- Optimally place data blocks on the disk.
- Use multiple disks.
- Add tertiary storage to gain additional capacity.
- Build storage hierarchies.

File Placement



Placing a File on a Single Disk

- Interleaving
  - Video, audio, text in single contiguous file per movie
  - Why?

Placement of Data Blocks of a File on a Single Disk

- **Discussion**
  - **Tradeoff between continuous placement vs. scattered placements**

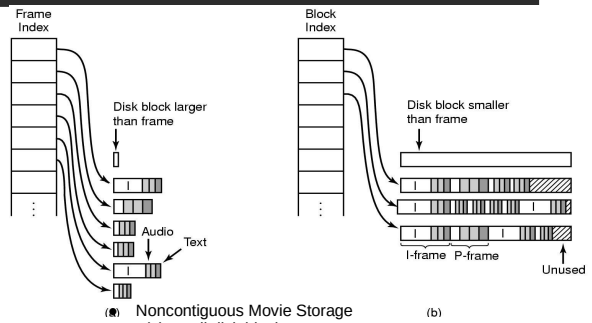
Tradeoff

- **Continuous Placement**
  - **Pros**
    - simple to implement
    - When reading a file, only 1 seek is needed
      - Scattered placements may require a seek for each blockhead---Intrafile seek.
      - Does this advantage still hold for
        - Multi-stream
        - Video editing
  - Cons
    - Fragmentation
    - Copying overhead for insertion and deletion to maintain continuity

Intrafile Seek

- Intrafile seek can be avoided in scattered layout if the amount read from a stream always evenly divides a block.
- Solution to intra-file seek: Select a sufficient large block and read one block in each round.
- If more than one block is required to prevent starvation prior to the next read, intra-file seek is necessary.
- Instead of avoiding intra-file seeks, reduce them to a reasonable bound, which leads to **Constraint Placement Approach**.

Two Alternative File Organization Strategies (1)

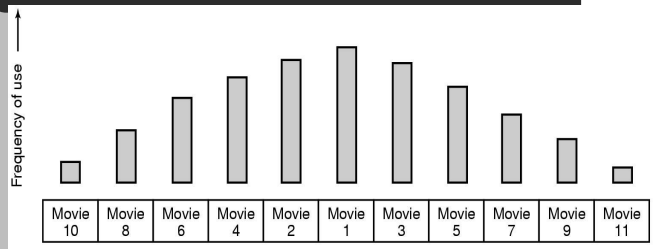


Noncontiguous Movie Storage
(a) small disk blocks
(b) large disk blocks

File Organization Strategies

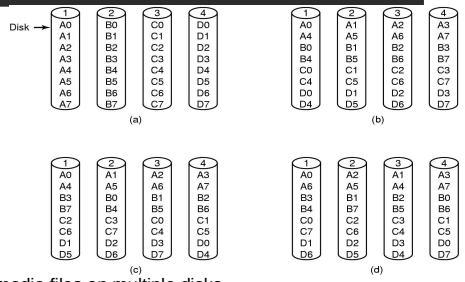
- **Small-block Organization**, called also *constant time length*, uses small disk blocks and organizes all data per frame. This organization uses a *frame index* per movie, where each entry points to the start of the frame. Each frame consists of video, audio, text tracks for that frame.
- **Large-block Organization**, called also *constant data length*, uses large disk blocks, and puts several frames into one block. This organization uses *block index* to point to each block.
- Comparison: Large-block organization suffers from internal fragmentation, small-block organization yields little waste of disk space. On the other hand, large-block organization has smaller index than the small-block organization.

Placing Multiple files on a Single Disk (2)



- Organ-pipe distribution of files on server
  - most popular movie in middle of disk
  - next most popular either on either side, etc.

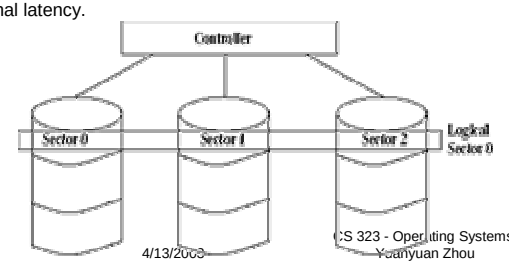
Placing Files on Multiple Disks



Organize multimedia files on multiple disks
(a) No striping
(b) Same striping pattern for all files
(c) Staggered striping
(d) Random striping

Data Striping

- Data striping means to scatter data across multiple disks using RAID (redundant arrays of inexpensive disks).
- The disks are spindle synchronized and operate in lock-step parallel mode.
- Physical and logical blocks have identical access time, and transfer rate increases.
- Striping improves bandwidth requirements, but does not improve seek and rotational latency.

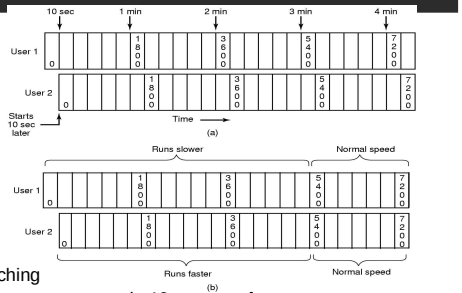


Data Interleaving

- Blocks are interleaved across the disk array with successive file blocks stored on different disks.
- A simple interleaved pattern stores blocks cyclically across an array of N disks.
- Difference to striped data: interleaved data are not spindle synchronized.
- Data interleaving operates independently.

Round k	Disk 1	Disk 2	Disk 3
1	File A, block 1	File B, block 1	File C, block 1
2	File C, block 2	File A, block 2	File B, block 2
3	File B, block 3	File C, block 3	File A, block 3
4	File A, block 4	File B, block 4	File C, block 4

Caching



Block Caching
(a) Two users, same movie 10 sec out of sync
(b) Merging two streams into one

File Caching

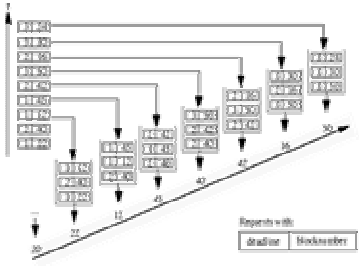
- Most movies stored on DVD or tape
  - copy to disk when needed
  - results in large startup time
  - keep most popular movies on disk
- Can keep first few min. of all movies on disk
  - start movie from this while remainder is fetched

Disk Scheduling Algorithms

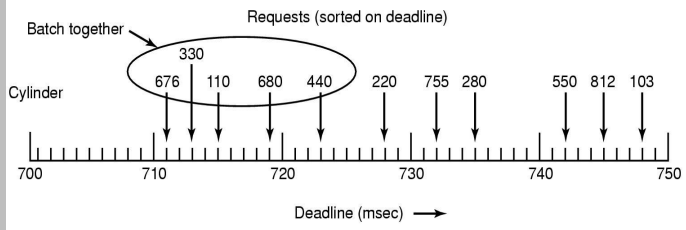
- *Goal of scheduling in traditional file systems*
  - reduce cost of seek time
  - achieve high throughput
  - provide fair disk access;
- *Goal of scheduling in multimedia file systems*:
  - meet deadlines of all time-critical tasks
  - keep the necessary buffer requirements low
  - serve many streams concurrently
  - find balance between time constraints and efficiency.

EDF Scheduling Algorithm

- **Earliest Deadline First**
- Problem: poor throughput and excessive seek time.
  - It is a preemptive scheduling algorithm, hence it has *high overhead*.



Dynamic Disk Scheduling



- Scan-EDF algorithm
  - uses deadlines & cylinder numbers for scheduling