Lecture 27: How does a computer... run many applications simultaneously?

Review:
What is an Operating System?

Operating System (OS):
Software that converts hardware into a useful form for applications
Very complex: millions of lines of code, 1000 person-years

Authors:
Review: What does OS do?

Manage hardware resources for applications
What do you think this entails?

Role #1: Provide standard library for accessing resources
- Allow applications to reuse common facilities
- Make different devices look the same
- Provide higher-level abstractions

Role #2: Coordinate usage of resources (i.e., manager)
- Virtualize resources so multiple users or applications can share
- Protect applications from one another
- Provide efficient and fair access to resources

Review: What are different HW Resources?

CPU: Process scheduler
- Determines when and for long each process executes

Memory: Memory manager
- Determines when and how memory is allocated to processes
- Decides what to do when main memory is full

Disk: File system
- Organizes named collections of data in persistent storage

Network: Networking
- Enables processes to communicate with one another
Review: Challenges with Sharing Memory

2) Not enough physical memory for all address spaces
   - What can OS do if processes need more memory?
   - What policy should OS use to determine what is kept in memory and what is not?

Review: Memory Hierarchy

Leverage memory hierarchy of machine architecture
   - If doesn’t fit in one level, move to next level down
Review: Challenge

What parts of each address space should OS keep in main physical memory vs. disk storage?

Policy Motivation: Inefficient Library

1000 checkouts/returns per day

Distance covered?

• 50ft x 2 x 1000 = 100,000 feet = ~ 20 miles

Please help, she’s worn down and customers are waiting too long!!!

Ideas?
Better Arrangement: Popular Books Nearby

“Most popular” shelf closer by!

Reserves

5 ft

50 ft

How to calculate distance per day?
Need to know how many requests go to bookshelf vs. reserves

Distance covered?

Distance
\[ \text{Distance} = \text{(Number of times walk to reserves} \]
\[ \times \text{ distance to reserves and back)} \]
\[ + \text{(Number of times walk to shelf} \]
\[ \times \text{ distance to shelf and back)} \]

Distance
\[ = \text{Number of requests} \times \]
\[ (\text{(probability book in reserves} \]
\[ \times \text{ distance to reserves and back)} \]
\[ + \text{(probability book on shelf} \]
\[ \times \text{ distance to shelf and back)}) \]

\[ = 1000 \times (P_{\text{miss}} \times 2 \times 50 \text{ ft} + P_{\text{hit}} \times 2 \times 5 \text{ ft}) \]
\[ = 1000 \times (1-P_{\text{hit}}) \times 100 \text{ ft} + P_{\text{hit}} \times 10 \text{ ft} \]

But, what is the probability, \( P_{\text{hit}} \): requests for nearby books?
80-20 “Rule”

Data collection of populations
- Pareto [1906]: 20% of people own 80% of wealth
- Juran [1930’s]: 20% of organization does 80% of work

General:
- Resources are not evenly distributed across population
- Small percentage of items tend to be disproportionately popular, lucky

Some books, movies, songs more popular than average
- Assume: 20% of books account for 80% requests

Can we now calculate distance?

Distance with Popular Objects Nearby?

<table>
<thead>
<tr>
<th>Reserves</th>
<th>Hold 80% of collection</th>
<th>Handles 20% of requests</th>
</tr>
</thead>
<tbody>
<tr>
<td>“Most popular” shelf</td>
<td>Hold 20% of collection</td>
<td>Handles 80% of requests</td>
</tr>
</tbody>
</table>

Distance covered per day?
\[= 1000 \times [(1-P_{\text{hit}}) \times 100 \text{ ft} + P_{\text{hit}} \times 10\text{ ft}]\]
\[= 1000 \times [.20 \times 100 + .80 \times 10] = 28,000 \text{ ft instead of 100,000}\]
**How can we improve?**

Reserves
- Hold 80% of collection
- Handles 20% of requests

"Most popular" shelf
- Hold 20% of collection
- Handles 80% of requests

**Even better arrangement?**

Reserves
- "Most popular" shelf:
  - 20% most popular books

"Most popular" shelf: Top 4%
- 5 ft
Computer Librarian arrangement

Reserves
Disk

“Most popular” shelf:
20% most popular books
RAM (Memory)

Multiple levels of hardware caching
Level 1: Closest to CPU
Level 2: Next closest...

Goal: Give illusion of storage that
Capacity of largest storage module
Speed of fastest storage module

Remaining Problem:
What to Cache???

How to select what should be placed in cache?

How would you predict most popular books?
  • Use past history (e.g., last month) to determine popularity today

When won’t that always work perfectly?
  • New book arrives; don’t know yet its popularity
  • Popularity of book changes (appears on talk show)
Problem Formulation

Fixed size “cache” (memory)
- Example: Can hold 3 items
  
Stream of requests from app (reads+writes)
- Example:

  A B C D E F A B E F B A E D C D C A

Request must be in cache to access

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
</table>

Request for A:
- Hits in cache

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
</table>

Request for D:
- Misses in cache
  - Must replace existing item
  - Bring item D in from disk

Replacement Algorithm: Which item to replace?

What is Goal of Replacement Algorithm?

Goal?
- Maximize number of times requests hit in cache
  - Same: Maximize probability of cache hit
  - Same: Minimize number of times miss

Optimal algorithm: Maximizes hit rate

Oracle
- Assumes perfect knowledge of future requests!
- Impossible in practice, but good comparison point

Which item should Oracle replace?
- Item not accessed for longest time in future...
  - (no use keeping it in cache)
Oracle Behavior

A B C D E A B E F B F A E D C D C A

A B C

How to Implement Without Oracle???

Use rule of thumb: Assume past predicts future
  • Similar to predicting job length in CPU scheduling!
Look backwards instead of forward
  • Replace item “Least Recently Used”
  • LRU replacement policy
A B C D E A B E F B F A E D C D C A

A B C
Details: Implement LRU Algorithm?

OS: Write software to figure out LRU

Two natural implementations...

1) Track access timestamp of each item in cache
   • On access:
     - update timestamp
   • When do replacement:
     - search for one with oldest time

2) Keep items in cache in ordered list
   • On access:
     - move item to front of list
   • Replacement:
     - pick one at end of list

What are different HW Resources?

CPU: Process scheduler
   • Determines when and for long each process executes

Memory: Memory manager
   • Determines when and how memory is allocated to processes
   • Decides what to do when main memory is full

Disk: File system
   • Organizes named collections of data in persistent storage

Network: Networking
   • Enables processes to communicate with one another
Motivation: 
I/O is Important

Applications have two essential components:
- Computation or Processing
- Input/Output (I/O)
  - What applications have no input?
  - What applications have no output?

Interesting I/O
- Persistent, long-term storage of data
- People want to save and later access their data!

Operating System helps manage persistent (non-volatile) storage
- Remembers state across reboots and power failures
  - RAM, caches, registers lose old values when lose power

Examples of Non-Volatile Storage

- Hard disk drive
- Tapes
- RAIDs
- Flash drives
- NVRAM
Disk Terminology

- spindle
- platter
- surface
- track
- cylinder
- sector
- read/write head

Blocks: Group of consecutive sectors

Disk in Action
Disk Performance

How long to read or write \( n \) sectors?
- Positioning time + Transfer time (\( n \))
- Positioning time: Seek time + Rotational Delay
  - Seek: Time to position head over destination cylinder
  - Rotation: Wait for sector to rotate underneath head
- Transfer time: \( n / (\text{RPM} \times \text{bytes/track}) \)

Typical access time?
- Order of milliseconds (RAM? Similar for all items; Nanoseconds!)

Logical view of disk by OS: Linear array (list) of blocks

Role of File System

Role #1: Standard library
- FS exports higher-level abstractions and functions
- Provides file and directory abstractions (instead of blocks)
- Provides operations like \texttt{mkdir}, create, read, write

Role #2: Coordinate resources
- FS allocates blocks of disk to different files
- User processes do not have direct access to devices; Why
  - Could crash entire system
  - Could read/write data without appropriate permissions
  - Could hog device unfairly
Abstraction: File

User view: Named collection of bytes
- Data (content) + meta-data
- Ex: photos, music, text, executables

Permanently and conveniently available

Meta-data: Additional info associated with file
- Examples? What do users want to know about a file?
  - File size
  - Type of file (used by what applications?)
  - Times: Creation, access, modification
  - Owner and Protection bits (who is allowed to read or write?)
Abstraction: File

Operating system view
- Map collection of bytes to blocks on physical storage device

How?
- Table to translate "file name" and offset within file to blocks X, Y, Z

Mapping File Data to Disk Blocks

Example: Three files (A, B, C, D)
- Corresponding data scattered on disk

| D | D | A | A | A | D | B | B | B | C | C | B | B | D | B | D |

What must OS remember?

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Mapping File Data to Disk Blocks

Example: Three files (A, B, C, D)
  - Corresponding data scattered on disk

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

What must OS remember?

Mapping File Data to Disk Blocks

Example: Three files (A, B, C, D)
  - Corresponding data scattered on disk

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>7</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>8</td>
<td>12</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>9</td>
<td>13</td>
<td>6</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>10</td>
<td></td>
<td>16</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>14</td>
<td></td>
<td>18</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>17</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Creating New Files

Scenario: User creates new file E of 5 blocks?

Where should FS allocate blocks for E?
- Achieve best performance if blocks consecutive
- Disk doesn’t need to wait for seek or rotations
- Avoid disk fragmentation: file blocks scattered throughout disk

Update mapping table to include file E

Today’s Summary

Operating System: Software that manages hardware
- Caching policies for Memory
  - Speed of fastest memory; Capacity largest memory
  - Optimal Replacement Algorithm requires knowledge of future
  - Use past to predict future (Least-Recently-Used)
- File System provides persistent storage of data
  - OS: Better abstraction to users than blocks on disk
    - files and directories
  - Implementation
    - OS tracks location of each file’s content on disk
    - OS uses directories to find mapping table for each file

Announcements
- HW 7 due today
- Project 2: Due 1 week