Advanced Algorithms January 5, 2006

The k-Server Problem - Lecture 11

Lecturer: Yair Bartal *Scribe: Anton Bar*

The k-Server Problem

The k-Server Problem is a generalization of the paging problem.

A metric space is a couple $M = (V, d)$, where V is a set of items, and d is a function $d : V \times V \rightarrow \Re^+$ with the following properties:

- Reflexivity: $D(u, v) = 0$ iff $u = v$
- Symmetry: $\forall u, v : d(u, v) = d(v, u)$
- Triangle inequality: $d(u, v) + d(v, w) \geq d(u, w)$

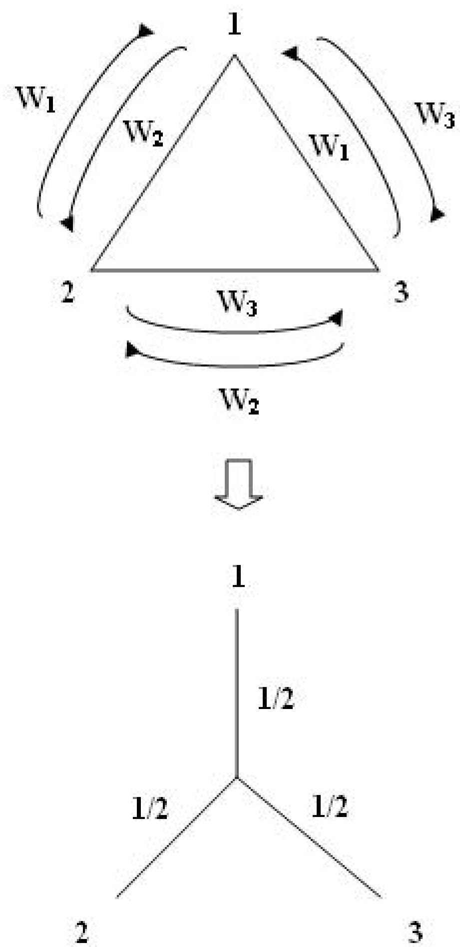
The k-Server Problem description: We are given metric space M over n items. There are k servers on items of M. $\sigma = \sigma_1\sigma_2...$ are series of requests, where each σ_i represents an item in M. The algorithm is required to move one of the servers to σ_i. The price of one movement equals to the distance between the source and the target items.

The offline version of the problem is polynomial.

Remark: the paging problem is a special case of the k-servers problem, where M is uniform, where all distances are 1 and each item represents a single memory page. Items with servers represent pages in cache.

There is a possible extension of the paging problem where each page has a different price of transferring to cache. Lets assume that page i has a price w_i.

Example 1: In the case of triangle we can perform the following transformation where the server always returns to the center:



Example 2: Another example is where M is a line, such as in the case of hard disk heads, with 3 items A, B and C such that $d(A, B) < d(B, C)$ and two servers on A and C. In this metric, greedy algorithm does not perform well in cases like this: $\sigma = BABABA...$ (k times BA). Greedy will pay $d(A, B) \cdot |\sigma|$, while OPT pays only $d(B, C)$, because it moves the server from C to B right from the beginning of the series. This is despite the fact that a single moving a server from A to B is less expensive.

Competitiveness: There are two definitions of competitiveness:

- Algorithm ON is competitive if there exists constant a s.t. $\forall\sigma : ON(\sigma) \le \alpha{\cdot}OPT(\sigma) + a \Rightarrow \lim\left(\sup_{|\sigma|\to\infty} \frac{ON(\sigma)}{OPT(\sigma)}\right) \le \alpha$.
- Algorithm ON is strongly competitive if $\forall\sigma : ON(\sigma) \le \alpha \cdot OPT(\sigma)$.

Theorem 1 (Mannase, McGeoch, and Sleator) *Competitive ratio of deterministic algorithms for the k-servers problem is at least k, for any metric space M.*

Theorem 2 (Koutsoupias and Papdimitriou) *For any metric space M, there exists an* $(2k-1)$*-competitive algorithm for the k-servers problem.*

Theorem 3 *There exists a k-competitive algorithm for the k-servers problem in a line metric space.*

Double Coverage Online Algorithm

Lets mark servers S_1, S_2,... S_k on a line from left to right. We can always keep this order because every movement of a server can be replaced by a series of movements of adjacent servers.

There are two different cases:

1. The request is located on one side of all servers. Without loss of generality we'll assume that it's on on the right of S_k. In this case we will move S_k.
2. The request is located on point r between S_i and S_{i+1}. Assume without loss of generality that r is closer to S_i. Denote $d(S_i, r) = \delta$. We will move S_i to f and also will move S_{i+1} in the direction of r at the distance of δ. Note that this solves the problem introduced in Example 2 above, because, unlike Greedy, this algorithm will gradually move the server located at C to B.

Analysis of the Algorithm:
Lets define potential function $\Phi : V^k \times V^k \to \Re^+$.

We distinguish between three different cases during the handling of a single request:

1. Time t - before request $t+1$ but after request t.
2. Time t' - intermediate step when the adversary already handled the new request, but ON did not yet.

3. Time $t+1$ - ON handles the new request.

Lets find out the properties of the potential function Φ:

1. $\forall t : \Phi_{t'} - \Phi_t \leq k \cdot \Delta ADV_t$, where ΔADV_t is the cost of ADV to serve request t.
2. $\forall t : \Phi_{t+1} - \Phi_{t'} \leq -\Delta ON_t$, where ΔON_t is the cost of ON to serve request t.

Lemma 4 *If such Φ exists then ON is k-competitive.*

Proof: From both properties follows: $\forall t : \Phi_{t+1} - \Phi_t \leq k \cdot \Delta ADV_t - \Delta ON_t$. Assuming that the length of our request series is f, $\sum_{0 \leq t \leq f} (\Phi_{t+1} - \Phi_t) \leq k \cdot \Delta ADV(\sigma) - \Delta ON(\sigma)$. The sum is telescopic, hence: $\Phi_0 \leq \Phi_f - \Phi_0$ and $ON(\sigma) \leq k \cdot ADV(\sigma) + \Phi_0$, where Φ_0 is a fixed cost of the initial configuration. ■

Now we will define the desired function Φ. It will be a combination of two functions: $\Phi = k\Psi + \Theta$, where Ψ measures the distance between ON and ADV, and Θ is a bound for the sum constant. Order both algorithms in two parts - all servers of ON in a line on one side and all servers of ADV in a line on the other side. Derive a fully connected bipartite graph. Find a minimal weight bipartite matching, which in the case of line metric, will connect ADV_1 to ON_1, ADV_2 to ON_2 and so on till ADV_k to ON_k.

$\Psi = \sum_{i=1}^{k} d(S_i, a_i)$ and $\Theta = \sum_{1 \leq i < j \leq k}^{k} d(S_i, S_j)$.

Lets analyze the properties of this function:

1. $t \to t' : \Delta\Phi = \Phi_{t'} - \Phi_t$. Lets assume that the adversary moves server a_l to the request at r, hence the cost is $\Delta ADV_t = d(a_l, r)$, $\Delta\Theta = 0$, $\Delta\Psi \leq d(a_l, r)$, $\Delta\Phi \leq k \cdot \Delta ADV_t$.
2. $t' \to t+1 : \Delta\Phi = \Phi_{t+1} - \Phi'_t$.
   a) The request r is on once side of all servers, assume that it's on the right of S_k. Thus $\Delta\Theta = (k-1)d(S_k, r)$, $\Delta\Psi = -d(S_k, r)$, $\Delta\Phi = -k\Delta\Psi + \Delta\Theta = -k \cdot d(S_k, r) + (k-1) \cdot d(S_k, r) = -d(S_k, r) = -\Delta ON_t$.
   b) The request r is between S_i and S_{i+1}. Then $\Delta ON_t = 2d(S_i, r) = 2\delta$ (two times because both S_i and S_{i+1} move). Thus $\Delta\Theta = -2\delta$, $\Delta\Psi \leq -\delta + \delta$ (minus distance between S_i and a_i, plus distance between S_{i+1} and a_i) for both $l \leq i$ and for $l > i$. Hence $\Delta\Psi \leq -2\delta = -\Delta ON_t$.