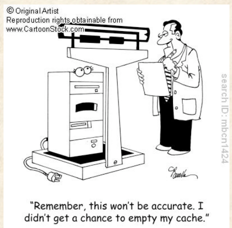
UNIVERSITY of WISCONSIN-MADISON
Computer Sciences Department

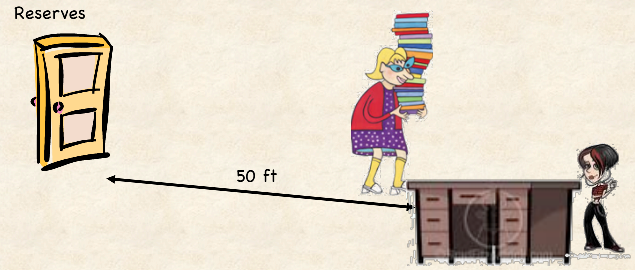
CS 202
Introduction to Computation

Professor Andrea Arpaci-Dusseau
Fall 2010

Lecture 27: How does a computer… access lots of data quickly?



Inefficient Library



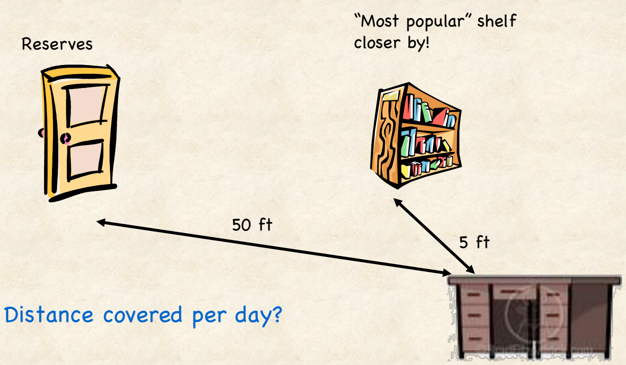
Distance covered?

- 1000 checkouts/returns per day
- 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



Distance covered per day?

Distance covered?

Distance
  = (Number of times walk to reserves
      * distance to reserves and back)
  + (Number of times walk to shelf
      * distance to shelf and back)

  = Number of requests *
    (probability book in reserves
      * distance to reserves and back
    + probability book on shelf
      * distance to shelf and back)

= 1000 * (P_miss * 2*50 ft + P_hit * 2*5 ft)
= 1000 * (1-P_hit) * 100 ft + P_hit * 10ft

But, what is the probability, P_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

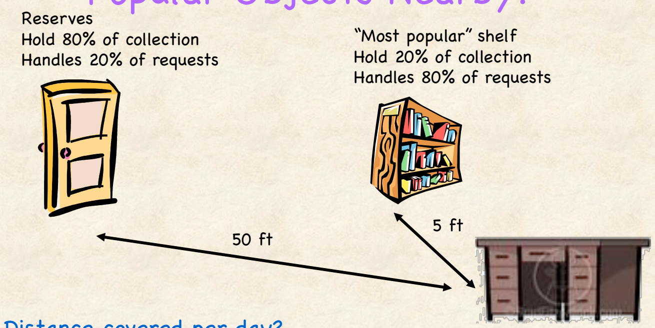
What can we assume about book popularity?

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?

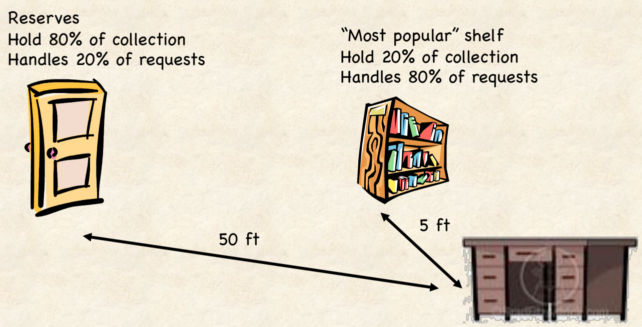


Distance covered per day?

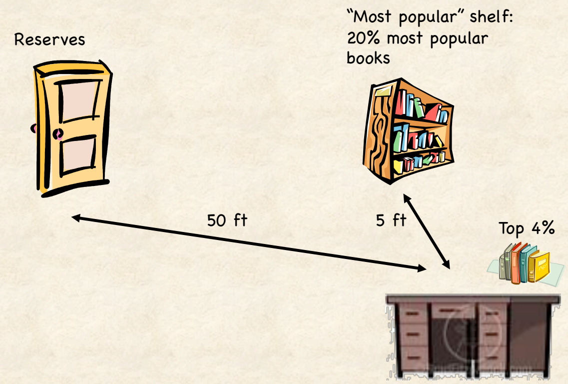
= 1000 * [(1-P_hit) * 100 ft + P_hit * 10ft]

= 1000 * [.20 * 100 + .80 * 10] = 28,000 ft instead of 100,000

How can we improve?



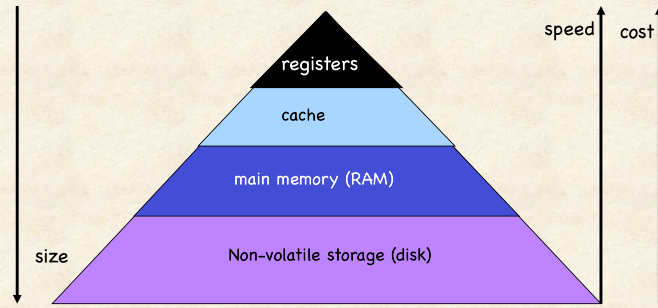
Even better arrangement?



Relationship with Computers?

Memory Hierarchy

Leverage memory hierarchy of machine architecture

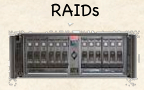


Examples of Non-Volatile Storage

Hard disk drive

Tapes

RAIDs

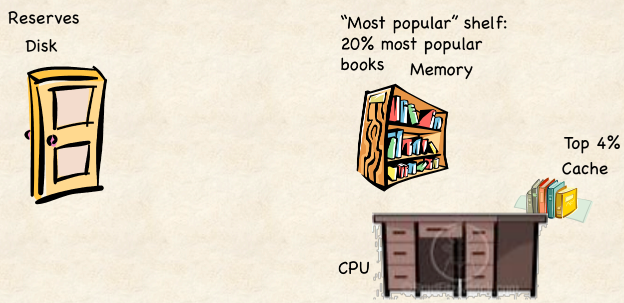


Flash drives

NVRAM

Disk in Action

Computer ~~Librarian~~ arrangement



Often, today's computers have even more levels of caching
Level 1 : Closest to CPU
Level 2: Next closest...

Similar Tasks across Layers of Memory Hierarchy

Hardware cache

HW keeps unreferenced words in RAM

HW moves word to cache when accessed by process

Hardware gives illusion of memory:

- Capacity of (large) RAM
- Speed of (fast) cache

Virtual Memory

OS keeps unreferenced pages on disk

OS moves page to RAM when accessed by process

OS and hardware cooperate to give illusion of storage:

- large as disk
- fast as main memory

Process can run when not all pages fit in RAM

Remaining Problem: What to Cache???

How to select what should be placed in cache?

Librarian: 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

- Example: Can hold 3 items | A | B | C |

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 F

Request must be in cache to access

| A | B | C | Request for A: Hits in cache

| A | B | C | Request for D: Misses in cache

| A | B | D | Must replace existing item

Replacement Algorithm: Which item to replace?

What is Goal of Replacement Algorithm?

Goal?

- Maximize number of times requests hit in cache (probability of cache hit)
- Same: Minimize number of times miss (pay penalty)

Optimal algorithm: Maximizes hit rate

- Very difficult to do this!

Oracle

- Assumes perfect knowledge of future requests!

Which item should Oracle replace?

- Item not accessed for longest time… (no use to keep it in cache)

Oracle Behavior

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

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

6 Misses + Initial

How without knowing the future???

Use rule of thumb

- Assume past predicts the future

Look backwards instead of forward

- Replace item "least recently used"
- LRU replacement policy

Works fairly well in practice

- Application likely to access same variables over and over

Similar to using past behavior of CPU bursts for scheduling

LRU Behavior

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

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

LRU: 10 Misses + Initial

OPT: 6 Misses + Initial

OPT will also do at least as well as LRU
- OPT may do better than LRU or may do same
- OPT cannot do worse than LRU

Implement LRU Algorithm?

Imagine you implement LRU for Operating System

How to determine which page should be replaced from RAM? How to know which is LRU item?

Implementation in Scratch

- You have a List:
  - Cache: Contains identifier for item (A, B, C, D, etc)
- You are notified when item x (A, B, C, D, etc) is Used
- You are notified when must make a replacement

What will you do when item is Used?

What will you do when must make replacement?

LRU Implementation 1

Track access timestamp of each item in cache

- Additional List: Timestamp
  - Elements in same order as Cache list
- On Use:
  - Update corresponding timestamp
- When do replacement:
  - Search for item with oldest time
  - Remove item (from both lists!)

LRU Implementation 2

Keep items in cache in ordered list

- LRU item at end; MRU item at beginning
- On access:
  - Find item in list
  - Move item to front of list
- Replacement:
  - Pick one at end of list

Today's Summary

Caching

- Goal:
  - Speed is close to that of fastest memory (cache)
  - Overall capacity is that of largest memory (disk)
- Optimal Replacement Algorithm requires knowledge of future
- Practice: Use past to predict future (Least-Recently-Used)

Announcements

- Homework 6 Due Today
  - (Accept late hw til 5 Monday; 2 pt penalty)
- Homework 7 available soon… (No programming!)