MaximizePayout

Isaac Gutt

March 2022

1 The Algorithm

Both lists are sorted from greatest to smallest using a comparator and there is a variable x set to 1, then there is a loop from 0 to the list size-1 that continues raising the power of the item in list A to the one in list B, and multiplying the results to x. My intuition is that this is a greedy algorithm because the sorting in the beginning causes the largest number in List A to be multiplied with the largest in List B, then the second largest from each are multiplied with each other, and so on. My greedy intuition is that if I multiply the largest numbers with each other, the largest payout will be achieved.

2 Proving Correctness

Let x denote the final variable that will be returned after running the greedy algorithm over the first i numbers of the two lists.

Lemma 1. *After i runs of the greedy algorithm taking the power of ai and bi, and multiplying it into x, x will be at least as large as after i runs in the optimal ordering of the list.*

Let m(i,H) denote the x variable after i runs of the greedy algorithm H. H^* is the optimal algorithm.
Prove by induction:
Base case: If i = 0, both the optimal and the greedy ordering will be at 1 because no runs will be made to increase it.
Inductive step: assume that the claim holds for some $0 \leq i \leq H^*$
Case 1: Proof for i+1:
If i = 1, the greedy algorithm will take the power of the biggest number in List A with the biggest in List B. This will produce the biggest possible number between the two lists, so $m(1, H) \geq m(1, H^*)$, the optimal algorithm can't possibly be ahead.
The greedy algorithm takes the power of the highest remaining numbers in each run, producing the highest possible number and multiplying it into the biggest possible number from the previous run, so $m(i+1, H) \geq m(i+1, H^*)$

Theorem 2. *Let $H = x$ after running the greedy algorithm, and $x = H^*$ after running the optimal algorithm, and $H \geq H^*$*

Proof: By the lemma, since each run the greedy algorithm will take the biggest two numbers, it will always be ahead or the same size as the optimal algorithm, even when it reaches the end. So $H \geq H^*$

Proof. After running the greedy and optimal algorithms, $H \geq H^*$ □

3 Running Time

The algorithm first sorts the two lists using list.sort() with a comparator. This is proven to be n log n. After, there is one loop that runs through both lists in one run, with Math.pow and a multiplication happening in each run, both of which are O(1) operations. Since n log n is the dominant run time of them all, the algorithm's run time is O(n logn).