Problem I
Intrinsic Interval

Submits: 42
Accepted: at least 1

First solved by: Jagiellonian 1
Jagiellonian University in Krakow
(Hlembotskyi, Stokowacki, Zieliński)
02:10:47

Author: Gustav Matula

An interval of the permutation is a consecutive subsequence consisting of consecutive numbers.

An interval of the permutation is a consecutive subsequence consisting of consecutive numbers.

An interval of the permutation is a consecutive subsequence consisting of consecutive numbers.

An interval of the permutation is a consecutive subsequence consisting of consecutive numbers.

2 3 1 6 4 7 5 8

An interval of the permutation is a consecutive subsequence consisting of consecutive numbers.

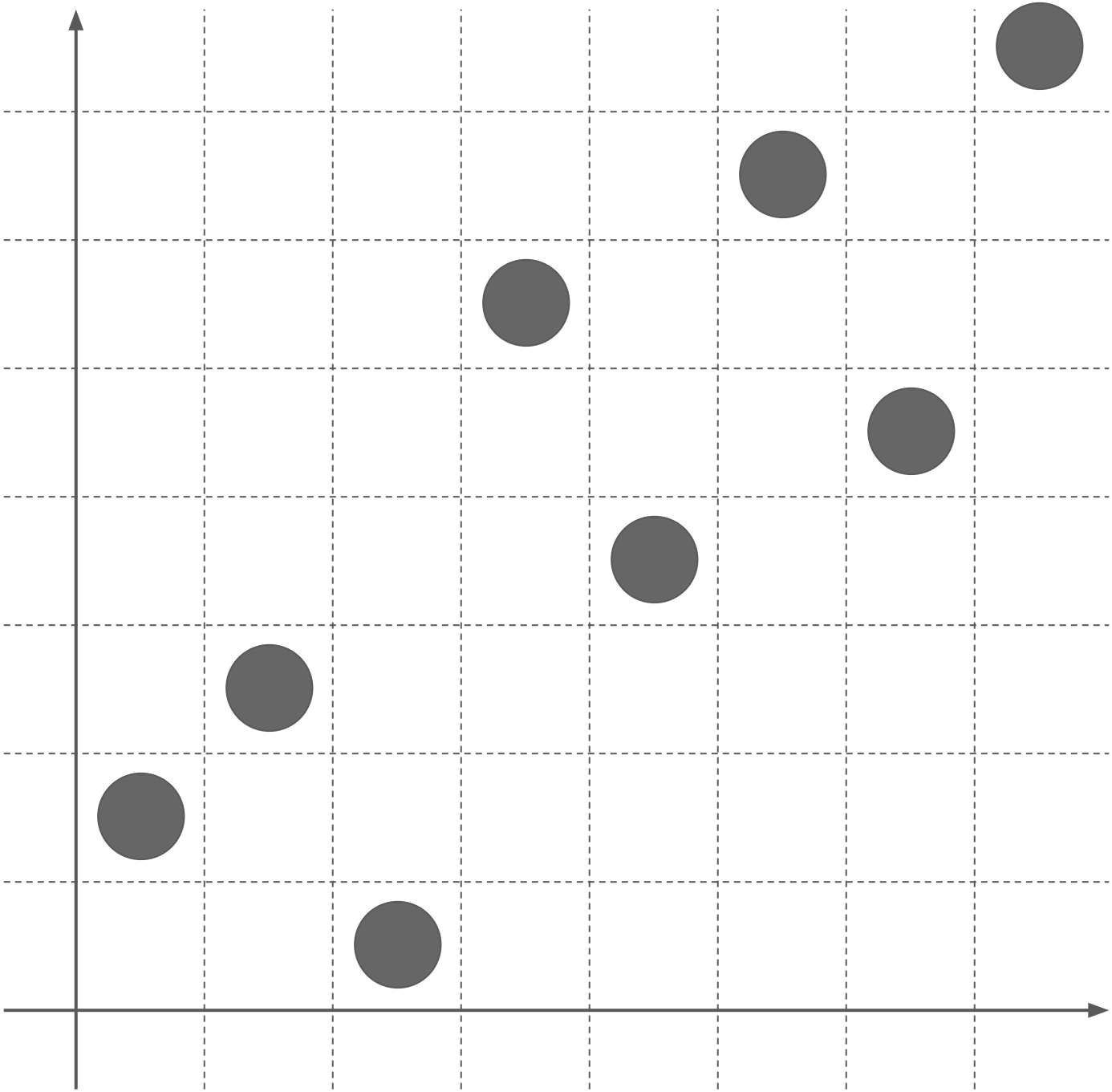
An interval of the permutation is a consecutive subsequence consisting of consecutive numbers.

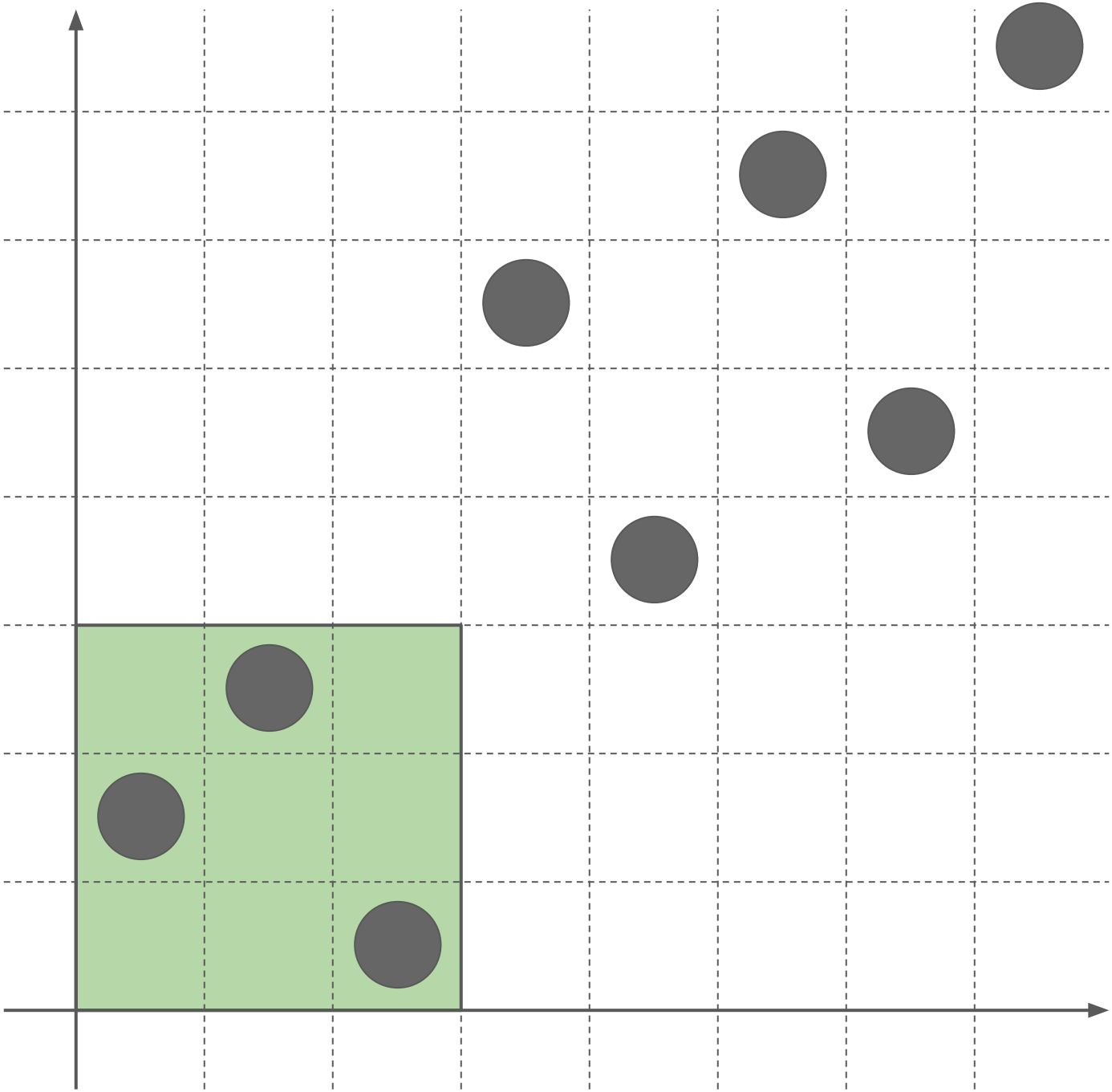
For a given subsequence we need to find the shortest enclosing interval.

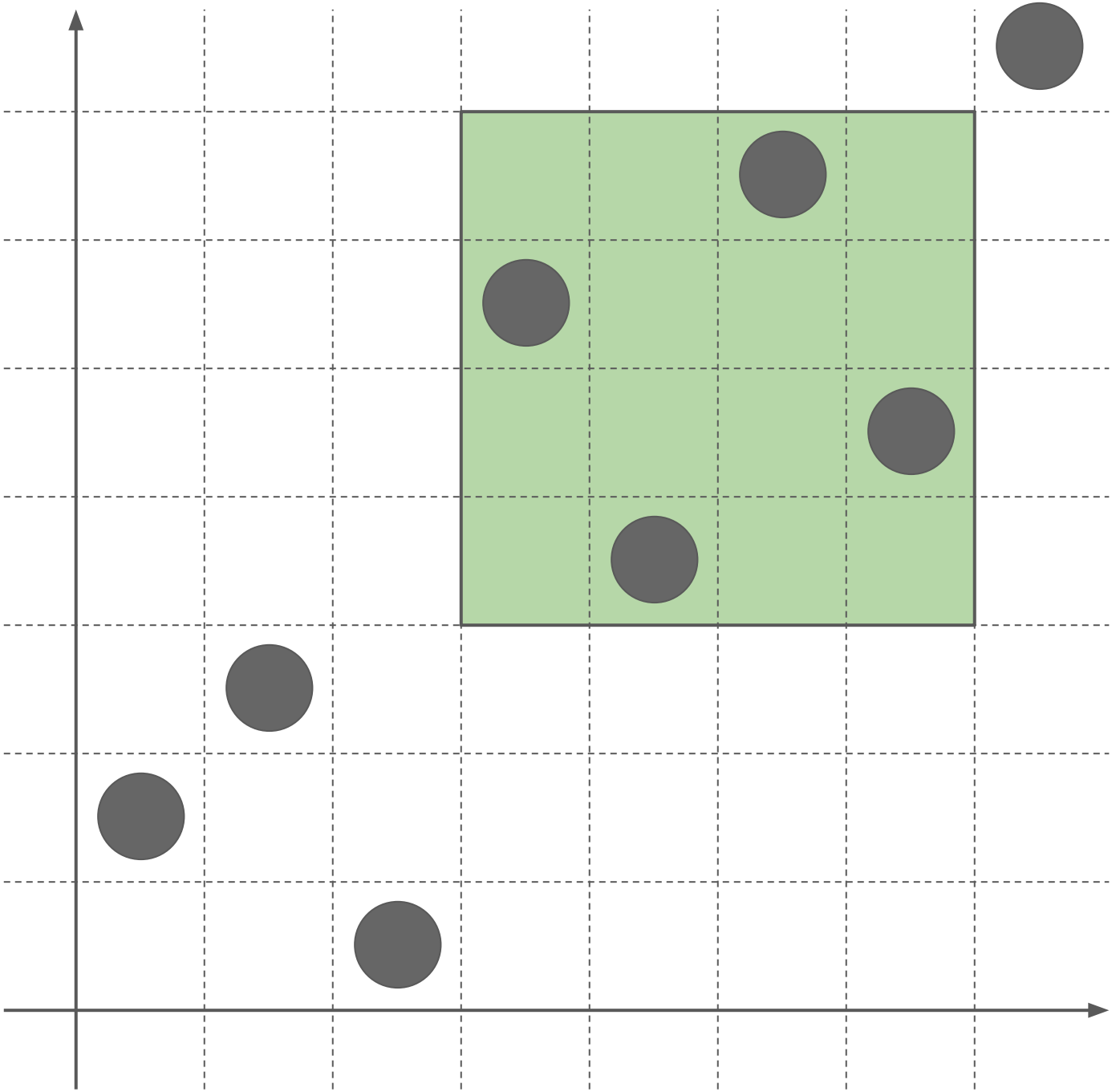
An interval of the permutation is a consecutive subsequence consisting of consecutive numbers.

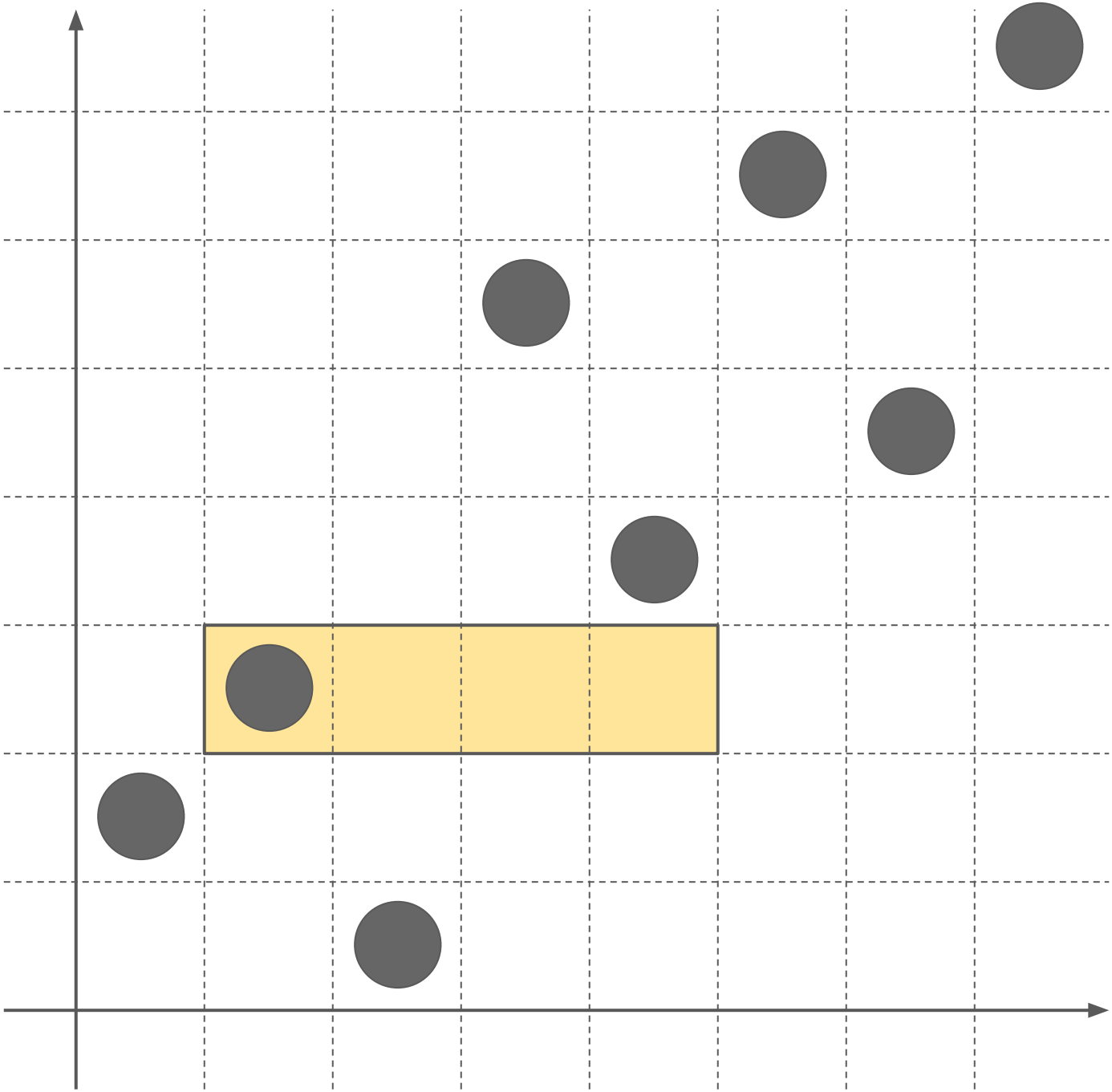
For a given subsequence we need to find the shortest enclosing interval.

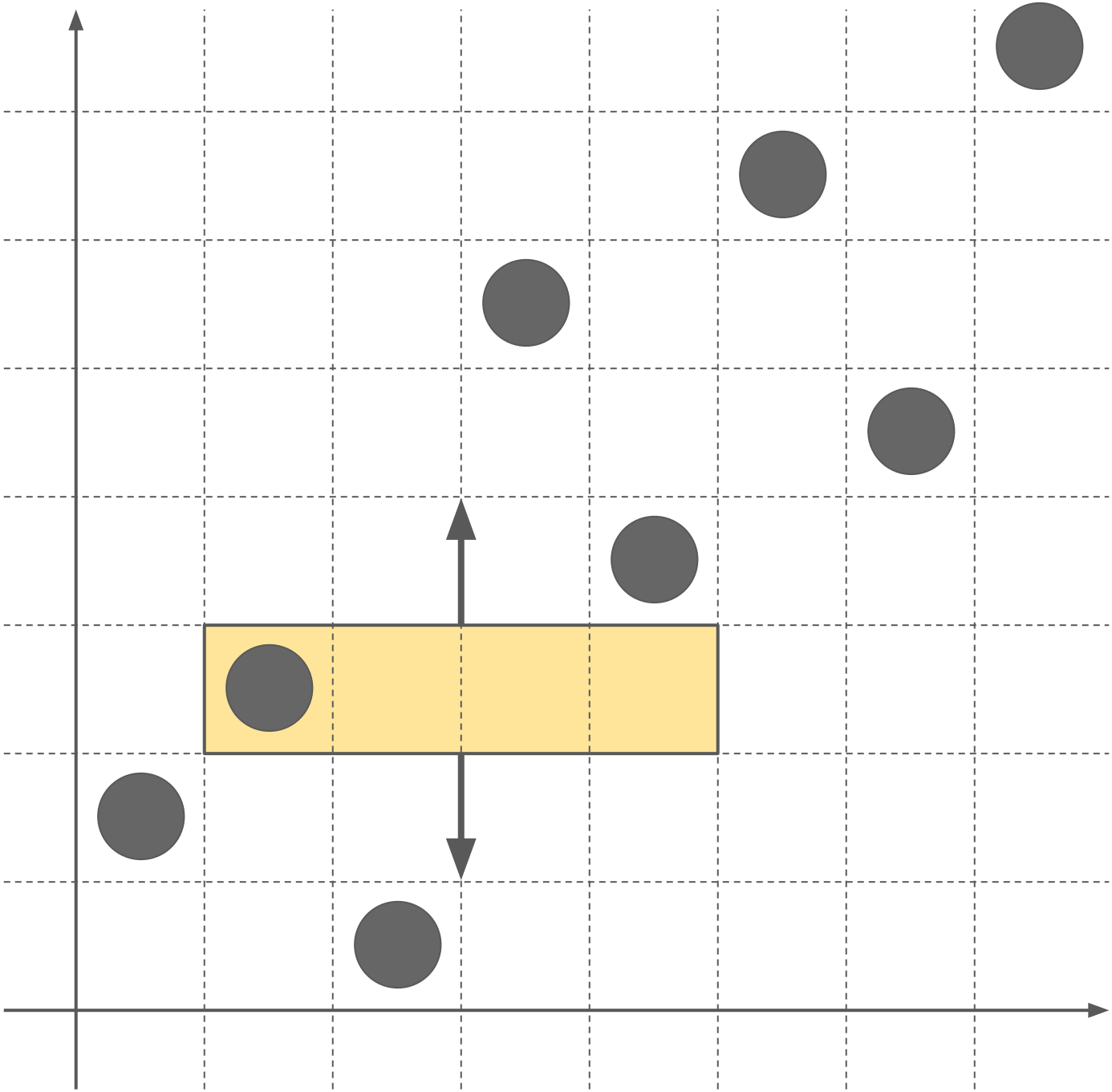
To see how we could expand the subsequence into the shortest enclosing interval, let's visualize the permutation in two dimensions.

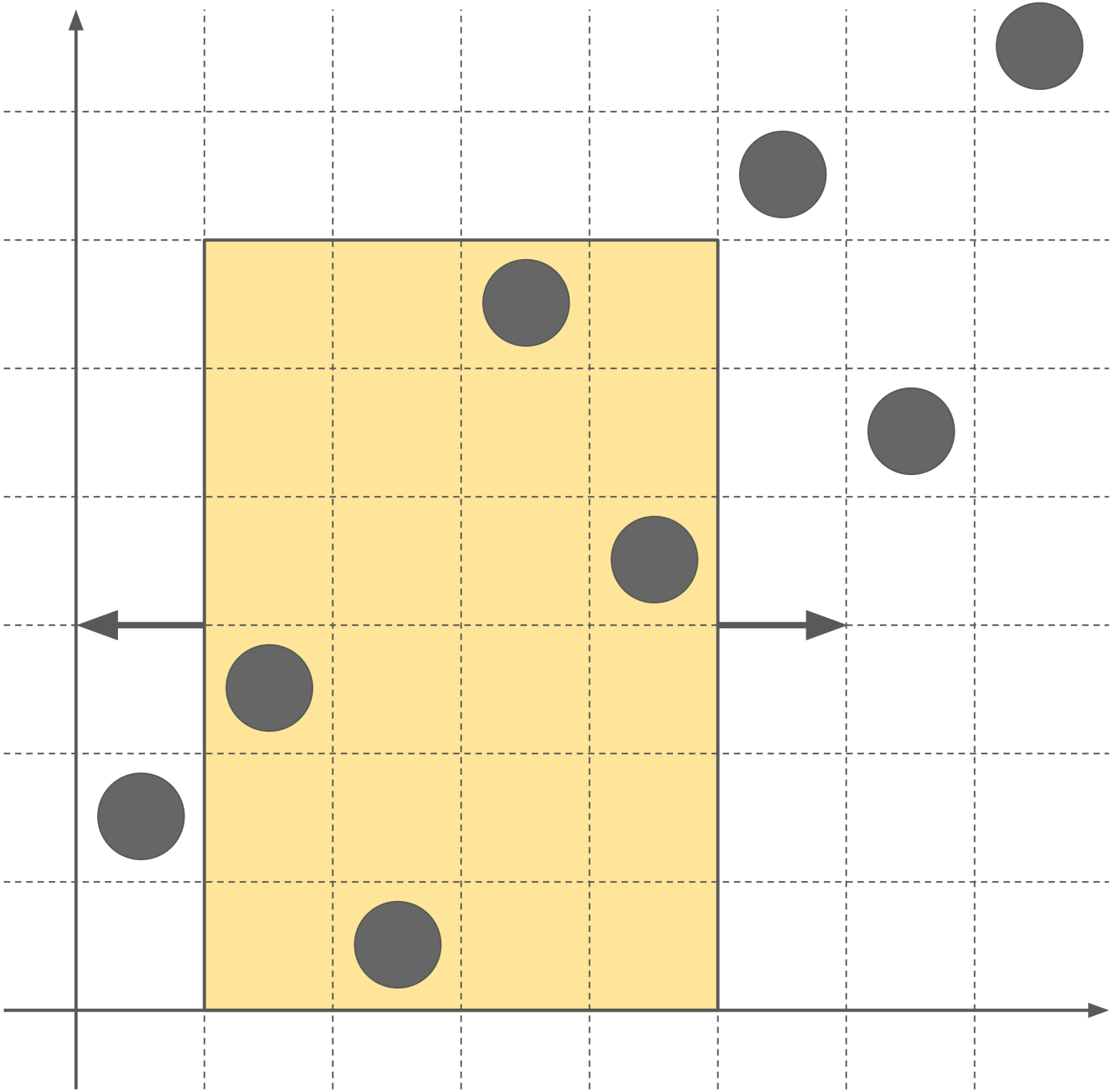


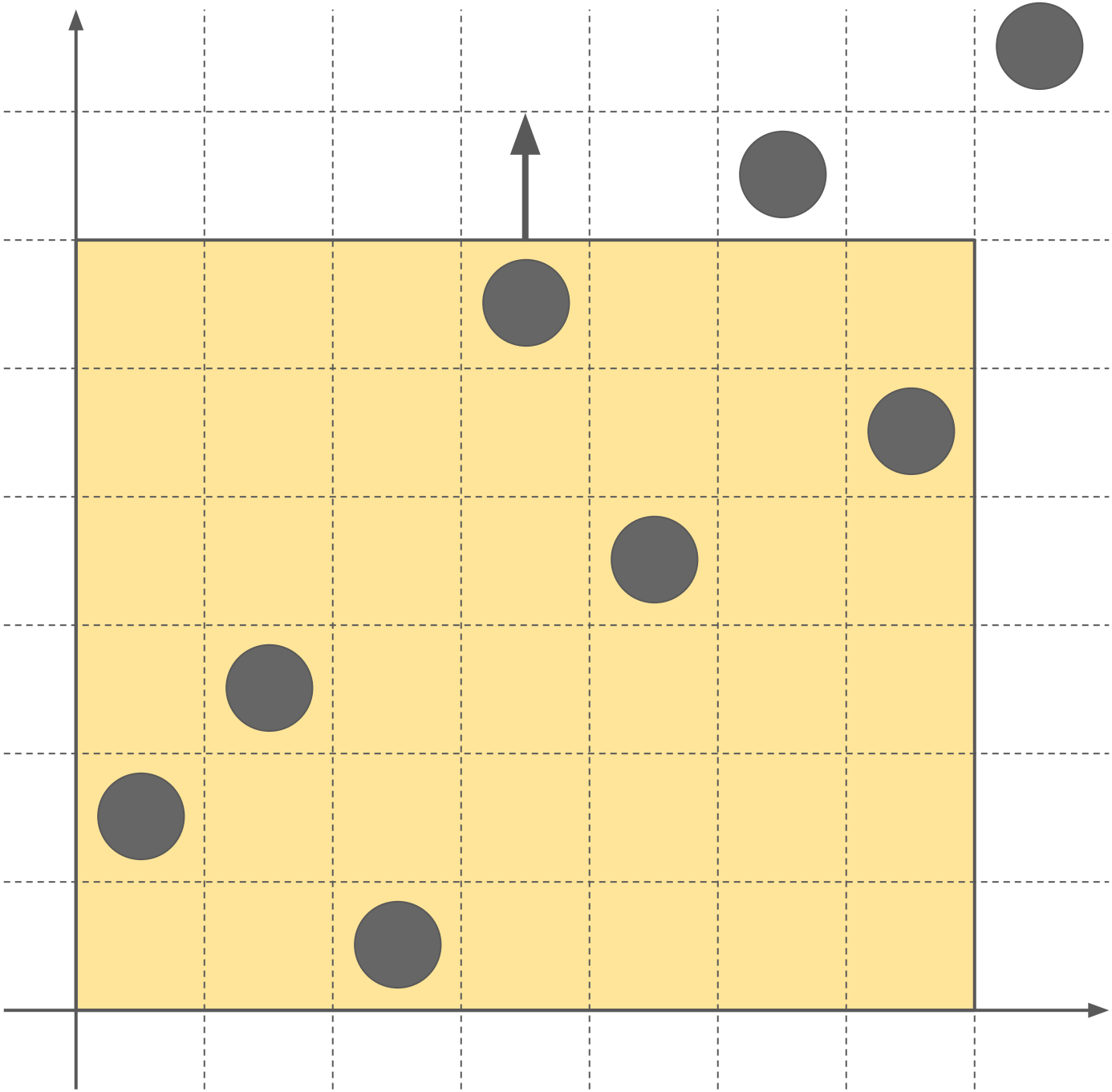


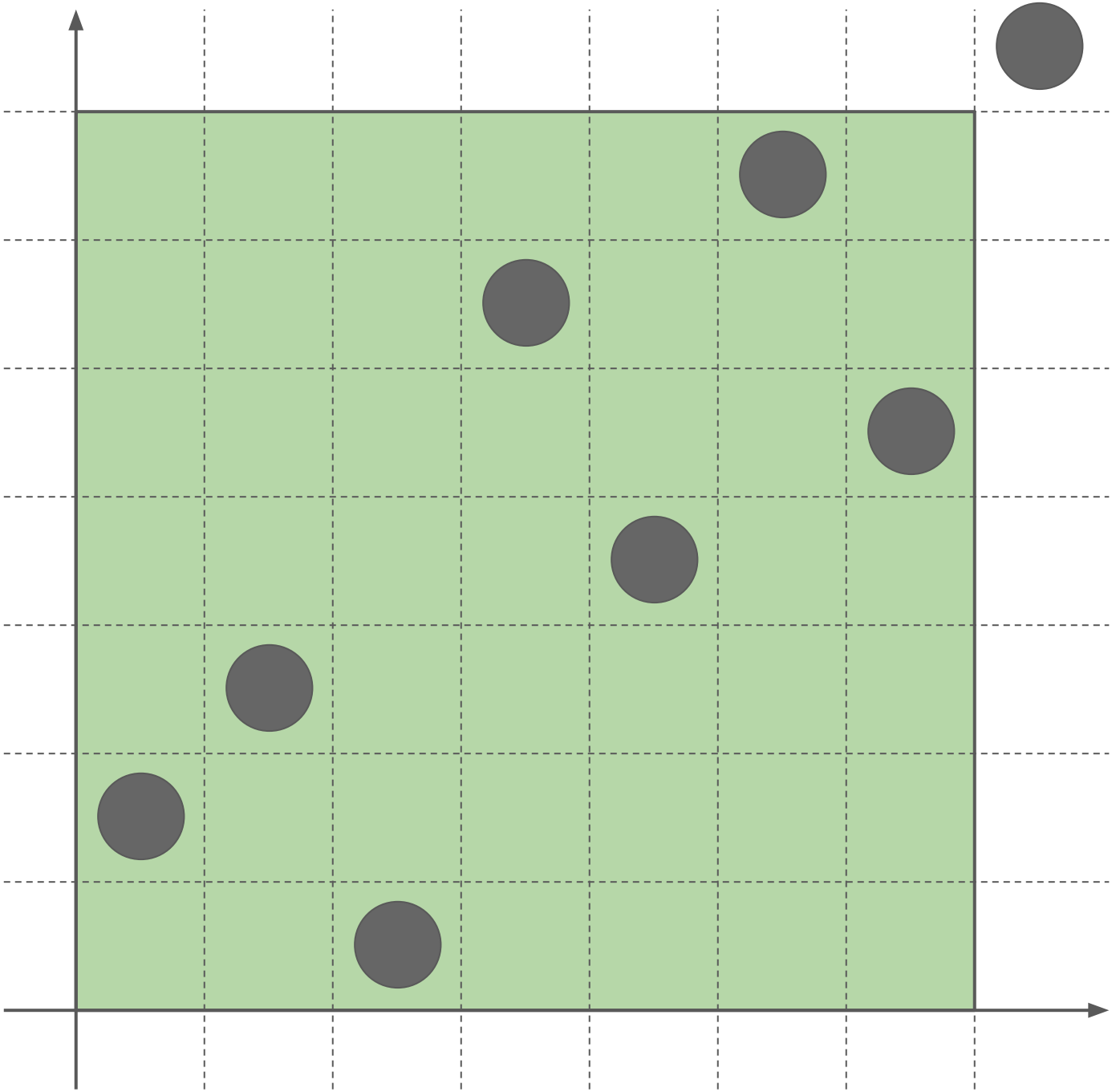












With careful implementation of the algorithm, it is possible to expand a subsequence [a, b] to an enclosing interval [x, y] in O(|y - x| - |b - a|).

However, that's too slow for this problem.

Instead, we'll develop divide and conquer algorithm to answer all queries at once.

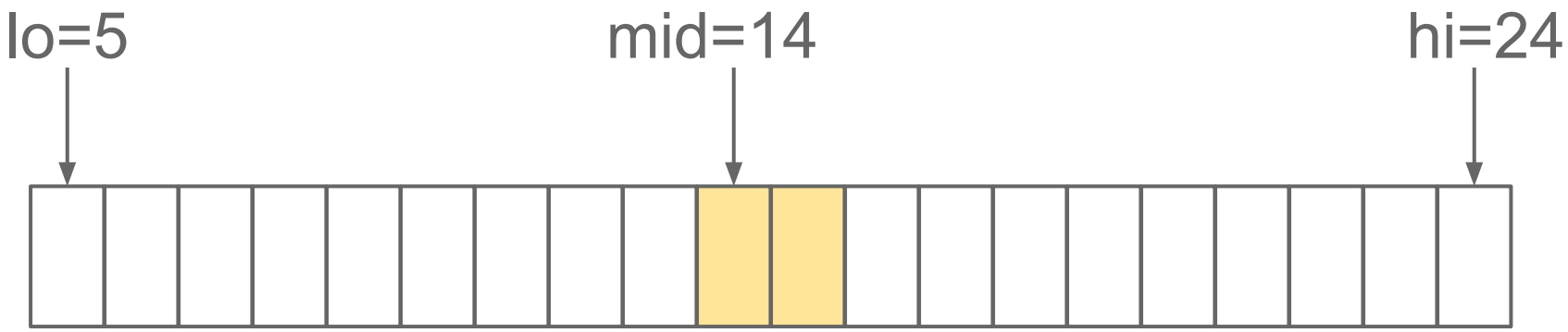
We initialize the result for each query with interval [1, n] and then we'll try to improve it.

Improve(queries, lo, hi) will try to improve each query in queries by considering intervals completely within [lo, hi] window.

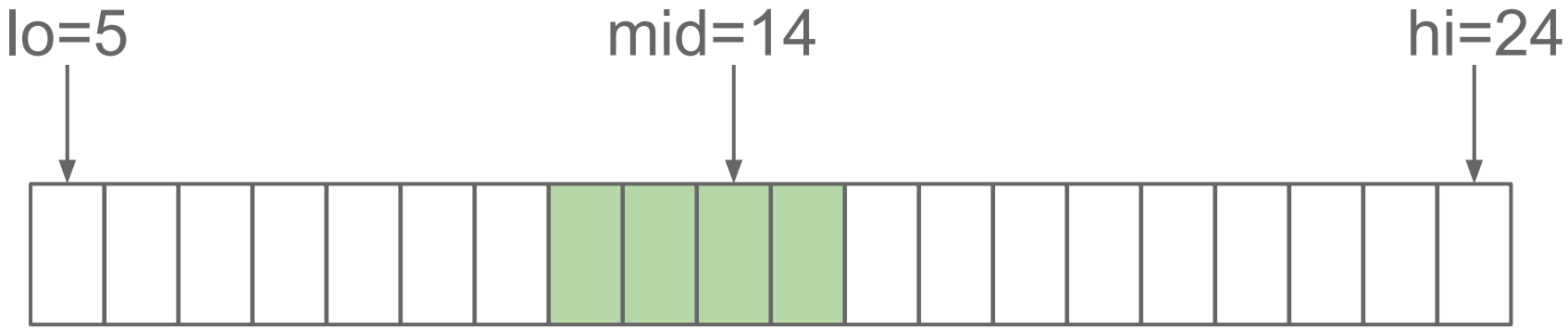
```
Improve(queries, lo, hi):
  if lo == hi: return
  mid = (lo + hi) / 2
  Improve([q in queries where q.b <= mid], lo, mid)
  Improve([q in queries where q.a > mid], mid + 1, hi)

  ImproveViaMid(queries, lo, mid, hi)
```

ImproveViaMid considers all intervals that contain [mid, mid + 1], and are within the [lo, hi] to improve provided queries.

A query participates in O(log(N)) ImproveViaMid calls.

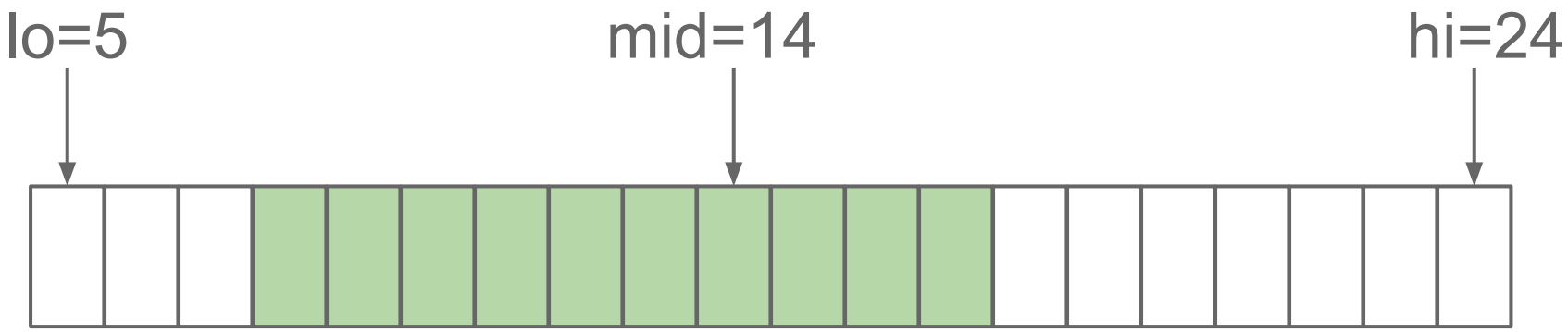


Starting from subsequence [mid, mid + 1], we expand it to the left, storing all intervals we encounter until we exit the [lo, hi] window.



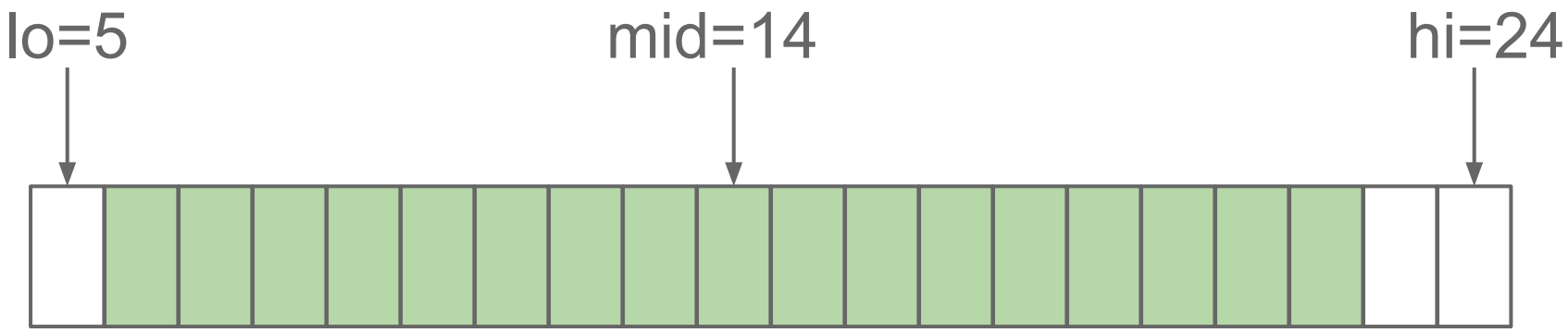
Left intervals: [12, 15]

Starting from subsequence [mid, mid + 1], we expand it to the left, storing all intervals we encounter until we exit the [lo, hi] window.



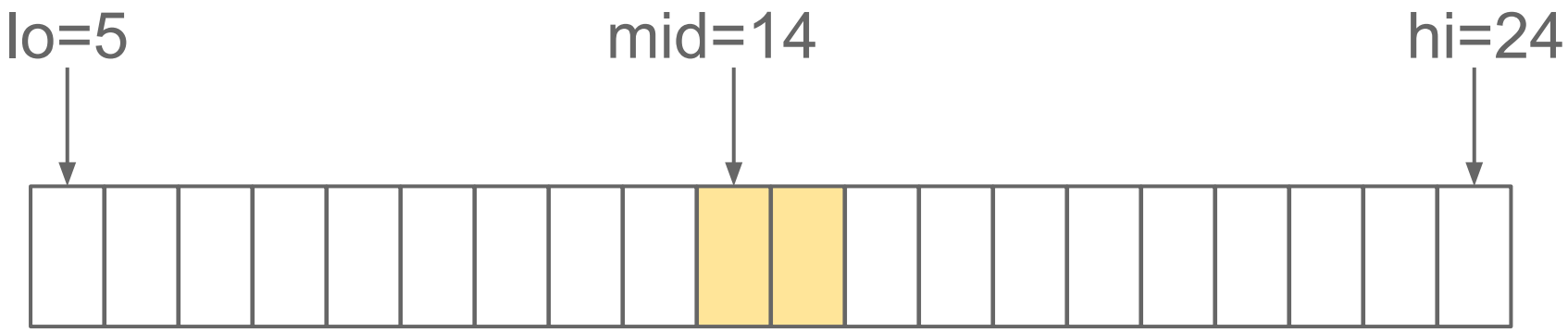
Left intervals: [12, 15], [8, 17]

Starting from subsequence [mid, mid + 1], we expand it to the left, storing all intervals we encounter until we exit the [lo, hi] window.



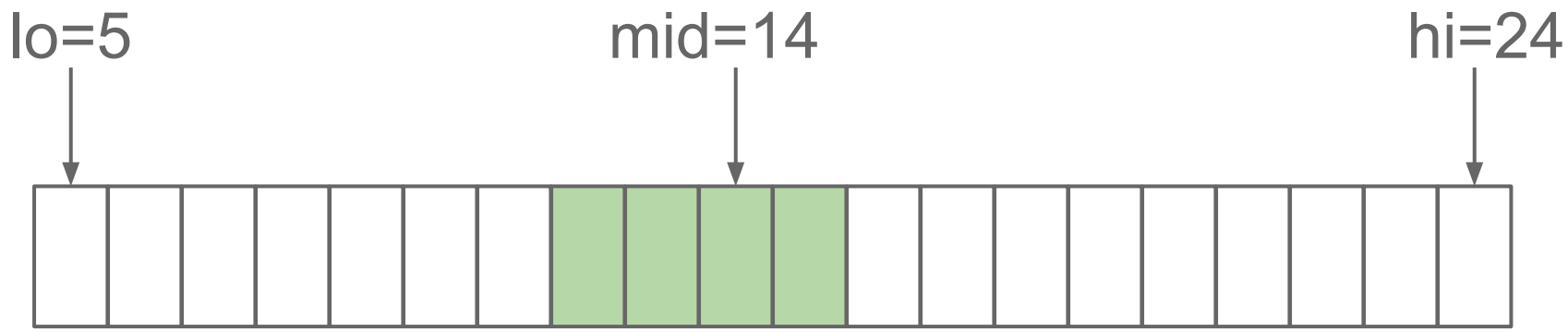
Left intervals: [12, 15], [8, 17], [6, 22]

Starting from subsequence [mid, mid + 1], we expand it to the left, storing all intervals we encounter until we exit the [lo, hi] window.



Left intervals: [12, 15], [8, 17], [6, 22]

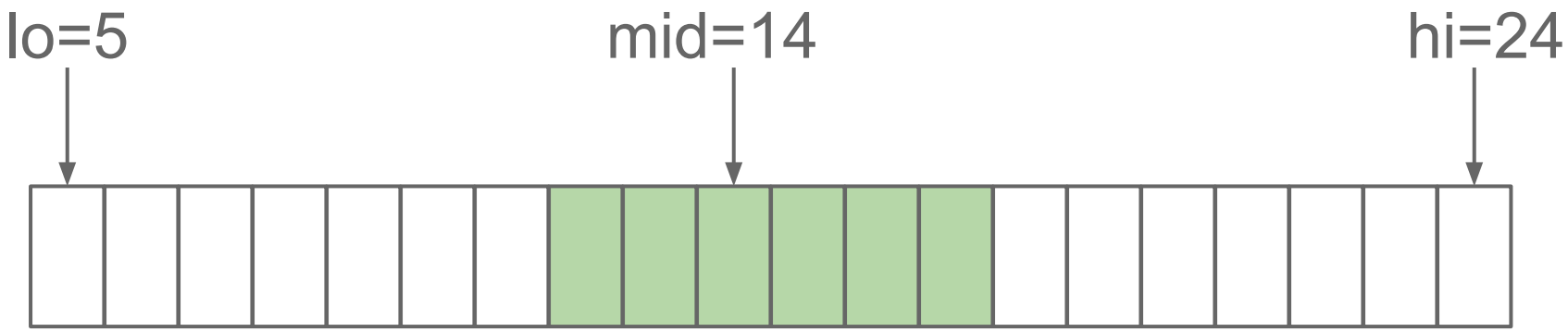
Again, starting from subsequence [mid, mid + 1], we expand it to the right, storing all intervals we encounter until we exit the [lo, hi] window.



Left intervals: [12, 15], [8, 17], [6, 22]

Right intervals: [12, 15]

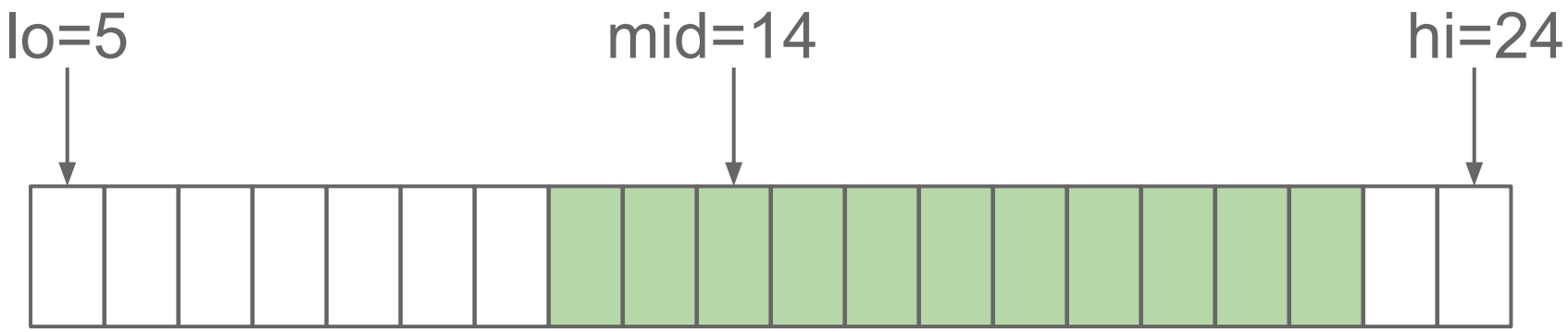
Again, starting from subsequence [mid, mid + 1], we expand it to the right, storing all intervals we encounter until we exit the [lo, hi] window.



Left intervals: [12, 15], [8, 17], [6, 22]

Right intervals: [12, 15], [12, 17]

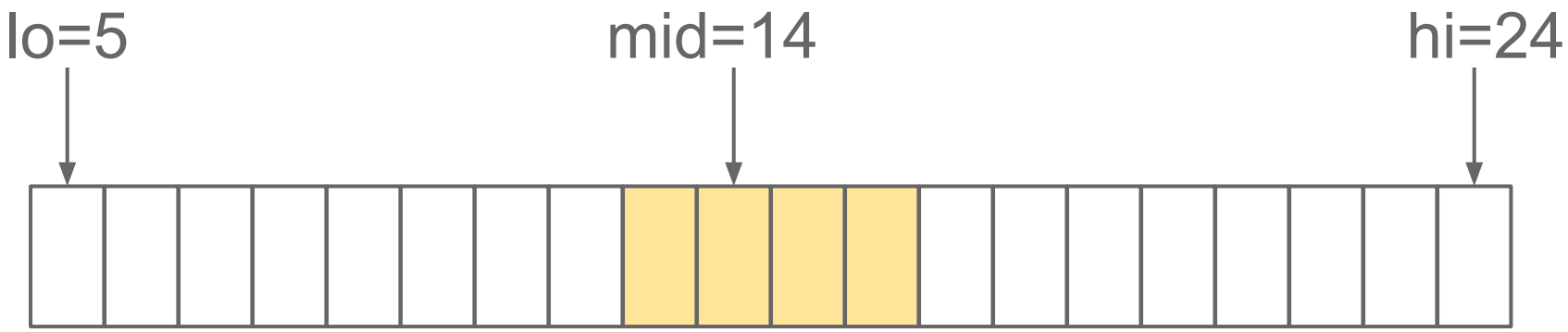
Again, starting from subsequence [mid, mid + 1], we expand it to the right, storing all intervals we encounter until we exit the [lo, hi] window.



Left intervals: [12, 15], [8, 17], [6, 22]

Right intervals: [12, 15], [12, 17], [12, 22]

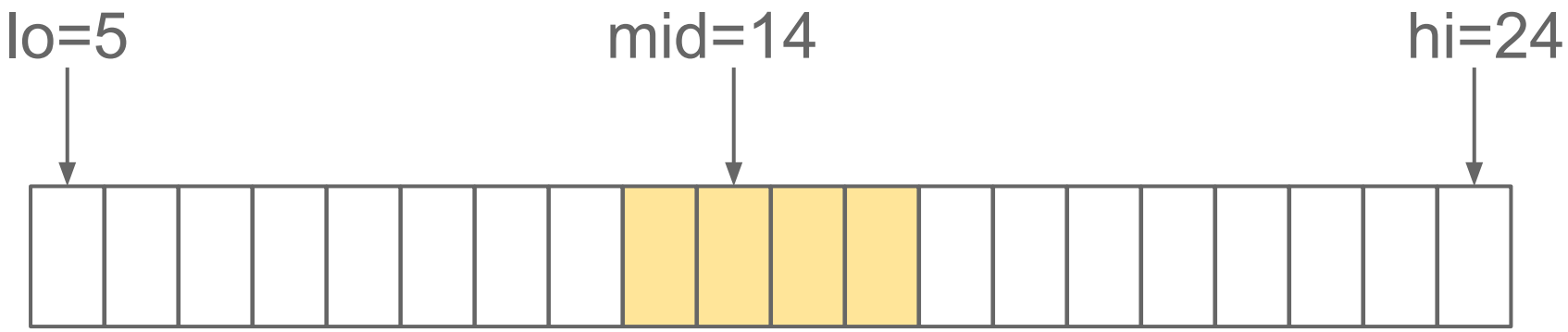
Again, starting from subsequence [mid, mid + 1], we expand it to the right, storing all intervals we encounter until we exit the [lo, hi] window.



Left intervals: [12, 15], **[8, 17]**, [6, 22]

Right intervals: [12, 15], **[12, 17]**, [12, 22]

Finally, for each query [a, b] we find the smallest left interval that contains it and the smallest right interval that contains it. The union of these two intervals is the smallest interval within [lo, hi] that contains the query.



Left intervals: [12, 15], **[8, 17]**, [6, 22]

Right intervals: [12, 15], **[12, 17]**, [12, 22]

We can implement ImproveViaMid(queries, lo, mid, hi) in O(|hi - lo| + queries.size()), for overall complexity of O((N + Q) log N).