Lecture 10: February 20

Lecturer: Yair Bartal *Scribe: Felix Wu*

In this lecture we describe the page migration problem, giving a 3-competitive randomized algorithm, a 5-competitive deterministic algorithm, and a deterministic algorithm which is 3-competitive for the uniform metric.

10.1 Page Migration

In the *page migration* problem, due to Black and Sleator [BS89], we are given a network, with a single page located at some node in the network. Each request is a node which wishes to access the page, and this request is served by establishing communication between the node x at which the page resides and the node y requesting the page. The cost of this communication is $d(x, y)$, the (shortest-path) distance between the nodes. In addition, we can decide to move the page from node x to node y at a migration cost of $D \cdot d(x, y)$. As always, the goal is to minimize costs, which in this case amounts to deciding when it would be useful to move the page from one node to another.

Note that if we have more than one page in the network, we can deal with each page separately and then superimpose the solutions, though this requires that there be no limit on the number of pages which can reside at a single node. Note also that in this formulation of the problem, we can only move pages around; we cannot copy them. In practice, this is realistic because the creation of multiple copies of a page leads to problems with maintaining consistency. More complicated models, such as the *file allocation* problem and *distributed paging* have been defined by Bartal, Fiat and Rabani [BFR92].

10.1.1 A 3-competitive Randomized Algorithm

The intuitive idea here is that we need about D requests in a region before we can decide that it is worthwhile to move the page to that region. Hence, for any one request, we move the page with probability about $1/D$.

More precisely, when the page is at s and the request is at r, the FLIP algorithm first serves the request, and then moves the page from s to r with probability $\frac{1}{2D}$. This algorithm is due to Westbrook [West91].

Claim 10.1 *FLIP is 3-competitive against adaptive on-line adversaries.*

Proof: We define the potential function

$$\Phi = 3D \cdot d(t, s),$$

where t is the location of the adversary's page, and s is the location of FLIP's page. To show that FLIP is 3-competitive, we want to show that for each request r,

$$\mathrm{E}[\Delta\Phi] \leq 3\mathrm{Cost}_{\mathrm{ADV}} - \mathrm{E}[\mathrm{Cost}_{\mathrm{FLIP}}].$$

To do this, we divide each request into a number of steps:

1. The adversary (possibly) moves.
2. Request occurs.
3. Request is serviced.
4. FLIP (possibly) moves.

Step 1 The adversary moves from t to t'.

$$\begin{aligned}
\mathrm{Cost}_{\mathrm{ADV}} &= D \cdot d(t,t'). \\
\mathrm{Cost}_{\mathrm{FLIP}} &= 0. \\
\Delta\Phi &= 3D(d(t',s) - d(t,s)) \\
&\leq 3Dd(t,t') \\
&= 3\mathrm{Cost}_{\mathrm{ADV}} - \mathrm{Cost}_{\mathrm{FLIP}}.
\end{aligned}$$

The inequality follows from the triangle inequality, in this case, $d(t',s) \leq d(t,s) + d(t,t')$.

Steps 2-4

$$\begin{aligned}
\mathrm{Cost}_{\mathrm{ADV}} &= d(t,r). \\
\mathrm{E}[\mathrm{Cost}_{\mathrm{FLIP}}] &= d(s,r) + \tfrac{1}{2D}(D \cdot d(s,r)) \\
&= \tfrac{3}{2}d(s,r). \\
\mathrm{E}[\Delta\Phi] &= (\tfrac{1}{2D})3D(d(t,r) - d(t,s)).
\end{aligned}$$

With probability $\frac{1}{2D}$, FLIP moves from s to r, thereby incurring an extra cost, and changing the value of Φ.

$$\begin{aligned}
\mathrm{E}[\Delta\Phi] + \mathrm{E}[\mathrm{Cost}_{\mathrm{FLIP}}] &= \tfrac{3}{2}(d(t,r) - d(t,s)) + \tfrac{3}{2}d(s,r) \\
&= \tfrac{3}{2}(d(s,r) - d(t,s) + d(t,r)) \\
&\leq \tfrac{3}{2}(d(t,r) + d(t,r)) \\
&= 3d(t,r) = 3\mathrm{Cost}_{\mathrm{ADV}}.
\end{aligned}$$

Here we are again using the triangle inequality.

Having shown now that $\mathrm{E}[\Delta\Phi] \leq 3\mathrm{Cost}_{\mathrm{ADV}} - \mathrm{E}[\mathrm{Cost}_{\mathrm{FLIP}}]$, it follows that FLIP is 3-competitive. ∎

10.1.2 Summary of Previous Results

The following are some results which have been proved regarding randomized algorithms for the page migration problem.

- A lower bound against an oblivious adversary is $2 + \frac{1}{2D}$ [CLRW93].
- Matching upper bounds can be obtained for 2-point metric spaces and for trees [CLRW93].
- There exists an algorithm which is $(1+\phi)$-competitive against an oblivious adversary [West91].
- There exists an algorithm which is 2.28-competitive against an oblivious adversary in a uniform metric space [West91].
- FLIP is 3-competitive against an adaptive on-line adversary. (There is also a matching lower bound.) [West91].

The following are some results which have been proved regarding deterministic algorithms for the page migration problem.

- For a 2-point metric space, we have a lower bound of 3 [BS89].
- There are 3-competitive algorithms for uniform and trees networks [BS89].
- For $D = 1$, we have a lower bound of $\frac{85}{27} \approx 3.148$ [CLRW93].
- We have 7-competitive [ABF93a] and 4.086-competitive [BCI96] algorithms.

10.1.3 A 5-competitive Deterministic Algorithm

This algorithm is a slight simplification of the 4.086-competitive algorithm of Bartal, Charikar and Indyk [BCI96]. This algorithm and those mentioned above all share the same basic structure as follows:

- Divide the request sequence into phases of length D.
- Within each phase, keep the page at the same node b.
- At the end of a phase, migrate the page from b to some other node x.

The difference in the algorithms comes in deciding how to choose the new location x. If $v_1, \ldots, v_D$ are the requests in a phase, the 7-competitive, Move-To-Min algorithm of Awerbuch, Bartal and Fiat [ABF93a] moves to the argmin of $\sum_{i=1}^{D} d(x, v_i)$. The idea is to move close to the request activity after D requests, since the optimal algorithm is either there or is paying costs much larger than the formula above to serve all of those requests from another location.

However, we also want to minimize the movement cost, so to get a 5-competitive algorithm, we take x to be the argmin of

$$\sum_{i=1}^{D} d(x, v_i) + \tfrac{1}{2} D d(b, x).$$

We call this algorithm Move-To-Local-Min (MTLM).

(To get a 4.086-competitive algorithm, we take phases of length cD for some constant c, and give different weights in the formula above, but we will not analyze that algorithm here.)

Claim 10.2 *The* Move-To-Local-Min *algorithm is 5-competitive.*

Proof: We take the potential function

$$\Phi = 2D \cdot d(b, a).$$

where b is the location of MTLM's page and a is the location of the adversary's page.

Consider some phase $v_1, \ldots, v_D$. Suppose MTLM's page is at b during the phase, while the adversary's page is at a_{i-1} before request v_i. (Note that the optimal algorithm might move its page around within the phase.)

$$\begin{aligned}
\text{Cost}_{\text{MTLM}} &= \sum_{i=1}^{D} d(b, v_i) + D \cdot d(b, x) \\
&\le \sum_{i=1}^{D} d(b, a_0) + \sum_{i=1}^{D} d(a_0, v_i) + D \cdot d(b, x) \\
&= \sum_{i=1}^{D} d(a_0, v_i) + D(d(b, a_0) + d(b, x)). \\
\text{Cost}_{\text{ADV}} &= \sum_{i=1}^{D} d(a_{i-1}, v_i) + D \sum_{i=1}^{D} d(a_{i-1}, a_i).
\end{aligned}$$

Lemma 10.3 $\forall j$, $\text{Cost}_{\text{ADV}} \ge \sum_{i=1}^{D} d(a_j, v_i)$.

Proof: For $r \le s$,

$$d(a_r, a_s) \le d(a_r, a_{r+1}) + \cdots + d(a_{s-1}, a_s) \le \sum_{k=1}^{D} d(a_{k-1}, a_k).$$

Hence, we get

$$d(a_j, a_{i-1}) \le \sum_{k=1}^{D} d(a_{k-1}, a_k).$$

Adding $d(a_{i-1}, v_i)$ to both sides and using the triangle inequality, we get

$$d(a_j, v_i) \leq d(a_{i-1}, v_i) + \sum_{k=1}^{D} d(a_{k-1}, a_k).$$

Summing over i gives us

$$\sum_{i=1}^{D} d(a_j, v_i) \leq \sum_{i=1}^{D} d(a_{i-1}, v_i) + D \sum_{k=1}^{D} d(a_{k-1}, a_k).$$

The right hand side of this last inequality is Cost_{ADV}, proving the lemma. ∎

We can now proceed with the proof of 5-competitiveness.

$$\begin{aligned}
\Delta\Phi &= 2D(d(a_D, x) - d(a_o, b)) \\
&= 2\sum_{i=1}^{D} d(a_D, x) - 2Dd(a_o, b) \\
&\leq 2\sum_{i=1}^{D} d(a_D, v_i) + 2\sum_{i=1}^{D} d(v_i, x) - 2Dd(a_o, b). \\
\Delta\Phi + \text{Cost}_{\text{MTLM}} &\leq 2\sum_{i=1}^{D} d(a_D, v_i) + 2\sum_{i=1}^{D} d(v_i, x) + \sum_{i=1}^{D} d(a_0, v_i) - Dd(a_0, b) + Dd(b, x).
\end{aligned}$$

We now use the fact that x is the argmin of $\sum_{i=1}^{D} d(x, v_i) + \frac{1}{2}Dd(b, x)$ to claim that

$$2\sum_{i=1}^{D} d(x, v_i) + Dd(b, x) \leq 2\sum_{i=1}^{D} d(a_0, v_i) + Dd(b, a_0).$$

This implies

$$\begin{aligned}
\Delta\Phi + \text{Cost}_{\text{MTLM}} &\leq 3\sum_{i=1}^{D} d(a_o, v_i) + 2\sum_{i=1}^{D} d(a_D, v_i) \\
&\leq 5\text{Cost}_{\text{ADV}}.
\end{aligned}$$

where the last inequality follows from the lemma above. ∎

Open Problem 10.4 *Close the gap between the deterministic upper and lower bounds, perhaps with a non-phase-based algorithm. Specifically, it would be interesting to analyze the work function algorithm for the page migration problem.*

10.1.4 Uniform Metric Algorithm

A 3-competitive uniform metric algorithm and a matching lower bound were first shown by [BS89]. We present here a simpler 3-competitive algorithm, based on [BFR92, BCI96]. Note that for a uniform metric, we can simply maintain a counter for each node, which effectively describes the request sequence so far.

The algorithm Count is as follows:

1. Initialize all counters $c(p)$ to 0.
2. When a request comes at p, increment $c(p)$.
3. If $c(p) = D$,
   (a) Migrate the page to p (if it is not already there).
   (b) Reset $c(p)$ to 0 (but not the other counters).
   (c) Wait until D requests occur at nodes other than p, not incrementing any counters in the meanwhile.
4. Begin incrementing counters again until the next counter reaches D.

Claim 10.5 *The algorithm above is 3-competitive.*

Proof: Divide the request sequence into phases, corresponding to the iterations of the loop. Each set contains the D requests at p which caused the page to be migrated to p, as well as the D or more requests during which no counters were incremented. Note that these phases may be overlapping in time, though every request belongs to exactly one phase.

Within each phase, $\text{Cost}_{\text{Count}} \le 3D$. The algorithm pays no more than D for the D requests at p, no more than D for the movement to p, and D for the D non-p requests during the waiting step.

For the adversary, we assign any migration away from x to the phase of x during which it occurred. We can then associate a cost $\text{Cost}_{\text{ADV}} \ge D$ to each phase as follows. If for all D requests at p, the adversary does not hold the page in p, the adversary pays a cost of D. Similarly, if for all D requests at processors other than p during the waiting step, the adversary holds the page in p, the adversary pays a cost of D. If neither of these two conditions hold, then the adversary has the page in p for at least one of the requests at p, but does not have it in p for at least one of the requests of the waiting step. Hence, somewhere between those two requests, the adversary must have migrated the page away from p, and the cost D of doing so would be assigned to this phase.

Hence, we get that $\text{Cost}_{\text{Count}}/\text{Cost}_{\text{ADV}} \le 3$. ■

References

[ABF93a] B. Awerbuch, Y. Bartal, and A. Fiat. Competitive Distributed File Allocation. In *Proc. 25th ACM Symp. on Theory of Computing*, pages 164–173, May 1993.

[BCI96] Y. Bartal, M. Charikar and P. Indyk. On Page Migration and Other Relaxed Task Systems. In *Proc. of the 8th Ann. ACM-SIAM Symp. on Discrete Algorithms*, January 1997.

[BFR92] Y. Bartal, A. Fiat, and Y. Rabani. Competitive algorithms for distributed data management. In *Proc. 24th ACM Symp. on Theory of Computing*, pages 39–50, 1992. Also accepted for publication in the STOC 92 special issue *Journal of Computer and Systems Sciences*, 1993.

[BS89] D.L. Black and D.D. Sleator. Competitive Algorithms for Replication and Migration Problems. Technical Report CMU-CS-89-201, Department of Computer Science, Carnegie-Mellon University, 1989.

[CLRW93] M. Chrobak, L. Larmore, N. Reingold, and J. Westbrook. Optimal Multiprocessor Migration Algorithms Using Work Functions. In *Proc. of the 4th International Symp. on Algorithms and Computation.* Also *Lecture Notes in Computer Science*, vol. 762, pages 406-415, Hong Kong, 1993, Springer-Verlag.

[LRWY94] C. Lund, N. Reingold, J. Westbrook, and D. Yan. On-Line Distributed Data Management. In *Proc. of European Symp. on Algorithms*, 1994.

[West91] J. Westbrook. Randomized Algorithms for Multiprocess or Page Migration. In *Proc. of DIMACS Workshop on On-Line Algorithms.* American Mathematical Society, February, 1991.