COT 6936: Topics in Algorithms

Giri Narasimhan

ECS 254A / EC 2443; Phone: x3748

giri@cs.fiu.edu

http://www.cs.fiu.edu/~giri/teach/COT6936_S12.html

https://moodle.cis.fiu.edu/v2.1/course/view.php?id=**174**

Bloom Filters

- Used to test set membership by using bit arrays to indicate which positions have been hashed to.

0	0	1	1	0	1	0	0	1

- Use k hash functions instead of 1.
- How large should k be for given error bound?

Power of Two (or d) Choices

- Each ball comes with $d = 2$ possible bins, each chosen independently at random
- Ball is placed in the least full bin among the d choices
  - ties broken arbitrarily
- MAGICALLY, with high prob:
  - MAX LOAD = $\ln \ln n / \ln 2 + O(1)$
  - Down from $\Theta(\ln n / \ln \ln n)$ (when $d = 1$)
  - In general, when $d \geq 2$,
    - MAX LOAD = $\ln \ln n / \ln d + \Theta(1)$

Applications

- Hashing with 2-way chaining
  - 2 hash function applied to each data item
  - Item inserted in shorter of two chains
- Dynamic Resource Allocation
  - Choosing a server among servers in a network
  - Choosing a disk to store an entity
  - Choosing a printer to serve a print job

COT 6936: Topics in Algorithms

Online Algorithms

Online Problems

- Should I buy a car/skis/camping gear or rent them when needed?
- Should I buy Google stocks today or sell them or hold on to them?
- Should I work on my homework in Algorithms or my homework in OS or on my research?
- Decisions have to be made based on past and current request/task

How to Analyze Online Algorithms?

- Competitive analysis
  - Compare with optimal offline algorithm (OPT)
- Algorithm A is α-competitive if there exists constants b such that for every sequence of inputs σ:
  - $\text{cost}_A(\sigma) \leq \alpha\,\text{cost}_{OPT}(\sigma) + b$

Ski Rental Problem

- Should Dr. Raju buy skis or rent them?
  - Rental is \$A per trip
  - Purchase costs \$B
- Idea:
  - Rent for m trips, where
    - m = B/A
  - Then purchase skis
- Analysis:
  - Competitiveness ratio = 2. Why?

Paging Problem

- Given 2-level storage system
  - Limited Faster Memory (k pages) "CACHE"
  - Unlimited Slower Memory
- Input: Sequence of page requests

Infinite,
Online

- Assumption: "Lazy" response (Demand Paging)
  - If page is in CACHE, no changes to contents
  - If page is not in CACHE, make place for it in CACHE by replacing an existing page
- Need: A "page replacement" algorithm

Well-known Page Replacement Algorithms

- LRU: evict page whose most recent access was earliest among all pages
- FIFO: evict page brought in earliest
- LIFO: evict page brought in most recently
- LFU: evict page least frequently used

Comparing online algorithms?

Game between Cruel Adversary and your algorithm

- Analyze: time? performance?
  - Input length?
  - Performance depends on request s ence
    - Probabilistic models? Markov Decisio rocess
- Competitive analysis [Sleator an Tarjan]
  - Compare with optimal offline algor thm (OPT)
    - OPT is clairvoyant; no prob assumptions; "worst-case"
- Algorithm A is α-competitive if there exists constants b such that for every σ:
  - $\text{cost}_A(\sigma) \leq \alpha\,\text{cost}_{OPT}(\sigma) + b$

Optimal Algorithm for Paging

- MIN (Longest Forward Distance): Evict the page whose next access is latest.
- Cost: # of page faults
- Competitive Analysis: Compare
  - # of page faults of algorithm *A* with
  - # of page faults of algorithm MIN
- We want to compute the competitiveness of LRU, LIFO, FIFO, LFU, etc.

Lower Bound for any algorithm

- Cannot achieve better than k-competitive!
  - No algorithm is α-competitive for α< k
    - Fix algorithm A,
    - Construct a request sequence σ, and
    - Show that: $cost_A(\sigma) \geq k\ cost_{OPT}(\sigma)$
- Sequence σ will only have k+1 possible pages
  - make 1..k+1 the first k+1 requests
  - make next request as the page evicted by algorithm A
    - A will fault on every request
    - OPT? Will fault every k requests

Adversary Model

Upper Bound: LRU is k-Competitive

- Observation: if any subseq has k+1 distinct pages, MIN (any alg) faults at least once
  - Let p be 1st request; p is in CACHE (LRU & MIN)
  - Let T be any subsequence of σ with exactly k faults for LRU & with p accessed just before T.
  - LRU cannot fault on same page twice within T
    - Otherwise it will have faults on k+1 different pages
  - LRU cannot fault on p within T
    - Otherwise it will have faults on k+1 different pages
  - Thus, p followed by T requests k+1 distinct pages and MIN must fault at least once in T

LRU is k-competitive

- Partition σ into subsequences as follows:
  - Let s_0 include the first request, p, and the first k faults for LRU
  - Let s_i include subsequence after s_{i-1} with the next k faults for LRU
  - Argument applies for $T = s_i$, for every $i > 0$
  - If both algorithms start with empty CACHE or identical CACHE, then it applies to $i = 0$ also
  - Otherwise, LRU incurs k extra faults
- Thus, $cost_A(\sigma) \leq k\ cost_{OPT}(\sigma) + k$

Other Page Replacement Algorithms

- FIFO is k-competitive (Homework!)
- MFU and LIFO?

How to Analyze Online Algorithms?

- Competitive analysis
  - Compare with optimal offline algorithm (OPT)
- Algorithm A is α-competitive if there exists constants b such that for every sequence of inputs σ:
  - $cost_A(\sigma) \le \alpha cost_{OPT}(\sigma) + b$

How to Analyze Rand Online Algorithms?

- Algorithm A is *α-competitive* if there exists constants b such that for every sequence of inputs σ:
  - $cost_A(\sigma) \le \alpha cost_{OPT}(\sigma) + b$
- Randomized Algorithm R is *α-competitive* if there exists constants b such that for every sequence of inputs σ:
  - $E[cost_R(\sigma)] \le \alpha cost_{OPT}(\sigma) + b$

Adversary provides request sequence at start