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_S10.html
https://online.cis.fiu.edu/portal/course/view.php?id=427

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:
  - $cost_A(\sigma) \leq \alpha \, cost_{OPT}(\sigma) + b$

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?

How to Analyze Rand Online Algorithms?

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

Adversary provides request sequence at start

Randomized Online algorithms

- Lower bound does not apply to randomized algorithms
  - Lower bound on randomized algorithms = H_k
- Proof uses 2 main principles
  - Cover time of a random walk on K_{k+1} is kH_k
  - Lower bound on competitiveness of randomized algorithms equals competitiveness of best deterministic algorithm A on "worst-case" distribution on request sequence
- H_k is k-th Harmonic number and < ln(k) + 1

Randomized Algorithm: RANDOM

- On a miss:
  - Evict an item chosen uniformly at random from all k items
- RANDOM is k-competitive

Randomized Marker Algorithm

- Algorithm proceeds in rounds
- Each of k pages has a marker bit
  - Start of round: each marker bit = 0 (unmarked)
  - If request is a hit: marker bit = 1 (marked)
  - If request is a miss: replace unmarked location chosen uniformly at random and mark it
  - If all pages are marked: start next round by unmarking all locations
- Marker algorithm is $2H_k$-competitive

k-Server Problem

- Problem: to efficiently "move" around k servers in a metric space (weighted graph) to service requests that appear online at the points of metric space

General Paradigm: k-Server Problem

- Given:
  - metric space (i.e., weighted graph),
  - k servers with initial location, and
  - (online) request sequence with location
    - Request to be served by server at given location
- Goal: minimize distance travelled by servers
- Variants: symmetric or asymmetric

Mobile

k-Server Problem: Applications

- Paging
  - node ≈ page of address space
  - All distances = 1
- Weighted Caching
  - Fonts in a printer or a bitmap display
- Two-headed Disk Drives

What we know: k-Server Problem

- Lower Bound on competitiveness (k) applies from before
- Conjecture: Upper bound for competitiveness is k [MMS, 1990]

Greedy Algorithm

- Let the nearest server serve the request
  - It minimizes the cost of each individual request
  - How competitive is this algorithm?

Balance Algorithm

- Choose a server that would have moved the minimum total distance of any server
  - Takes care of previous bad example since eventually the second server would be employed
  - Can be shown to be k-competitive if k = n-1
  - Can do poorly in other situations

Follow-OPT

- On i^{th} request compute final configuration X achieved by OPT
- Use the server that would result in the same configuration X

RES Algorithm for k = 2

- Define Residues
  - $R_c(\sigma, S) = c \cdot C_{OPT}(\sigma, S) - C_A(\sigma)$
- v_1 = location of last request
- v_2 = location of other server
- Figures out which server would result in smaller residues.
- RES is 2-competitive

HARMONIC Algorithm

- Natural, memoryless, randomized algorithm
  - Choose a server with probability inversely proportional to its distance to request location
- Expected to be α-competitive
  - $\alpha = 3^{17000}$ for k = 3
  - $\alpha = O(k2^k)$ for general k

Related Problems and Results

- Points on a Line
- Points on a circle
- Points on a tree

- (2n-1)-competitive algorithms exist

Work Function (WF) Algorithm

- Compute the configuration X_i achieved by OPT and closest to previous configuration X_{i-1}
  - Very expensive computationally
- WF is (2k-1)-competitive
- WF is 2-competitive for k = 2

Notation

- Metric Space M with distances d(a,b)
- Configuration S = subset of k vertices from M (location of the k servers)
- Requests: $\sigma = \{r_1, r_2, \ldots\}$
- Solution: Sequence of configurations $S_0, S_1, \ldots$
- Algorithm A: $D_A(S_0, \sigma) = \Sigma_t d(S_{t-1}, S_t)$
  - $d(S_a, S_b)$ = min weight matching between S_a & S_b
- Analysis: $D_A(S_0, \sigma) \leq \rho D_{OPT}(S_0, \sigma) + f(S_0)$

Performance Ratio

OPT: Offline Algorithm

- Argue that you only need to consider lazy moves (no unnecessary moves)
- Use dynamic programming
  - Recurrence?
  - Subproblems?

Function of states & request seq

$$C_{OPT}(\epsilon, S) = \begin{cases} 0, & \text{if } S = S_0 \\ \text{undefined}, & \text{otherwise.} \end{cases}$$

$$C_{OPT}(\sigma v, S) = \begin{cases} \min_T C_{OPT}(\sigma, T) + d(T, S), & \text{if } v \text{ is covered in } S \\ \text{undefined}, & \text{otherwise.} \end{cases}$$

Important Open Problems

- Minimize ρ, where
  - $D_A(S_0, \sigma) \leq \rho D_{OPT}(S_0, \sigma) + f(S_0)$
- Competitive ratio of Algorithm/Problem
- k-Server Conjecture: For every metric space, the competitive ratio of the k-server problem is exactly k
- Randomized k-Server Conjecture: For every metric space, there exists a randomized algorithm with competitive ratio O(log k)