Algorithms – CS-27200

On-line 2-server problem

János Simon

The *competitive ratio* of an online algorithm for an optimization problem is the approximation ratio achieved by the online algorithm (the worst-case ratio between the solution found by the online algorithm and the optimal solution with foresight, i. e., with knowledge of all the input).

Consider the following **2-server on a line** problem. There are 2 servers, initially positioned at given integer coordinates x_0 and y_0, respectively, on the number line. At times $t = 1, 2, \cdots$ there is a *request*, an integer f_i. The request must be satisfied by moving one of the two servers to f_i. The *cost* of satisfying the request is the sum of the distances travelled by the two servers from their previous positions to the configuration where at least one of them is positioned at f_i. The 2-server problem asks for an online algorithm that satisfies a sequence of n requests. The cost of the algorithm is the sum of the costs of satisfying each request. The online algorithm does not know f_{i+1} when choosing the moves to satisfy f_i.

(a) ("Greedy is bad") Consider the (obvious) greedy algorithm: move the closest server to f_i. This has unbounded competitive ratio. **Exercise:** Prove! (Hint) Wlog, assume $y > x$ Consider the sequence of requests
$y + (y - x)/4$
$y - (y - x)/4$
repeated n/2 times.

(again, we may assume x, y multiples of 4) Eventually, each pair will incur a cost of $y - x$ to greedy, and cost 0 for the optimal solution.

Consider the following algorithm DC (Double Coverage):

Let x_i, y_i and f_i denote the positions of the servers before the i-th request, and the position of the i-th request, respectively.

- `if` $f_i \leq x_i$ `move the server at` x_i `to the request`
- `if` $f_i \geq y_i$ `move the server at` y_i `to the request`
- `otherwise` [the request is in the interval (x_i, y_i)] `move both servers, at the same speed towards the request, until one reaches it`

Claim: DC is 2-competitive.

Proof sketch. We will run both the optimal algorithm OPT, and DC, and use a potential function to limit the cost incurred by DC. Note that wlog, OPT always moves a single server (we can always add the motion by the other server later.)

Consider the position of the servers of OPT and of DC before the ith move. Consider the distance between x_i and the server of OPT closest to it: let this distance be a_i.

Similarly, let b_i be the distance between the other server of OPT and y_i. Define $M_i = a_i + b_i$

Let $S_i = y_i - x_i$ be the distance between the two servers of DC. We define the potential function $\Phi = 2M + S$. Before dealing with the i-th request, the value of the potential will be $\Phi_i = M_i + S_i$.

Now, consider a sequence of requests. We will first serve it with OPT, and estimate the change go Φ caused by the move, as well as the distance travelled by the OPT server. Then we will use DC and again compute both the change in Φ and the distances travelled by the two servers.

We will charge DC both the change in Φ and the distance.

Cost of OPT

The server moved some distance d_{OPT}. DC did not move, but will be charged the difference in Φ, for its amortized cost. The distance M between the OPT server that moved and the corresponding DC server increased by at most d_{OPT} (notice that the DC server closest to the OPT server may have changed: the bound still holds!). Since S is unchanged, the change in Φ is at most $2d_{OPT}$.

Cost of DC

There are two cases: whether the request is between the two DC servers or not.

In the case of an outside request (a request outside the interval (x, y) of the current positions of the DC servers), a single DC server moves some distance d_{DC}. Before its move, the corresponding OPT server was at the site of the request, so M decreases by exactly d_{DC}. On the other hand, S, the distance between the two DC servers increased by d_{DC}. The total change in the potential function Φ is 0.

In the case of a request between the two DC servers, we claim

- M does not increase. In order to serve the request, one of the DC servers moved to the location where an OPT server is, so the distance between them becomes 0. If the distance travelled by this DC server was d_{DC}, then M decreased by this amount. The other DC server also moved a distance of d_{DC}, which may have increased the distance between it and the corresponding OPT sever by this amount. Still, the net change in M is at most 0.
- The distance between the two DC servers decreased by $2d_{DC}$

Thus, the potential function Φ only increases during the move of OPT, and increases by at most $2d_{OPT}$ when OPT incurs a cost of d_{OPT}. Since the potential "pays" for the amortized cost of DCs moves, in a sequence of server requests these inequalities imply that the total length covered by DC's servers will be bounded by twice the length of server moves by OPT.