Online Algorithms

Xiaolan Zhang

Topics

1. Introduction to paging algorithms
2. Definition of competitiveness
3. Randomized online algorithms
4. Oblivious adversary
5. Game theory and an application

PAGING PROBLEM DESCRIPTION

A processor requests a piece of data item from a local cache.

Cache size The cache can hold at most k items.

Cache hit The item requested is located in the cache, no cost for data access.

Cache miss The item requested is not located in the cache. The item will be fetched from the main memory at one unit cost and evict one existent item in cache to make room for the new comer.

Cost measure The number of misses on a sequence of requests.

A paging algorithm will decide which k items to retain in the cache in order to minimize the miss rate.

PAGING ALGORITHMS

Category of algorithms:

Offline algorithms generates optimal solution given the input of complete information.

Online algorithms makes decision given the input of information up to time. No future information is available at the decision moment.

Realistic paging algorithms are online because future memory access information is not given at the time of cache line eviction.

Online deterministic algorithms:

LRU cache evicts the Least Recently Used item.

FIFO cache evicts the oldest item.

LFU cache evicts the Least Frequently Used item.

Offline optimal algorithm **MIN**: cache evicts the item whose next request occurs furthest in the given sequence. It has been proved to be optimal among all offline algorithms.

FORMAL MODEL

Definitions:

A online algorithm

O offline optimal algorithm

$\Phi_N = (\rho_1 \rho_2 \ldots \rho_N)$ a sequence of requests of length N

$f_A(\Phi_N)$ number of times A misses on sequence Φ_N

$f_O(\Phi_N)$ number of times O misses on sequence Φ_N

Some conclusions:

Theorem 1. *For any deterministic online algorithm A, there is always a sequence resulting the miss rate 100%*

Theorem 2. *If the memory is of size $k+1$, MIN misses no more than N/k times on any sequence of length N.*

UPPER BOUND FOR MIN

Proof. Suppose initially k items in a cache. Only one item in the memory is not in the cache. The first miss will occur when the request refers to the one not in cache.

Claim the *worst case sequence* of requests be the one evicted by MIN located at $k+1$th position in sequence. According to MIN, it won't be requested during the next $k-1$ accesses. Since the size of cache is k, the rest $k-1$ items are still located in the cache. No miss will occur. If this request pattern repeats, MIN will have 1 miss on every k requests. Therefore the total miss on N items is N/k.

Then we show that *worst case sequence* actually results the maximum miss rate. It is impossible to construct a sequence such that the one evicted is located at $\leq k$th position because $k-1$ items in cache will not all show up in a sequence of $k-2$. If the one evicted is located $\geq k+1$, fewer misses will result.

Therefore, miss rate N/k is the worst case for MIN.

□

DEFINITION OF COMPETITIVENESS

Definition 1. An algorithm A is $\mathcal{C} - competitive$ if there exists a constant b such that for all sequence Φ_N,

$$f_A(\Phi_N) - \mathcal{C} f_O(\Phi_N) \leq b$$

Denote the competitiveness coefficient $\mathcal{C}_A$ be the infimum of $\mathcal{C}$. Note that b is not related to N.

Competitiveness is a measurement of the effectiveness of an online algorithm. Consider the online algorithm and the offline algorithm is a pair of player in a game. The goal of A is to minimize $\mathcal{C}$ by reducing $f_A(\cdot)$ while O acts as an opponent to hinder A's minimization by reducing $f_O(\cdot)$. Consider $\mathcal{C}_A$ be the best strategy A could ever had to beat O.

Theorem 3. *For any deterministic online algorithm* A, $\mathcal{C}_A \geq k$.

Proof. It trivially follows the result of Theorem 2. A sequence can be constructed such that A misses N times. However, N/k is the smallest misses O could ever have among all worst case sequences because worse case miss rate is proportional to the memory size. $N - \mathcal{C}N/k \leq b$ holds for all sequences since $f_A \leq N$ and $f_O \geq N/k$ as largest lower bound. Since b is a constant independent to N, $\mathcal{C}_A \geq k$. □

RANDOMIZED ONLINE ALGORITHMS

A randomized algorithm makes random choice for each step from all deterministic algorithms in some probability distribution.

Denote R be a randomized algorithm, the number of misses on Φ_N is a random variable $f_R(\Phi_N)$.

ADVERSARIES

Consider adversaries are opponents offline algorithm O who plays "minimize competitiveness games" with a randomized online algorithm. Adversaries are categorized by powerfulness.

Oblivious adversary has no knowledge of the random choices made by R.

Consider a game such that O makes all choices before R make any choice. Of course, R will never "see" O's choices since it is an online algorithm.

Adaptive adversary determines its strategy based on the past choices made by A.

Online adversary makes choices step by step as R makes choices.

Offline adversary makes choices after R finishes all choices over a sequence. It is most powerful.

OBLIVIOUS ADVERSARIES

Oblivious adversary is the weakest overall adversaries.

Definition 2. An algorithm R is $\mathcal{C} - competitive$ over oblivious adversary if there exists a constant b such that for all sequence Φ_N,

$$E[f_R(\Phi_N)] - \mathcal{C} f_O(\Phi_N) \leq b$$

Denote the competitiveness coefficient $\mathcal{C}_R^{obl}$ be the infimum of $\mathcal{C}$. Note that b is not related to N.

Similarly we could define competitiveness over adaptive online adversary $\mathcal{C}_R^{aon}$ and adaptive offline adversary $\mathcal{C}_R^{aof}$. The following inequality holds

$$\mathcal{C}_R^{obl} \leq \mathcal{C}_R^{aon} \leq \mathcal{C}_R^{aof} \leq \mathcal{C}_A$$

Consider there is a minimization game: the goal of player a is to minimize profit. The goal of player b is to prevent a to minimize profit. Let $\mathscr{A}$ be the set of strategies a uses, $\mathscr{B}$ be the set of strategies b uses. Assume $\mathscr{A}\mathscr{B}$ are finite and indexable. $H(i,j)$ be the profit a gains by using strategy $A_i \in \mathscr{A}$ and b using strategy $B_i \in \mathscr{B}$.

Definition 3. a saddle point (i^*, j^*) in a game is a point such that

$$H(i, j^*) \geq H(i^*, j^*) \geq H(i^*, j)$$

If a plays with i^*, it guaranteed to get no more than $H(i^*, j^*)$ no matter how strong b is. Similarly, if b plays with j^*, it is guaranteed to make a gain less than $H(i^*, j^*)$. A zero-sum game doesn't have to have a saddle point (For example, the rock, paper, scissors game has no saddle point.) A saddle point is considered to be an equilibrium, because neither player would have incentive to change, given the choice of the other player.

We use x to denote a probability distribution over i, and y to denote a probabilty distribution over j. A choice of i is a pure strategy for the first player, and a choice of x is a randomized strategy. We use $H(x, y)$ to denote the average of $H(i, j)$, when i and j are chosen randomly and independently, according to x and y. That is, $H(x, y) = \sum_{i,j} H(i, j)x_i y_j$.

Theorem 4 (von-Neumann's Minimax Theory)**.** *For any finite two person's zero sum game,*

$$\max_y \min_x H(x, y) = \min_x \max_y H(x, y)$$

Whoever, a or b, plays hard first against all other's choices, the result would be the same.

Theorem 5 (Yao's Principle)**.** *For any finite two person's zero sum game,*

$$\min_x H(x, y) \leq \min_x \max_y H(x, y)$$

APPLICATION ON ONLINE PAGING

Consider paging problem is a minimization game between algorithms and adversaries. In order to find a lower bound for $\mathcal{C}_R^{obl}$, we find a equivalent strategy set that deterministic algorithm A plays against optimal offline algorithm O over sequences generated by some probability distribution P. Denote $f_P(\Phi_N) = E[f_O(\Phi_N)]$. Then, A's opponent becomes P. Similarly, we define $\mathcal{C}_A^P$ as the infimum of C such that $E[f_A(\Phi_N)] - \mathcal{C} f_P(\Phi_N) \leq b$

Let the paging game cost be $H(x, y) = \mathcal{C}_x^y$. According to von-Neumann's Principle,

$$\inf_R \sup_{obl} \mathcal{C}_R^{obl} = \inf_R \mathcal{C}_R^{obl} = \sup_P \inf_A \mathcal{C}_A^P$$

Therefore, for any randomized algorithm R and any probability distribution P on the input sequences, $\mathcal{C}_R^{obl} \geq \inf_A \mathcal{C}_A^P$. That is, the performance ratio of the optimal deterministic algorithm for a given distribution on input sequences, can be used to bound below the competitive ratio (i.e. bound above the performance) of what randomized algorithms can accomplish.

References

[1] R. Motwani and P. Raghavan, *Randomized Algorithms*, 1995

[2] A. Borodin and Rl El-Yaniv, *Online Computation and Competitive Analysis*, 1998