857. Minimum Cost to Hire K Workers

Detailed explanation O(NlogN)

5.6K VIEWS

79

Last Edit: October 13, 2018 11:29 PM

(/lee215) lee215 (/lee215) 5569

Let's read description first and figure out the two rules:

"1. Every worker in the paid group should be paid in the ratio of their quality compared to other workers in the paid group."
So for any two workers in the paid group,
we have `wage[i] : wage[j] = quality[i] : quality[j]`
So we have `wage[i] : quality[i] = wage[j] : quality[j]`
We pay wage to every worker in the group with the same ratio compared to his own quality.

"2. Every worker in the paid group must be paid at least their minimum wage expectation."
For every worker, he has an expected ratio of wage compared to his quality.

So to minimize the total wage, we want a small ratio.
So we sort all workers with their exoected ratio, and pick up `K` first worker.
Now we have a minimum possible ratio for `K` worker and we their total quality.

As we pick up next worker with bigger ratio, we increase the ratio for whole group.
Meanwhile we remove a worker with highest quality so that we keep `K` workers in the group.
We calculate the current ratio * total quality = total wage for this group.

We redo the process and we can find the minimum total wage.
Because workers are sorted by ratio of wage/quality.
For every ratio, we find the minimum possible total quality of K workers.

Time Complexity
`O(NlogN)` for sort.
`O(NlogK)` for priority queue.

C++:

```
    double mincostToHireWorkers(vector<int> q, vector<int> w, int K) {
        vector<vector<double>> workers;
        for (int i = 0; i < q.size(); ++i)
            workers.push_back({(double)(w[i]) / q[i], (double)q[i]});
        sort(workers.begin(), workers.end());
        double res = DBL_MAX, qsum = 0;
        priority_queue<int> pq;
        for (auto worker: workers) {
            qsum += worker[1], pq.push(worker[1]);
            if (pq.size() > K) qsum -= pq.top(), pq.pop();
            if (pq.size() == K) res = min(res, qsum * worker[0]);
        }
        return res;
    }
```

Java:

```
    public double mincostToHireWorkers(int[] q, int[] w, int K) {
        double[][] workers = new double[q.length][2];
        for (int i = 0; i < q.length; ++i)
            workers[i] = new double[]{(double)(w[i]) / q[i], (double)q[i]};
        Arrays.sort(workers, (a, b) -> Double.compare(a[0], b[0]));
        double res = Double.MAX_VALUE, qsum = 0;
        PriorityQueue<Double> pq = new PriorityQueue<>();
        for (double[] worker: workers) {
            qsum += worker[1];
            pq.add(-worker[1]);
            if (pq.size() > K) qsum += pq.poll();
            if (pq.size() == K) res = Math.min(res, qsum * worker[0]);
        }
        return res;
    }
```

Python:

```
    def mincostToHireWorkers(self, quality, wage, K):
        workers = sorted([float(w) / q, q] for w, q in zip(wage, quality))
        res = float('inf')
        qsum = 0
        heap = []
        for r, q in workers:
            heapq.heappush(heap, -q)
            qsum += q
            if len(heap) > K: qsum += heapq.heappop(heap)
            if len(heap) == K: res = min(res, qsum * r)
        return res
```

FAQ:
Question: "However, it is possible that current worker has the highest quality, so you removed his quality in the last step, which leads to the problem that you are "using his ratio without him".
Answer: It doesn't matter. The same group will be calculated earlier with smaller ratio.
And it doesn't obey my logic here: For a given ratio of wage/quality, find minimum total wage of K workers.

Comments: 29

Sort By

Type comment here... (Markdown is supported)

Preview | Post

(/vallhallavskyoshmkufa)
VallhallavskyOSHMKUFA (/vallhallavskyoshmkufa) ★ 18 October 11, 2018 4:23 AM
Wow, just came up with the question that the new added worker might be ruled out from the heap immediately, then I saw the FAQ part ... Your explanation is really a silver bullet!

0 Share Reply

SHOW 1 REPLY

(/fanjingliu)
FanjingLiu (/fanjingliu) ★ 11 October 10, 2018 6:16 AM
I keep learning from you my friend, give me your venmo number!

1 Share Reply

SHOW 2 REPLIES

(/sstcurry)
sstcurry (/sstcurry) ★ 9 October 3, 2018 3:58 AM
Thanks for the explanation!

I still don't fully understand the Q&A part, so I keep my priority queue at size K-1 and combine it with the current worker to calculate total cost.

Read More

0 Share Reply

SHOW 1 REPLY

(/wangnima)

wangnima (/wangnima) ★ 16 October 2, 2018 11:19 PM

Thanks for sharing!

0 Share Reply

(/zetelight)

zetelight (/zetelight) ★ 1 October 2, 2018 1:43 AM

I really like the FAQ here! I want to add more comments for some people if they do not understand the Q.
Example:
w(wage): 50 30 70 240
q(quantity): 20 5 10 30
r(ratio): 2.5 6 7 8

Read More

0 Share Reply

(/crunner)

CRunner (/crunner) ★ 24 October 2, 2018 12:06 AM

This solution is smart. However, it takes a while for me to understand it. Your code may need induction to prove its correctness during an interview. For example, given a list of workers sorted ascending by w/q ratio, we need to prove that for each worker at position >= K (assuming that position starts from 1), the sum of the K-1 qualities in the priority queue excluding the current worker itself is minimum so that the best cost we can get for this ratio is found. For the worker at position K, it's obviously true since you can only pick the K-1 works before it. Assuming that for the worker at position N > K it's true, for the worker at position N+1, we can also make it true by doing the following: if the quality at position N is larger than the priority queue peek, do nothing, otherwise the

Read More

3 Share Reply

(/james9277)

james9277 (/james9277) ★ 7 September 27, 2018 1:13 PM

Hi, I have a question here. How can you guarantee that each time I remove largest quality can get smallest sum. Do we also need to take ratio into consideration?
What if the largest quality is the one I just added into the queue, and it got removed later, but I still use the current ratio to calculate total.

0 Share Reply

SHOW 1 REPLY

(/littlerainrain)

littleRainRain (/littlerainrain) ★ 28 September 18, 2018 8:15 PM

I don't really understand the following part .... can someone help ? haha :)

for (double[] worker: workers) {
qsum += worker[1];
pq.add(-worker[1]);

Read More

0 Share Reply

SHOW 3 REPLIES

(/azeemmohd)

azeemmohd (/azeemmohd) ★ 28 September 18, 2018 7:40 AM

A more detailed explanation based on this solution and a bit verbose yet very simple and readable code can be found here:
https://leetcode.com/problems/minimum-cost-to-hire-k-workers/discuss/171679/Detailed-Plain-English-Explanation
(https://leetcode.com/problems/minimum-cost-to-hire-k-workers/discuss/171679/Detailed-Plain-English-Explanation)

0 Share Reply

(/azeemmohd)

azeemmohd (/azeemmohd) ★ 28 September 18, 2018 6:33 AM

My doubt is when q size exceeds K, we need to remove a worker, so we do pq.poll(), shouldn't we remove this persons quality from the qsum by doing qsum -= pq.poll() instead of adding it to the qsum like in the code below:

```
if (pq.size() > K) qsum += pq.poll();
```

0 Share Reply

SHOW 1 REPLY

‹ 1 2 3 ›