Randomized Online Paging - 2

Ankit Aggarwal, Movin Jain

Supervisor: Dr. Naveen Garg, Dr. Kavitha Telikepalli

1 Problem

We have a cache of capacity k pages. The paging device is getting requests for pages in a sequence (unknown) and it has to look for these in the cache. If a page is found in the cache, we call it a HIT, and otherwise we call it a MISS. On a HIT, the requested page is simply served from the cache and no cost is incurred. On a MISS, one of the existing pages in the cache is *evicted*, and in its place the requested page is loaded into the cache from the disk, and a cost of one unit (one disk access) is incurred.

The problem is to design the EVICTION STRATEGY.

The *competitive ratio* of a deterministic online paging algorithm is defined as follows:

A deterministic online paging algorithm A is said to be c-competitive if for every request sequence, the cost incurred by the algorithm is at most c times the cost incurred by the optimal offline algorithm.

Here clearly the *cost* incurred by any algorithm would be the number of misses incurred by it.

For a randomized online paging algorithm, the above definition of competitive ratio changes in the sense that the cost incurred is replaced by the *expected* cost incurred.

2 Theorem

In this section we wish to prove that:

If H is any randomized online paging algorithm, then its competitive ratio is no better than H_k, the k^{th} harmonic number:

$$H_k = \sum_{j=1}^{k} \frac{1}{j}$$

2.1 Preliminaries

For simplicity we assume that there are a total of $k+1$ pages. This means that at any instant of time there would be only 1 page which is not present in the cache.

Let $A_1, A_2, \ldots, A_D$ be all the deterministic online paging algorithms and $R_1, R_2, \ldots, R_\lambda$ be all possible request sequences of length N such that no two consecutive requests are same.

Now, any randomized online paging algorithm H is really a probability distribution over deterministic online paging algorithms, because we can pretend that all the coin tosses that take place during the execution of H are carried out at the very beginning, which fixes our strategy of eviction for that particular run of H (Which would be nothing but a deterministic online paging algorithm).

Now consider the following table:

	R_1	...	R_j	...	R_λ
A_1	.	.	.	.	.
A_2	.	.	.	.	.
...	.	.	.	.	.
...	.	.	.	.	.
A_i	.	.	C_{ij}	.	.
...	.	.	.	.	.
...	.	.	.	.	.
A_D	.	.	.	.	.

Each entry c_{ij} denotes the ratio of the cost incurred by algorithm A_i on input R_j, to the cost incurred by the optimal offline algorithm OPT on the request sequence R_j.

Let $P = (p_1, p_2, \ldots, p_D)$ be a probability distribution on the rows of the above table, and $Q = (q_1, q_2, \ldots, q_\lambda)$ be a probability distribution on the columns. Hence we have,

$$\sum_{i=1}^{D} p_i = 1$$

and

$$\sum_{j=1}^{\lambda} q_j = 1$$

.

2.2 Claim

$$\max_{1\leq j\leq \lambda} \sum_{i=1}^{D} p_i c_{ij} \geq \min_{1\leq i\leq D} \sum_{j=1}^{\lambda} q_j c_{ij}$$

This claim follows directly from *Yao's Minimax Lemma*.

2.3 Development of the Proof

Let H be any randomized online paging algorithm. Let P be the probability distribution over all the deterministic online paging algorithms (rows of our above table) that represents the executions of H. Then, the left hand side in Yao's Minimax Lemma becomes exactly the competitive ratio of H. This is because, the competitive ratio achieved by A_i on R_j is $c_{ij} = \alpha_{ij}/\beta_j$ where α_{ij} is the cost incurred by A_i on R_j, and β_j is the cost incurred by the optimal offline algorithm on R_j. Hence the expected cost incurred by H on R_j would be

$$\sum_{i=1}^{D} p_i \alpha_{ij}$$

Hence the competitive ratio of H, considering only R_j as the possible input, would be

$$\frac{\sum_{i=i}^{D} p_i \alpha_{ij}}{\beta_j}$$

$$= \sum_{i=1}^{D} p_i c_{ij}$$

Therefore the competitive ratio of H for all general inputs would be

$$\max_{j} \sum_{i=1}^{D} p_i c_{ij}$$

which is precisely the left hand side of Yao's Lemma.

Now by Yao's Lemma, we know that the left hand side is greater than or equal to the right hand side. Hence, the competitive ratio of H would be no better than the value of the right hand side for any distribution Q. Now we know that

$$c_{ij} = \alpha_{ij}/\beta_j$$

We fix Q as follows:

$$q_j = \frac{\beta_j}{\sum_{k=1}^{\lambda} \beta_k}$$

Hence the sum

$$\sum_{j=1}^{\lambda} q_j c_{ij}$$

becomes

$$\sum_{j=1}^{\lambda} \frac{\beta_j}{\sum_{k=1}^{\lambda} \beta_k} \frac{\alpha_{ij}}{\beta_j}$$

$$= \frac{\sum_{j=1}^{\lambda} \alpha_{ij}}{\sum_{k=1}^{\lambda} \beta_k}$$

Therefore the right hand side which is equal to

$$\min_i \sum_{j=1}^{\lambda} q_j c_{ij}$$

becomes

$$\min_i \left(\frac{\sum_{j=1}^{\lambda} \alpha_{ij}}{\sum_{k=1}^{\lambda} \beta_k} \right)$$

Note in this expression that the *min* operator just serves to minimize the argument over all the $A_i's$

Now consider any deterministic online paging algorithm A_i. Let us attempt to evaluate the term inside the *min* operator above. The numerator is the only part that would depend on A_i. Let us look at the numerator only.

$$\sum_{j=1}^{\lambda} \alpha_{ij}$$

which is really the total cost incurred by A_i on all the inputs $R_1, R_2, \ldots, R_\lambda$. Let it be called S_i. Then, we have

$$\frac{S_i}{\lambda}$$

= (Expected cost incurred by A_i on an input from $R_1, R_2, \ldots, R_\lambda$, chosen uniformly at random)

To find the expected cost incurred by A_i, we can simply add the probabilities of miss at each step. (Considering that the input is coming randomly but according to the specified restriction that no two consecutive requests can be same). Now at any step, there are k pages in the cache, and one of these was just requested. Now any one of the k others can be requested, each with probability $1/k$. But only one would lead to a miss (the one that is not in the cache currently). Hence probability of a miss is $1/k$. There are N requests in total, so the expected number of misses is N/k.

Hence,

$$\frac{S_i}{\lambda} = \frac{N}{k}$$

. Therefore,

$$S_i = \frac{N\lambda}{k}$$

Now in the right hand side of Yao's Lemma, we saw that the *min* operator is being applied to the ratio of S_i and the denominator. But the denominator is independent of i. So we may consider that the *min* operator is only being applied to the numerator. But we have seen that the numerator S_i is simply $N\lambda/k$. Hence the *min* operator makes no difference. Hence the expression for the right hand side now becomes

$$\frac{N\lambda/k}{\sum_{j=1}^{\lambda}\beta_j}$$

Let us now attempt to evaluate the denominator. In the optimal offline algorithm, we know that a miss occurs when $k+1$ distinct pages have been requested after the last miss. Hence, if we mentally divide all the inputs $R_1, R_2, \ldots, R_\lambda$ into pieces, each of which has exactly $k+1$ distinct requests, then the cost incurred by the optimal offline algorithm would simply be the number of pieces.

$$\sum_{j=1}^{\lambda}\beta_j$$

= Total cost incurred by optimal offline algorithm on all inputs $R_1, R_2, \ldots, R_\lambda$
= Total number of pieces (say l).

Now, total length of all pieces = $N\lambda$.

Therefore,

$$\frac{N\lambda}{No.\ of\ pieces}$$

= Expected length of a piece

Now, the expected length of a piece is simply the expected number of requests to be seen in order to cover all $k+1$ distinct requests. This can be modelled as a random walk on a complete graph G on $k+1$ vertices, where the current vertex cannot be visited in the immediately next step. Let u_m be the expected number of steps taken to see the $m+1^{th}$ distinct vertex after m vertices have been seen.

So we need the sum $u_1 + u_2 + \ldots + u_k$

Now consider a situation where we have already seen m vertices, and we take the next step. The current vertex is one of the m already visited vertices. All other vertices can be visited in the next step, each with probability $1/k$. Hence we can visit an already visited vertex with probability $(i-1)/k$, and we can visit a new vertex with probability $(k+1-i)/k$.

Hence we can write:

$$u_m = (\frac{i-1}{k})(1+u_m) + (\frac{k+1-i}{k})(1)$$

Hence, $u_m = k/(k+1-i)$.

Therefore, the sum

$$u_1 + u_2 + \ldots + u_k$$

becomes

$$k + k/2 + k/3 + \ldots + k/k \quad = \quad kH_k$$

.

Hence the denominator in the right hand side of Yao's Lemma becomes

$$\frac{N\lambda}{kH_k}$$

Therefore the right hand side becomes

$$\frac{N\lambda/k}{N\lambda/kH_k} \quad = \quad H_k$$

Hence we conclude that the competitive ratio of any randomized online paging algorithm cannot be better than H_k, the k^{th} harmonic number.