CS6170: Randomized Algorithms
Problem Set #3

Name: Chathurvedhi Talapaneni — Marks: 25

Roll No: CS20B021 — Due: Oct 25, 23:59

Problem 1 — 5 marks

Consider the following deterministic paging algorithms.

- Longest-Time First: Whenever there is a cache-miss evict the item that has been in the cache for the longest period of time.
- Least Frequently Used: Whenever there is a cache-miss, evict the item that has been requested least often.

(a) (3 marks) Show that the LTF paging scheme is k-competitive where k is the size of the cache.

Solution:

Let us take a case where LTF paging scheme has n page faults and we divide the series of requests into phases of k faults each.

Case (1): There is a repeating page fault in a phase.

Let r_a be the repeating request which causes a page fault. From the first r_{a1} request to the second to the next one, r_{a2}, there is a point where it is removed from the cache. Let r_b be the request which removes r_{a1}.

Right after r_{a1}, r_a is the item in the cache that is present for the shortest time. Thus for it to be removed, k-1 different items are to be called to move r_a to be the item in the cache.

Let at some point the cache be $[new...r_a....old]$, such that the item in the cache present for the longer time is on the right. For moving r_a to the right we must call for an item not present in the current stack. When r_a reaches he right all the $k-1$ items to the left of r_a count to $k-1$ distinct element between r_{a1} and r_b. Along with r_a and r_b, it makes $k+1$ distinct elements per phase for case(1), $\therefore$ at least on page fault in OPT.

Case (2): All page-faults are distinct in a phase. Let σ be the request just before the phase.

(a) If σ is not one of the page faults in the phase, we have k distinct page faults plus σ present in the distinct from the k requests. Thus OPT has at least 1 page fault.

(b) If σ is one of the page faults in the phase, we have a similar case of at least $k-1$ distinct elements between σ and the element which ejects σ. $k-1$ along with item ejecting σ and the σ request in the phase, we have $k+1$ distinct elements.

$\therefore$ We have ≥ 1 page fault in OPT per k page faults in LTF. Thus making LFT k-competitive.

(b) (2 marks) Show that the competitive ratio for LFU is unbounded.

Solution:

Let us take the cache to be filled with 1 to k to begin with. As LFU is the criteria for caching, we can force the items 1 to k-1 to be called an arbitrary number of times(n). Now the items from 1 to $k-1$ have been used $n+1$ times and k has been used 1 times. After which we can call $k+1, k, k+1, k....ntimes$. This causes a total of $2n$ page faults. But OPT will only have one page fault. As n is an arbitrarily chosen number, we can change the number making the competitive ratio for LFU unbounded.

Problem 2 — 7 marks

An *edge coloring* of a graph is an assignment of colors to the edges of a graph such that no two edge that share a vertex are assigned the same color. Let us look at the online version of edge coloring where the number of vertices in the graph are fixed, and the edges arrive in an online fashion. We will assume that the degree of every vertex in the graph is at most Δ.

(a) (3 marks) Show that there exists an online algorithm that can edge color the graph with at most $2\Delta - 1$ colors.

The best that an online algorithm can be do is $\Delta + 1$, and follows from Vizing's theorem in graph theory.

Solution:

Let the online algorithm fill an edge with an appropriate color when it encounters the edge request. When an edge request is to be colored, the vertices may have a certain set of colors that have already been placed for their edges. In the worst-case scenario, let (u, v) be the edge being added and u has $\Delta - 1$ and v has $\Delta - 1$ colors which cannot be used for (u, v). Thus if even both the color sets of u and v are disjoint, we add a new color for (u, v), with a total of $2(\Delta - 1) + 1$ colors being used. Thus an online algorithm can color a graph with $2\Delta - 1$ colors.

(b) (4 marks) Show that there is no deterministic algorithm that uses fewer than $2\Delta - 1$ colors in the worst case.

To that end, consider a graph consisting of disjoint stars with Δ vertices, and edges connecting the center vertex of each of the stars to another fixed vertex. Show that for any deterministic algorithm, there is an adversarial order that can force $2\Delta - 1$ colors.

Solution:

Let us prove by contradiction by using only $2\Delta - 2$ colors.

As the adversary chooses the edges sent in an online fashion, we can take a graph with a given number of stars with degree $\Delta - 1$. We know that each of these stars has $\Delta - 1$ different colors. If have a large enough number of stars, we can get a selection of Δ number of such stars which all have the same color, identical stars in a manner. To have one set of Δ identical stars, we would need $1 + (\Delta - 1)\binom{2\Delta-2}{\Delta-1}$ total number of stars. Due to the Pigeon hole principle, we have at least one set of identical stars which is of size Δ.

Now that we have a guarantee of Δ identical stars of each with degree $\Delta - 1$, we can introduce edges to a vertex connected to all of these stars. This would need Δ more colors other than the $\Delta - 1$ colors used in the identical stars. This amounts to $2\Delta - 1$ colors which is a contradiction.

Problem 3 — 7 marks

Our goal in this problem is to show that there does not exist an online algorithm (randomized or deterministic) for the bipartite matching problem that gives a competitive ratio better than $1 - 1/e$. We will use Yao's minimax principle to achieve this.

To that end, let us construct a probability distribution over the input instances of the online bipartite matching problem. First, let us assume that the n vertices in R are known, and let π be a permutation of these vertices chosen uniformly at random. The n vertices in L arrive one-by-one and the i^{th} vertex in L has edges to the last $n - i + 1$ vertices in R according to the permutation π.

(a) (1 mark) Show that every bipartite graph sampled via the process above has a perfect matching.

Solution:
We can match the i^{th} in L to the i^{th} in R of the permutation π to create a perfect matching from the bipartite graph.

(b) (3 marks) Let A be any deterministic online algorithm for bipartite matching. Prove that for every $i \in \{1, 2, \ldots, n\}$, the probability that A matches the i^{th} vertex of R to some vertex in L is at most

$$\min\left\{\sum_{j=1}^{i} \frac{1}{n - j + 1}, 1\right\}.$$

Solution:
The probability that there is a matching for the i^{th} element from L to R is:
$\sum_{j=1}^{i}$ Pr(Matching i in the j^{th} step)
Let us try to obtain the probability for an individual step.
Pr(Matching i in the j^{th} step)
$\Rightarrow$ Pr(i not matched before j^{th} step) * Pr(i is matched / i not matched prior)
Let k be the number of vertices in L are already matched to R in the range j to n.
Let $p = n - j + 1$ and $k \leq p$
Pr(i has not been matched before j^{th} step) = $\binom{p-1}{k} / \binom{p}{k} = (p - k)/p$
Pr(i is matched / i not matched prior) = $1/(p - k)$
$\Rightarrow$ Pr(Matching in j^{th} step) = $1/p = 1/(n - j + 1)$ if $k < p$
$\Rightarrow$ Pr(Matching in j^{th} step) $\leq 1/(n - j + 1)$

Probability that i^{th} vertex is matched:
$\sum_{j=1}^{i} \Pr(\text{Matching i in the } j^{th} \text{ step})$
$\leq \sum_{j=1}^{i} \frac{1}{n-j+1}$
As the sum crosses 1 for a given j,
$\leq \min\left\{\sum_{j=1}^{i} \frac{1}{n-j+1}, 1\right\}$

(c) (3 marks) Use the part above to conclude that for any deterministic online algorithm for bipartite matching, the the expected size of the matching computed by it is at most $n(1 - 1/e)$.

Solution:
Expected size of matching:
$\sum_{i=1}^{n} \min\left\{\sum_{j=1}^{i} \frac{1}{n-j+1}, 1\right\}$
Let k be a number such that
For $i = n - k \Rightarrow \sum_{j=1}^{i} \frac{1}{n-j+1} < 1$
For $i = n - k + 1 \Rightarrow \sum_{j=1}^{i} \frac{1}{n-j+1} \geq 1$
Now evaluating the sum for expected size of matching:
$\Rightarrow \sum_{i=1}^{n-k} \sum_{j=1}^{i} \frac{1}{n-j+1} + \sum_{i=n-k+1}^{n} 1$
$\Rightarrow \sum_{i=k+1}^{n} \frac{i-k}{i} + k$
$\Rightarrow (n-k) - \sum_{i=k+1}^{n} \frac{k}{i} + k$
$\Rightarrow n - k \sum_{i=k+1}^{n} \frac{1}{i}$
$\Rightarrow n + 1 - k \sum_{i=k}^{n} \frac{1}{i}$
$\Rightarrow n + 1 - k \sum_{j=1}^{n-k+1} \frac{1}{n-j+1} \leq n - k + 1$
Now we need a lower bound for k.
We have For $i = n - k \Rightarrow \sum_{j=1}^{i} \frac{1}{n-j+1} < 1$ and let $n - k = p$
$\sum_{j=1}^{p} \frac{1}{n-j+1} \leq 1$
$\Rightarrow \sum_{j=0}^{p-1} \frac{1}{n(1-j/n)} \leq 1$
$\Rightarrow \int_{0}^{p-1} 1/(1-x)dx < \sum_{j=0}^{p-1} \frac{1}{n(1-j/n)} \leq 1$
$\Rightarrow -ln(1 - \frac{p-1}{n}) < 1$
$\Rightarrow n - k = p < n(1 - 1/e) + 1$
Expected size of matching $\leq n - k + 1 < n(1 - 1/e) + 2$

Problem 4 — 6 marks

A coloring of a graph is an assignment of colors to the vertices in the graph. A graph is said to be k-colorable, if there is a coloring of the graph with k colors such that no two adjacent vertices have the same color. Let G be a 3-colorable graph.

(a) (2 marks) Show that there exists a coloring of the graph with 2 colors such that no triangle is monochromatic. (A monochromatic triangle is one that has all vertices of the same color.)

Do not use Part (b) to solve this question.

Solution:

Let us start with a coloring of 3 colors for the 3-colorable graph. Let the 3 colors be c1, c2, c3. For any triangle in the graph, it has all the 3 colors in it. Thus if we pick all the vertices with the color c3, they cover one vertex of all the possible triangles with the other colors c1 and c2. Thus if we recolor all the vertices with c3 to c1 or c2, we can be sure that all the triangles are not monochromatic as they have at least 1 c1-colored vertex and 1 c2-colored vertex.

(b) (4 marks) Consider the following algorithm for coloring G with two colors so that no triangle is monochromatic. The algorithm begins with an arbitrary 2-coloring. While there are monochromatic triangles in G, the algorithm chooses one such triangle and changes the color of a randomly chosen vertex of the triangle. Derive an upper bound on the expected number of such recoloring steps before the algorithm finds a 2-coloring with the desired property.

Solution:

Let us take a possible 3-colored instance of the graph G and let the colors be c_1, c_2, c_3. Let the set of vertices be colored as c_i be V_i and $|V_1| \le |V_2| \le |V_3|$. From the previous question we get that if any one color is chosen from the 3 and is changed to either of the other 2, we obtain a graph with no monochromatic triangles.

Thus we take the 2 possible end configurations such that all vertices of V_1 are of one color, all vertices of V_2 are of the other color and all vertices of V_3 are of any color. Thus we have a Markov chain of $|V_1|+|V_2|+1$ nodes from 0 to $|V_1|+|V_2|$ and endpoints be viable configurations. A random 2 coloring places the graph at any position on the Markov chain.

Node 0 represents all V_1 be colored as c_2 and opposite for V_2.

Node $|V_1| + |V_2|$ represents all V_1 be colored as c_1 and opposite for V_2

As we choose a monochromatic triangle and flip the color of one vertex, we have the possible cases:

(1) With probability $1/3$, We flip $x \in V_3$ and stay at the same node.

(2) With probability $1/3$ We flip $x \in V_1$ and we either move forward or backward wp $1/3$

(3) Similar to case(2) for $x \in V_2$

For both endpoints, self-loop wp 1.

Thus the Markov chain has (1)Self-loop wp $1/3$, (2)Forward edge wp $1/3$, (3)Backward edge wp $1/3$.

Now we have to estimate the expected number of moves from a configuration in node k such that it reaches one of the endpoints. Let S_k be the expected number of moves to reach an endpoint from node k.

$|V_1| + |V_2| = m < 2n/3$

$S_k = (1/3)((S_k + 1) + (S_{k+1} + 1) + (S_{k-1} + 1))\forall x \in [2, m - 2]$

$S_1 = (1/3)((S_1 + 1) + (S_2 + 1) + (1))$

$S_{m-1} = (1/3)((S_{m-1} + 1) + (S_{m-2} + 1) + (1))$

From induction, we prove $S_k = 3(k)(n - k)/2$

Now for the expected number of moves to reach an endpoint

$\Rightarrow \sum_{k=1}^{m-1} S_k \binom{m}{k}/2^m$

$\Rightarrow (3/2^{m+1}[n(\sum k\binom{m}{k})) - (\sum k^2\binom{m}{k}))]$

$\Rightarrow (3/2^{m+1}[m(m.2^{m-1}) - (m.2^{m-1} + m(m - 1)2^{m-2})]$

$\Rightarrow 3(m^2 - m)/8 \leq (3/8)(4n^2/9 - 2n/3)$

Expected number of recoloring steps $\leq (2n^2 - 3)/12$