Introduction to Algorithms

Meng-Tsung Tsai

10/01/2019

Course Materials

Textbook

Introduction to Algorithms (I2A) 3rd ed. by Cormen, Leiserson, Rivest, and Stein.

Reference Book

Algorithms (JfA) 1st ed. by Erickson. An e-copy can be downloaded from author's website: http://jeffe.cs.illinois.edu/teaching/algorithms/

Websites

http://e3new.nctu.edu.tw for slides, written assignments, and solutions.

http://oj.nctu.me for programming assignments.

Office Hours

Lecturer's

On Wednesdays 16:30 - 17:20 at EC 336 (工程三館).

TA. Erh-Hsuan Lu (呂爾軒) and Tsung-Ta Wu (吳宗達)

On Mondays 10:10 - 11:00 at ES 724 (電資大樓).

TA. Yung-Ping Wang (王詠平) and Chien-An Yu (俞建安)

On Thursdays 11:10 - 12:00 at ES 724 (電資大樓).

Announcements

Programming Assignment 1 is due by Oct 9, 23:59. at https://oj.nctu.me

Written Assignment 1 is due by Oct 15, 10:20. at https://e3new.nctu.edu.tw

We will not normalize the points that you receive from assignments. 100 points is a perfect score, and extra points are considered as a bonus.

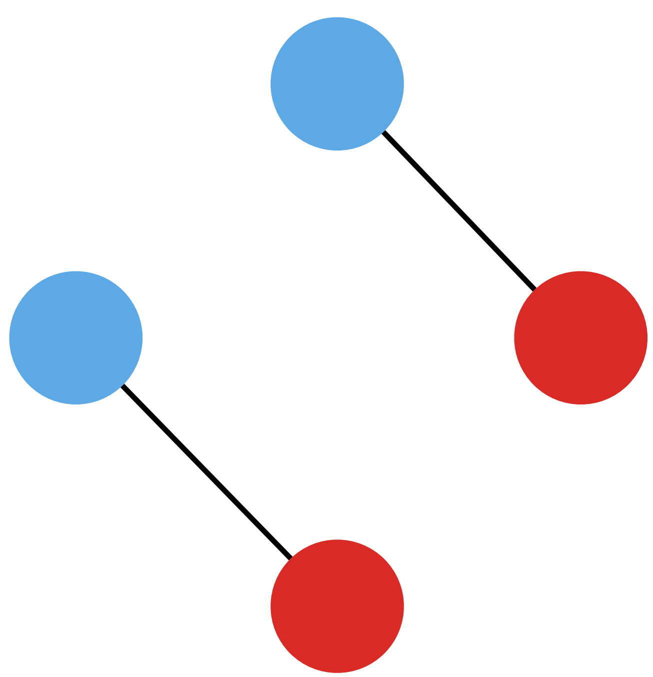
Caution: it is very difficult to solve all problems in an assignment.

Bichromatic Planar Matching

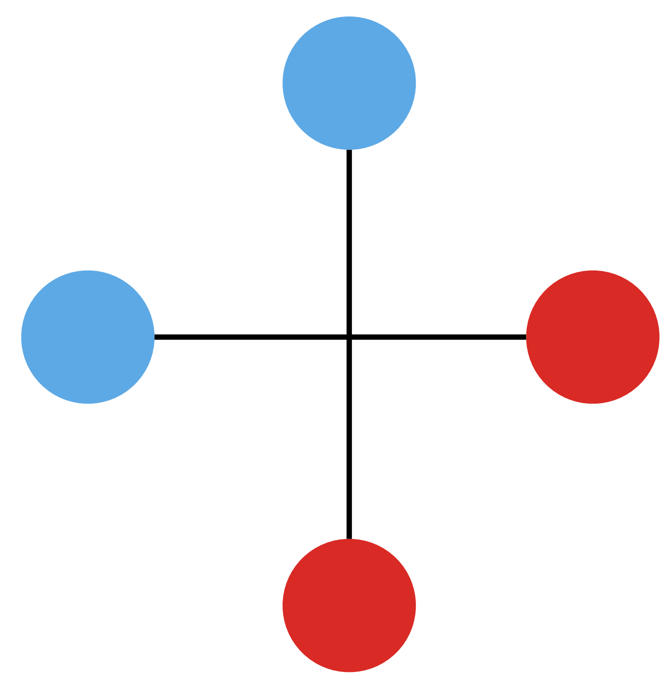
Bichromatic Planar Matching

Input: a set $B = \{b_1, b_2, ..., b_n\}$ of n blue points in the plane and a set $R = \{r_1, r_2, ..., r_n\}$ of n red points in the plane. No 3 points are colinear.

Output: a matching between B and R so that the line segments connecting the matched pairs of points do not intersect.



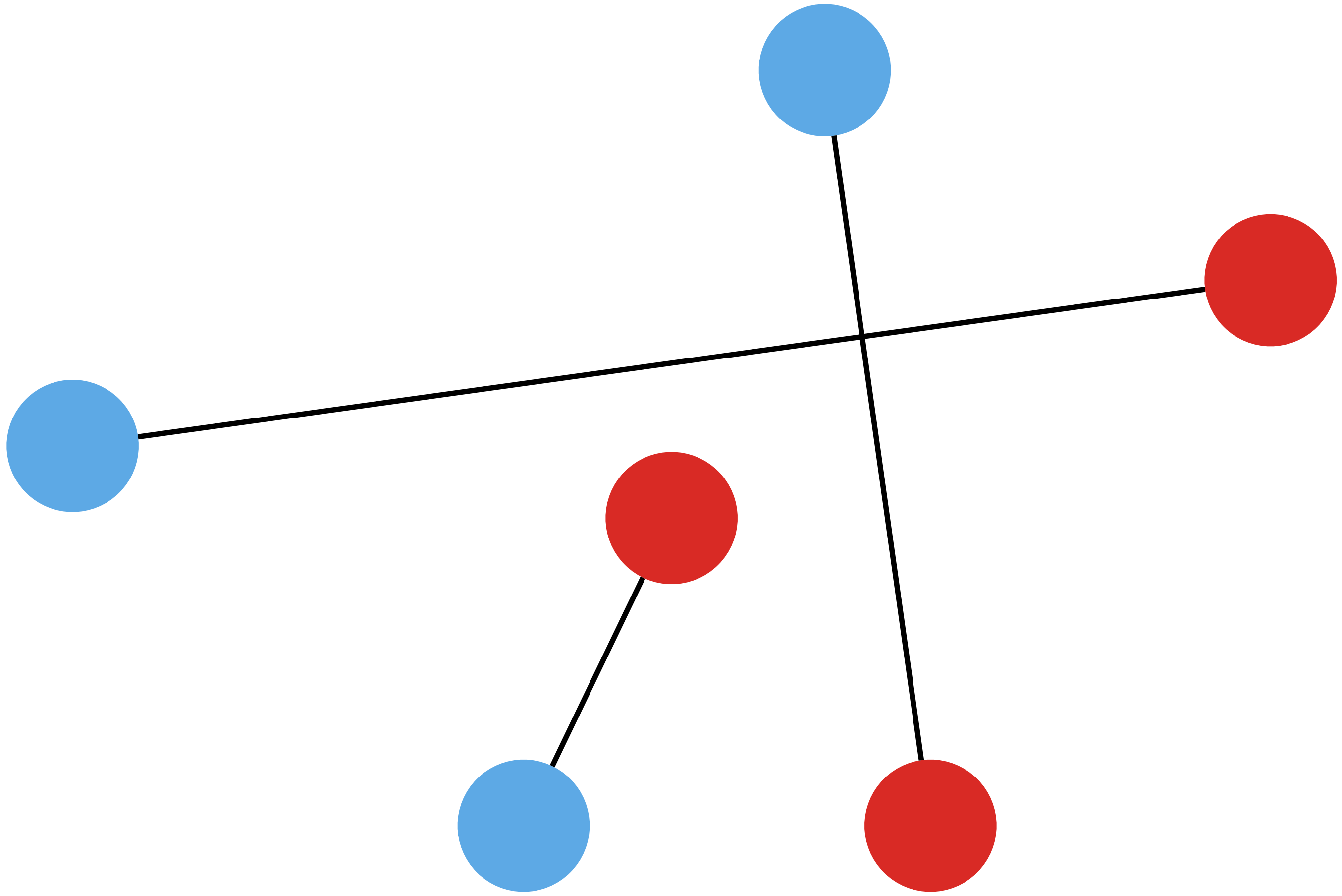
A feasible matching.



An infeasible matching.

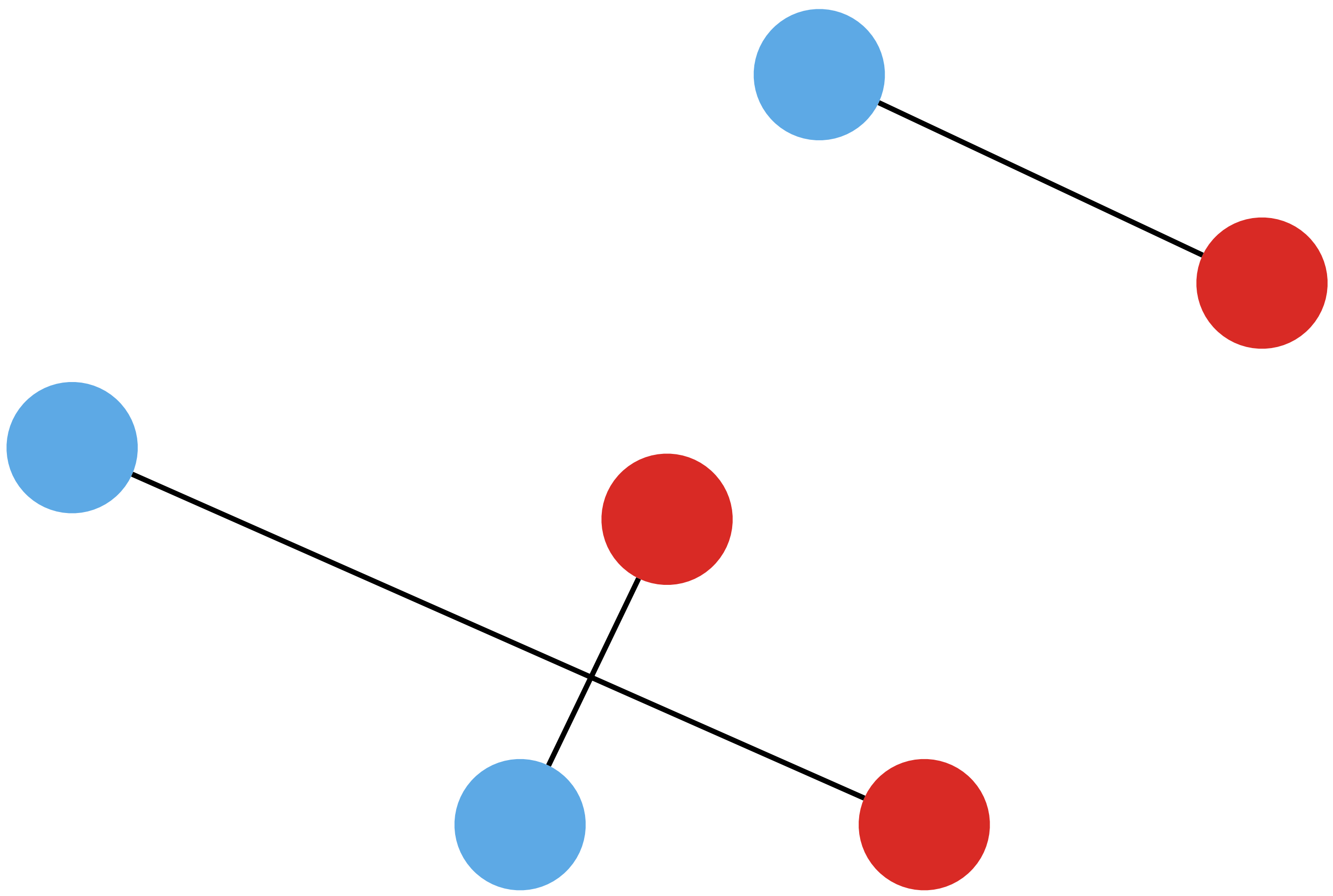
Bichromatic Planar Matching

Plan: arbitrarily assign a matching and resolve the crossing points one by one.



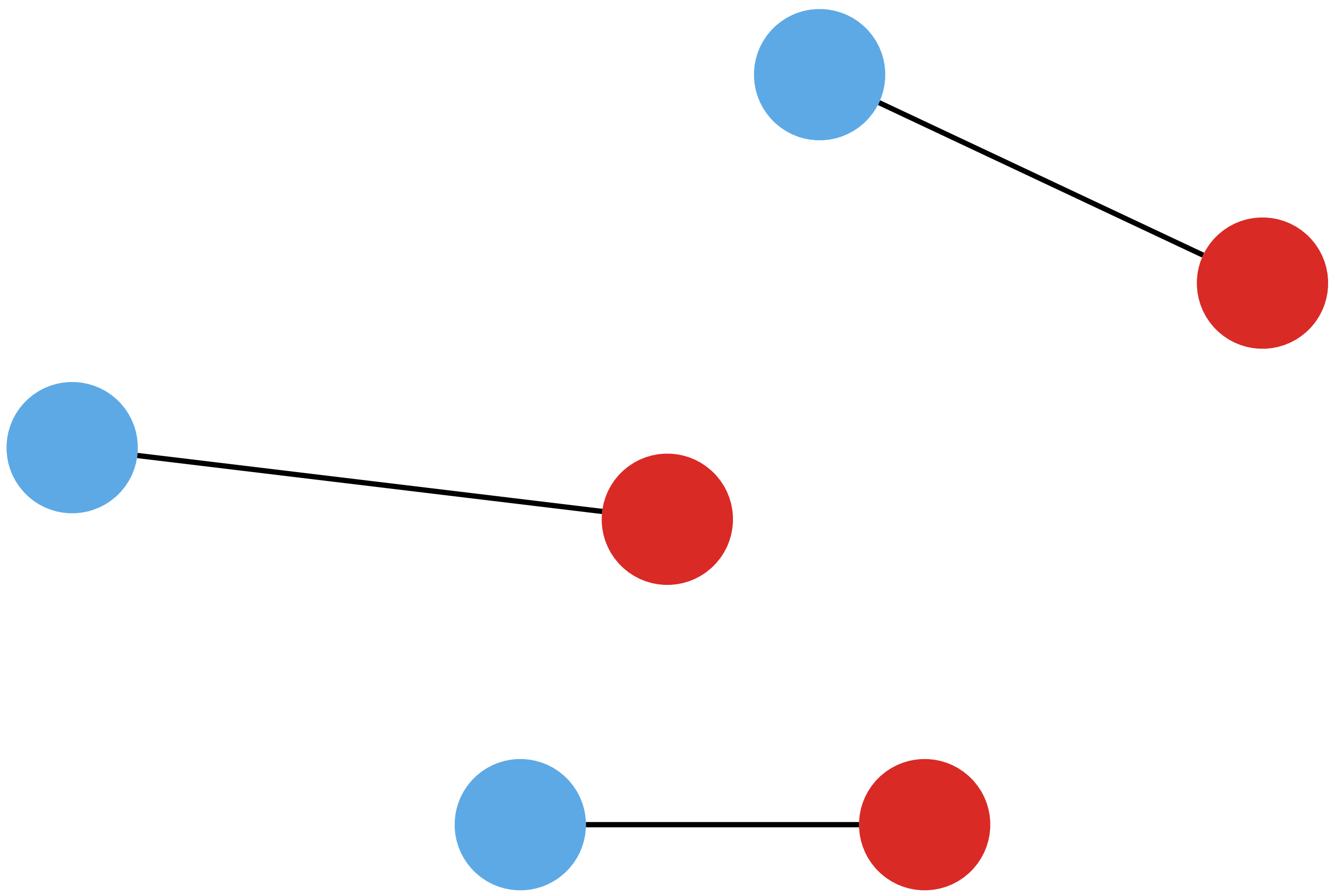
Bichromatic Planar Matching

Plan: arbitrarily assign a matching and resolve the crossing points one by one.



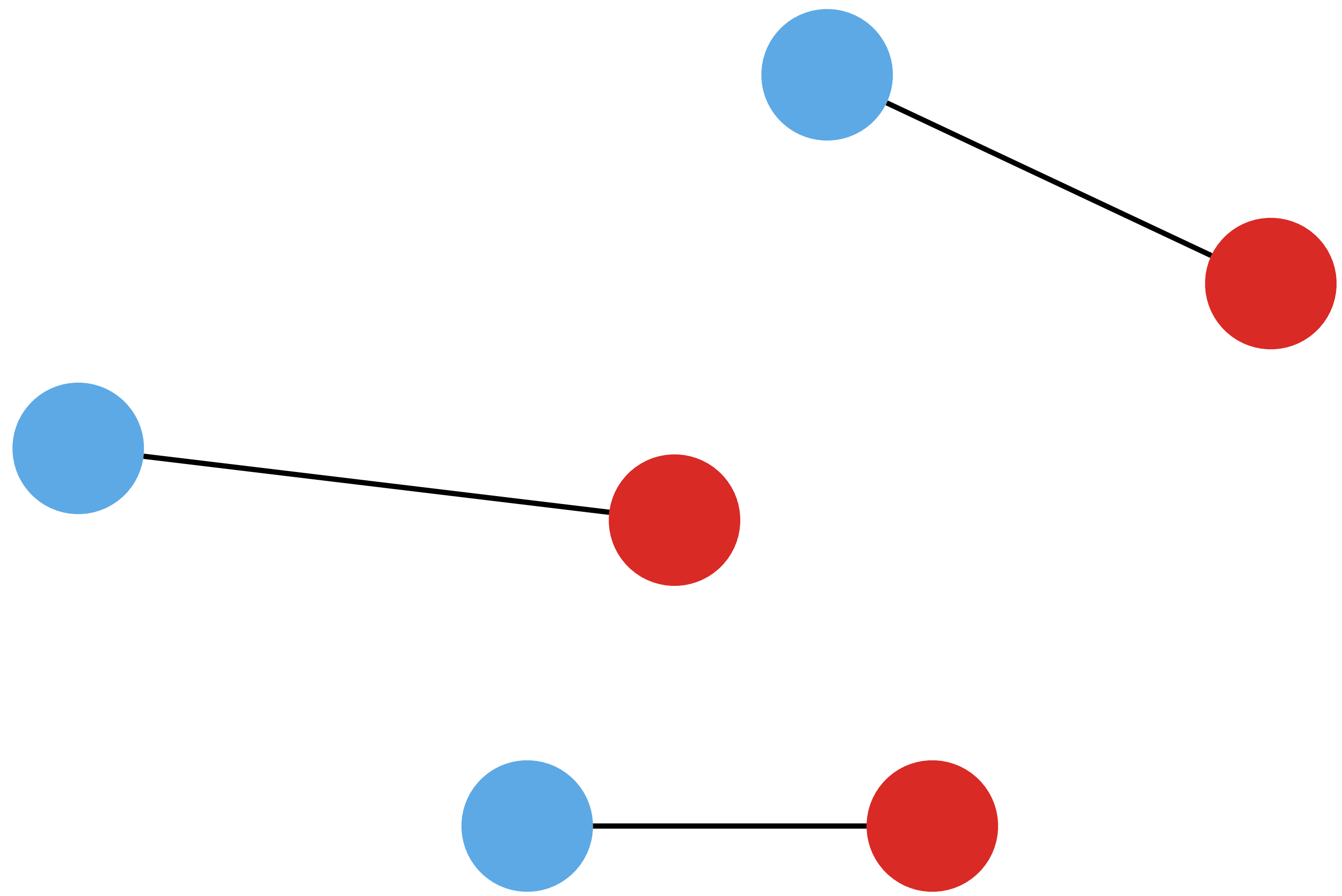
Bichromatic Planar Matching

Plan: arbitrarily assign a matching and resolve the crossing points one by one.



Bichromatic Planar Matching

Plan: arbitrarily assign a matching and resolve the crossing points one by one.



Can we always find a feasible matching by this approach?

Bichromatic Planar Matching

The answer is **Yes**.

Let $\phi(M) = \sum_{(b,\, r) \in M}$ distance(b, r).

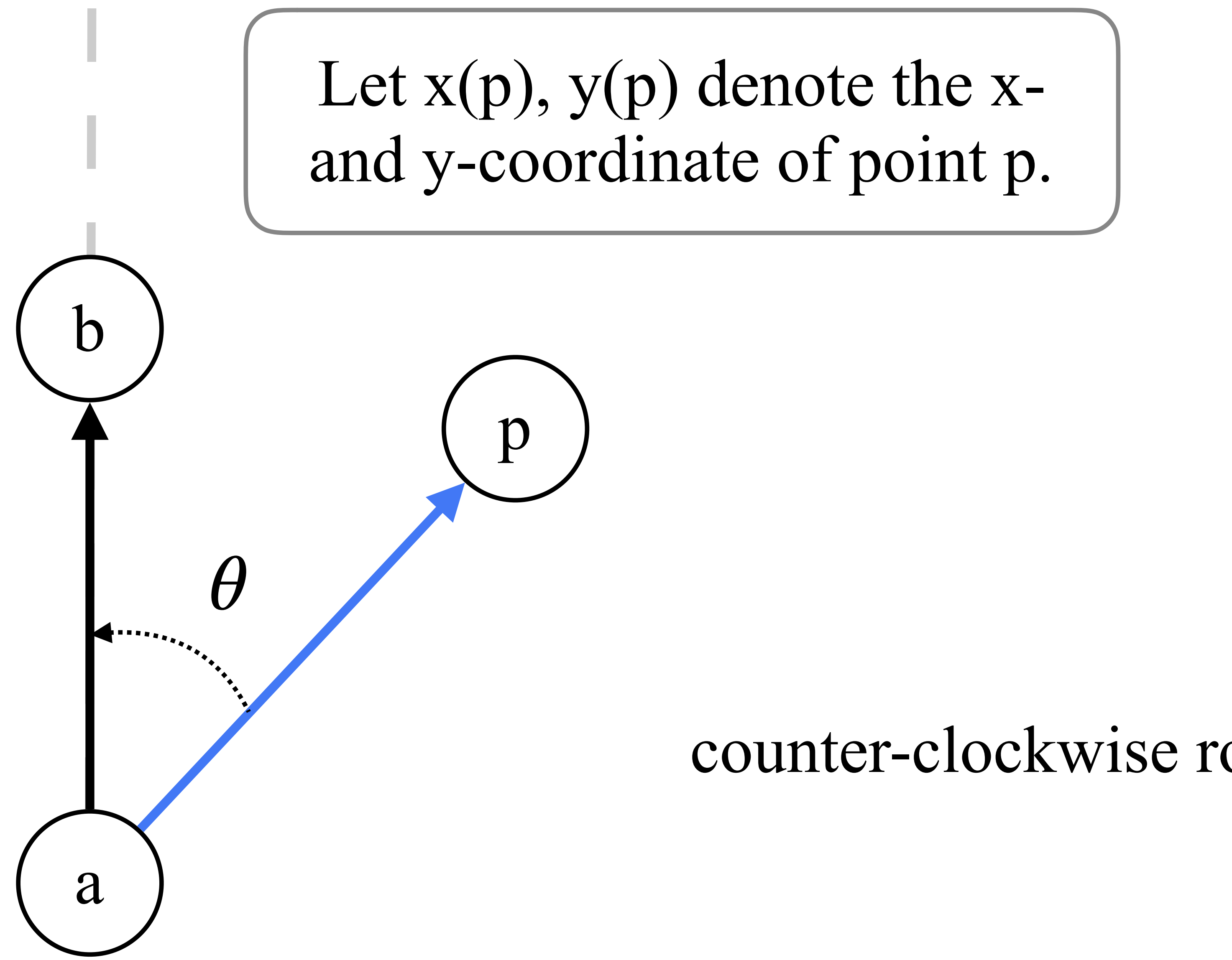
If a matching M is infeasible, then one can find a crossing in M and resolve it, which yields another matching M'. Observe that

$$\phi(M') < \phi(M).$$

There are n! different matchings and iteratively resolving crossing points cannot visit a matching twice. So the proposed algorithm can be done in O(n!) time.

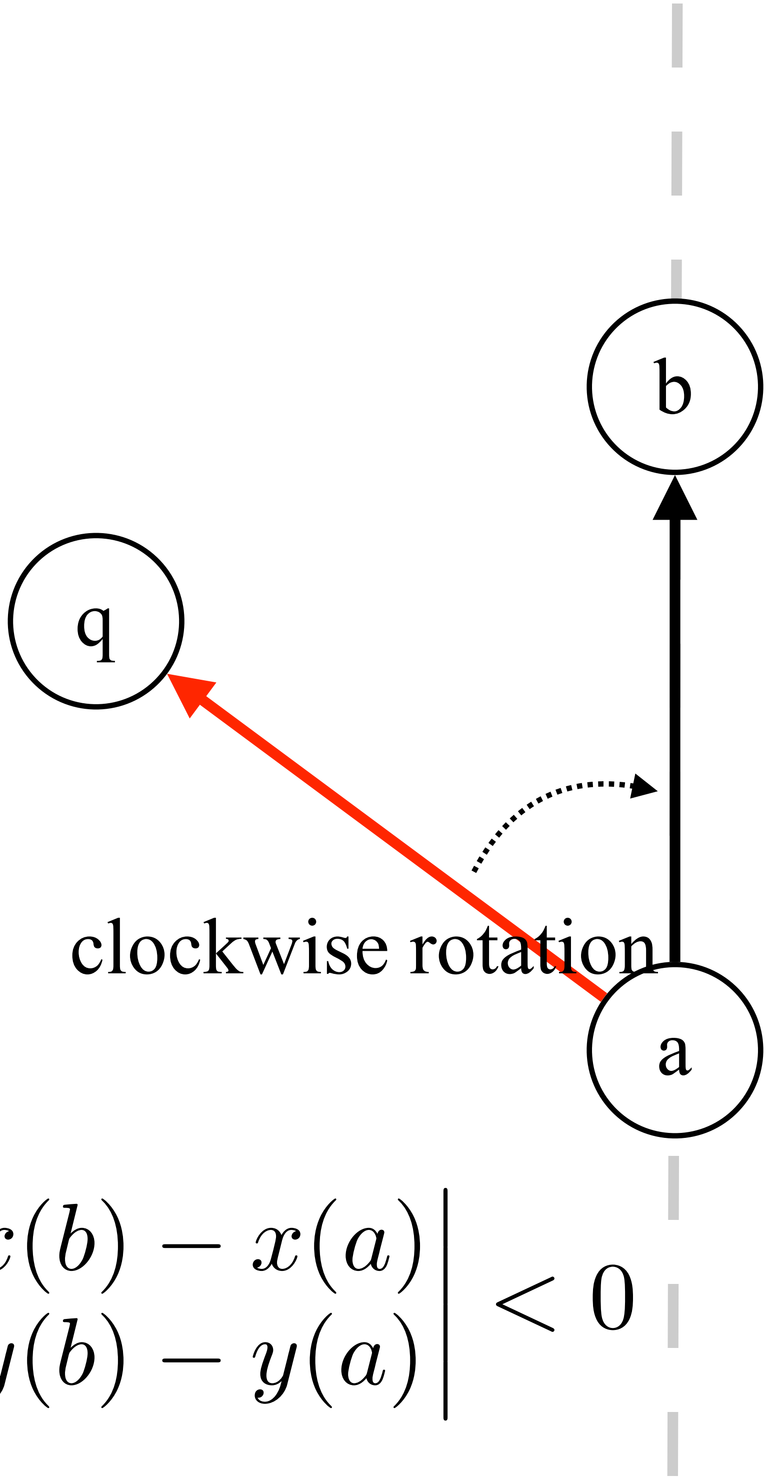
Deciding Whether Two Line Segments Intersect

Direction



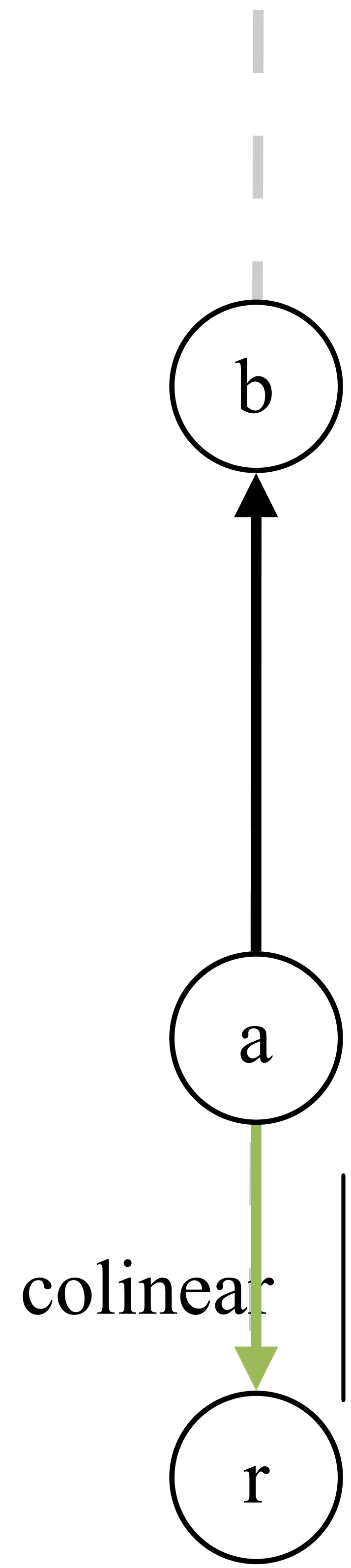
$$\overrightarrow{ap} \times \overrightarrow{ab} = |\overrightarrow{ap}||\overrightarrow{ab}| \sin\theta = \begin{vmatrix} x(p) - x(a) & x(b) - x(a) \\ y(p) - y(a) & y(b) - y(a) \end{vmatrix} > 0$$

Direction



$$\begin{vmatrix} x(\textcolor{red}{q}) - x(a) & x(b) - x(a) \\ y(\textcolor{red}{q}) - y(a) & y(b) - y(a) \end{vmatrix} < 0$$

Direction



$$\begin{vmatrix} x({\color{green}r}) - x(a) & x(b) - x(a) \\ y({\color{green}r}) - y(a) & y(b) - y(a) \end{vmatrix} = 0$$

Direction

int **D(p_i, p_j, p_k)**{

$$\text{return } \overrightarrow{p_i p_k} \times \overrightarrow{p_i p_j};$$

}

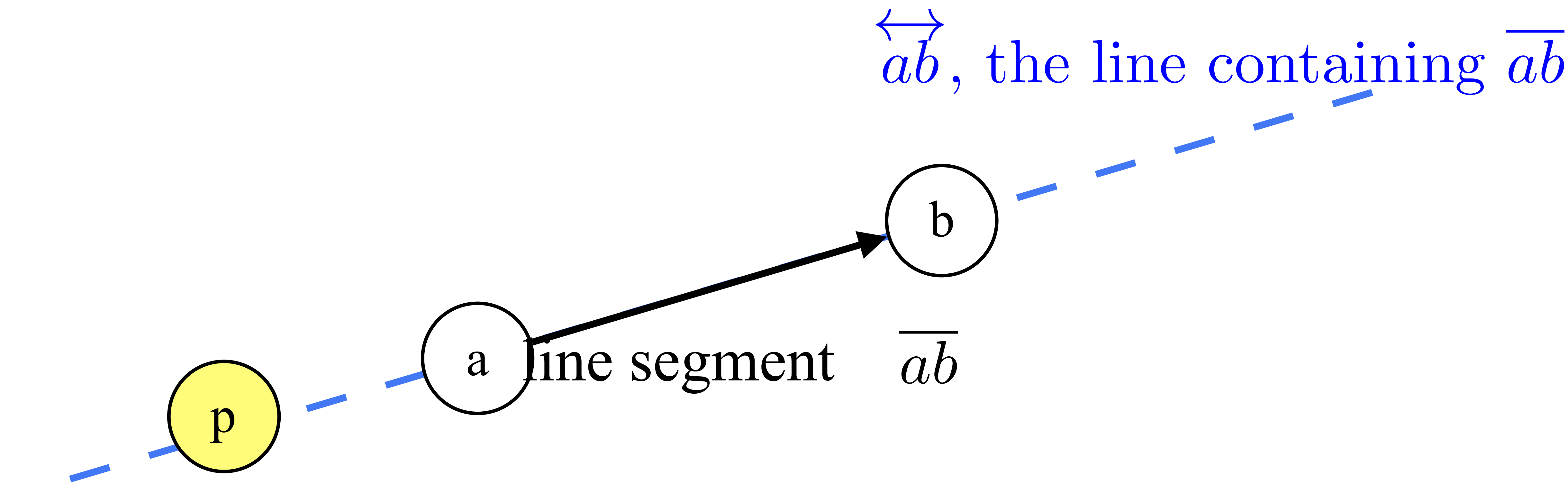
Case 1. If D(p_i, p_j, p_k) > 0, then p_k is in the right halfplane of $\overrightarrow{p_i p_j}$.

Case 2. If D(p_i, p_j, p_k) < 0, then p_k is in the left halfplane of $\overrightarrow{p_i p_j}$.

Case 3. If D(p_i, p_j, p_k) = 0, then p_i, p_j, p_k are colinear.

On a Segment

On-Segment(a, b, p){
    return "Yes" if point p is on the segment $\overline{ab}$
    or otherwise return "No."
}



1. D(a, b, p) = 0; // p on $\overleftrightarrow{ab}$
2. (x(p)-x(a))(x(p)-x(b)) ≤ 0 and (y(p)-y(a))(y(p)-y(b)) ≤ 0

Exercise

1. D(a, b, p) = 0; // p on $\overleftrightarrow{ab}$
2. (x(p)-x(a))(x(p)-x(b)) ≤ 0 and (y(p)-y(a))(y(p)-y(b)) ≤ 0

Comparing the y coordinates here is not redundant. Explain why.

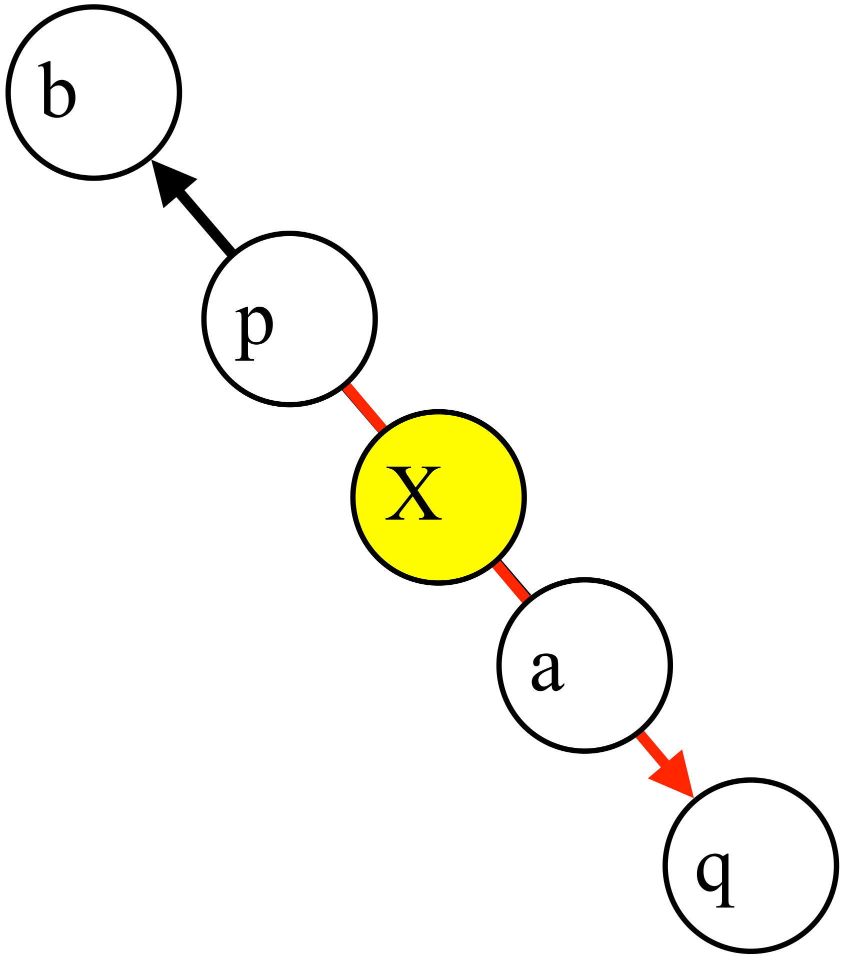
Line segment intersection

Segments-Intersect(a, b, p, q){

return "Yes" if segments $\overline{ab}$ and $\overline{pq}$ intersect at a point X or otherwise return "No."

}

Case 1. If X is not unique, one of {a, b, p, q} is a witness of X.



Line segment intersection

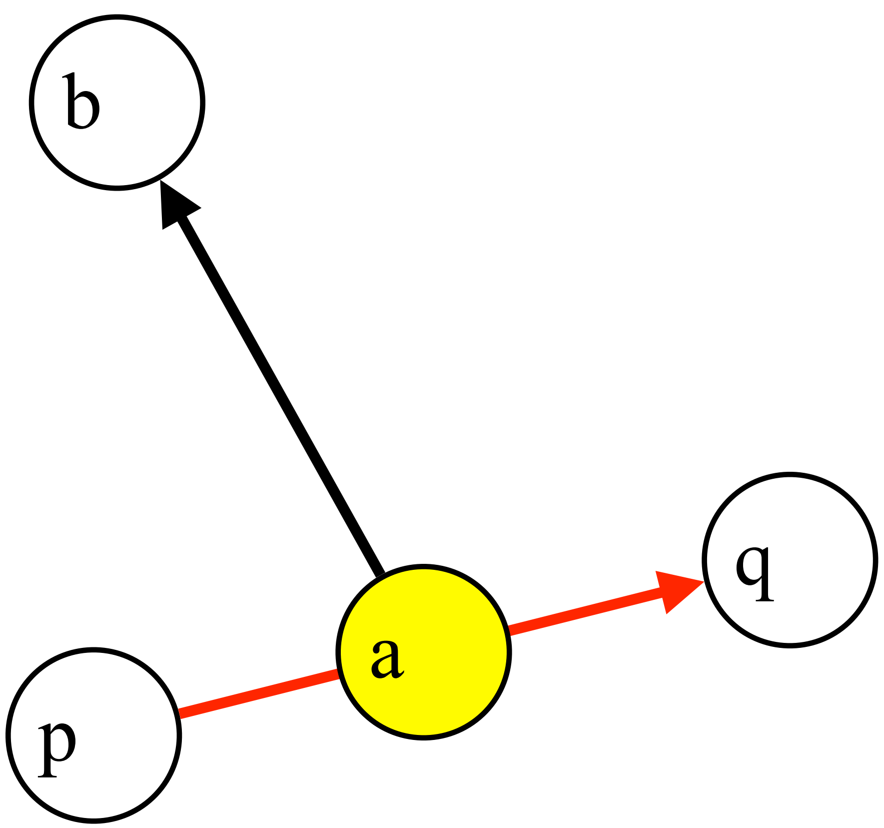
Segments-Intersect(a, b, p, q){

return "Yes" if segments $\overline{ab}$ and $\overline{pq}$ intersect at a point X or otherwise return "No."

}

Case 2. X is unique and X $\in$ {a, b, p, q}.

(1) If On-Segment(p, q, a), output "Yes" because X=a.
(2) If On-Segment(p, q, b), output "Yes" because X=b.
(3) If On-Segment(a, b, p), output "Yes" because X=p.
(4) If On-Segment(a, b, q), output "Yes" becasue X=q.



Line segment intersection

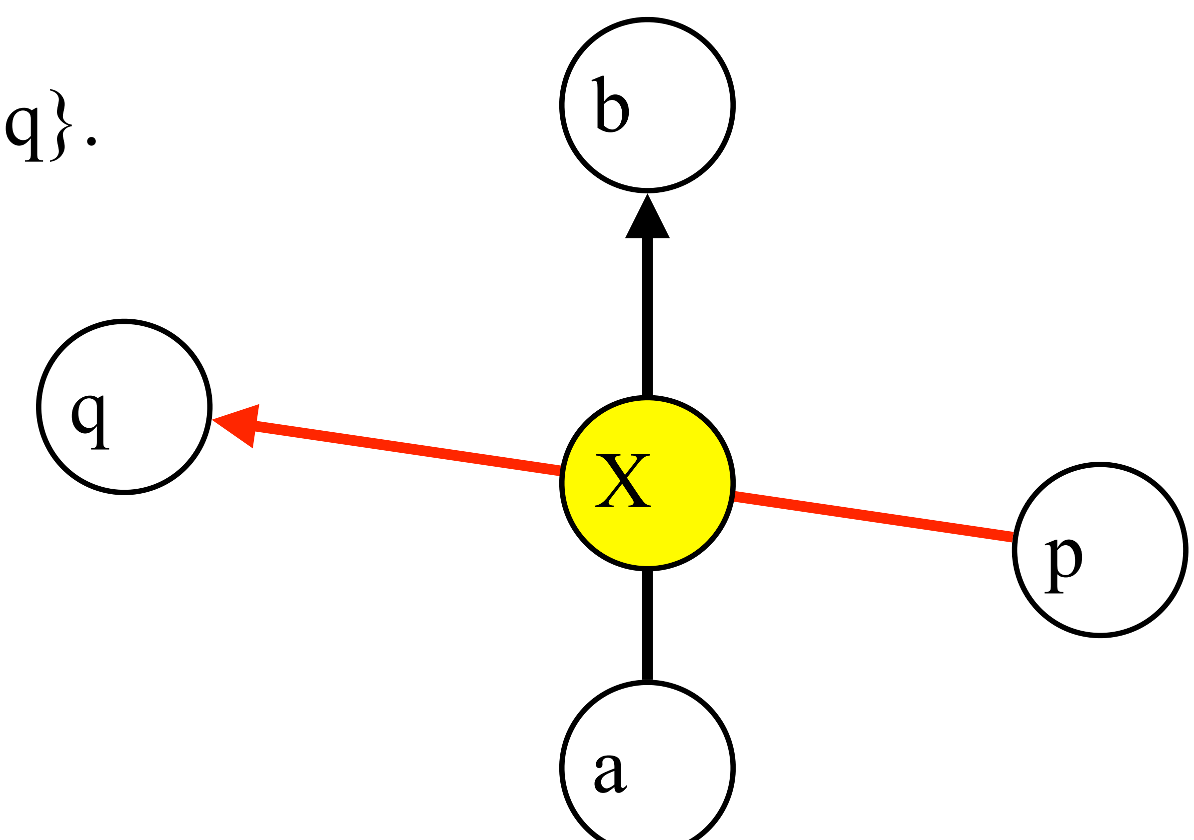
Segments-Intersect(a, b, p, q){

return "Yes" if segments $\overline{ab}$ and $\overline{pq}$ intersect at a point X
or otherwise return "No."

}

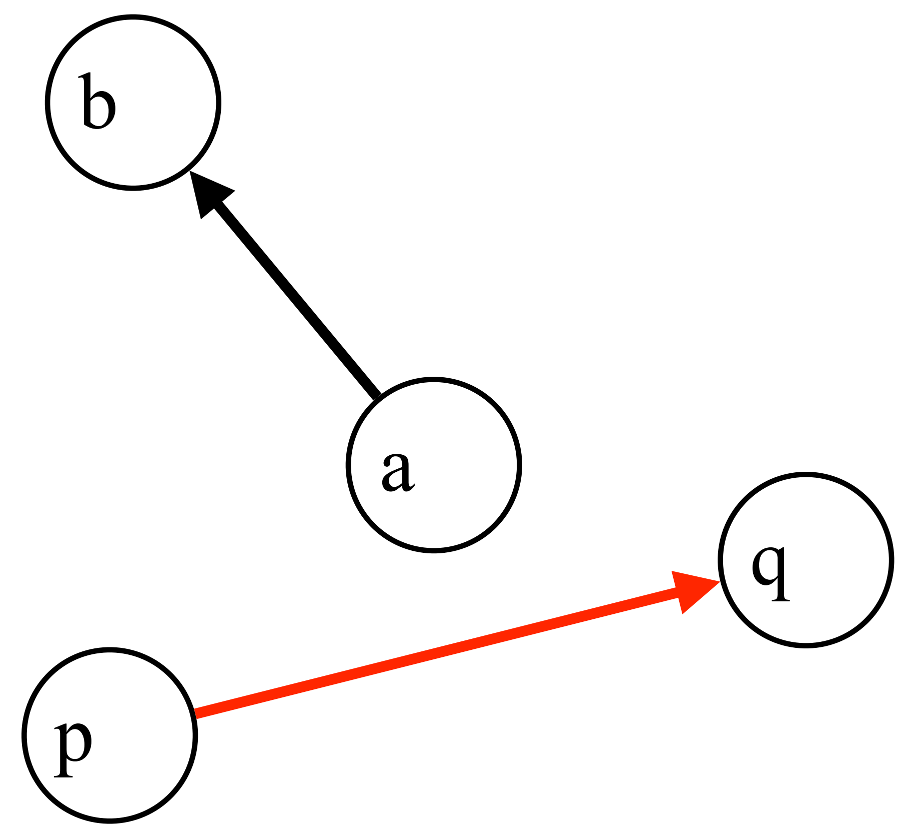
Case 3. X is unique and $X \notin \{a, b, p, q\}$.

(1) D(a, b, p)D(a, b, q) < 0
(2) D(p, q, a)D(p, q, b) < 0



Line segment intersection

Otherwise, there exists no intersection X. Output “No.”

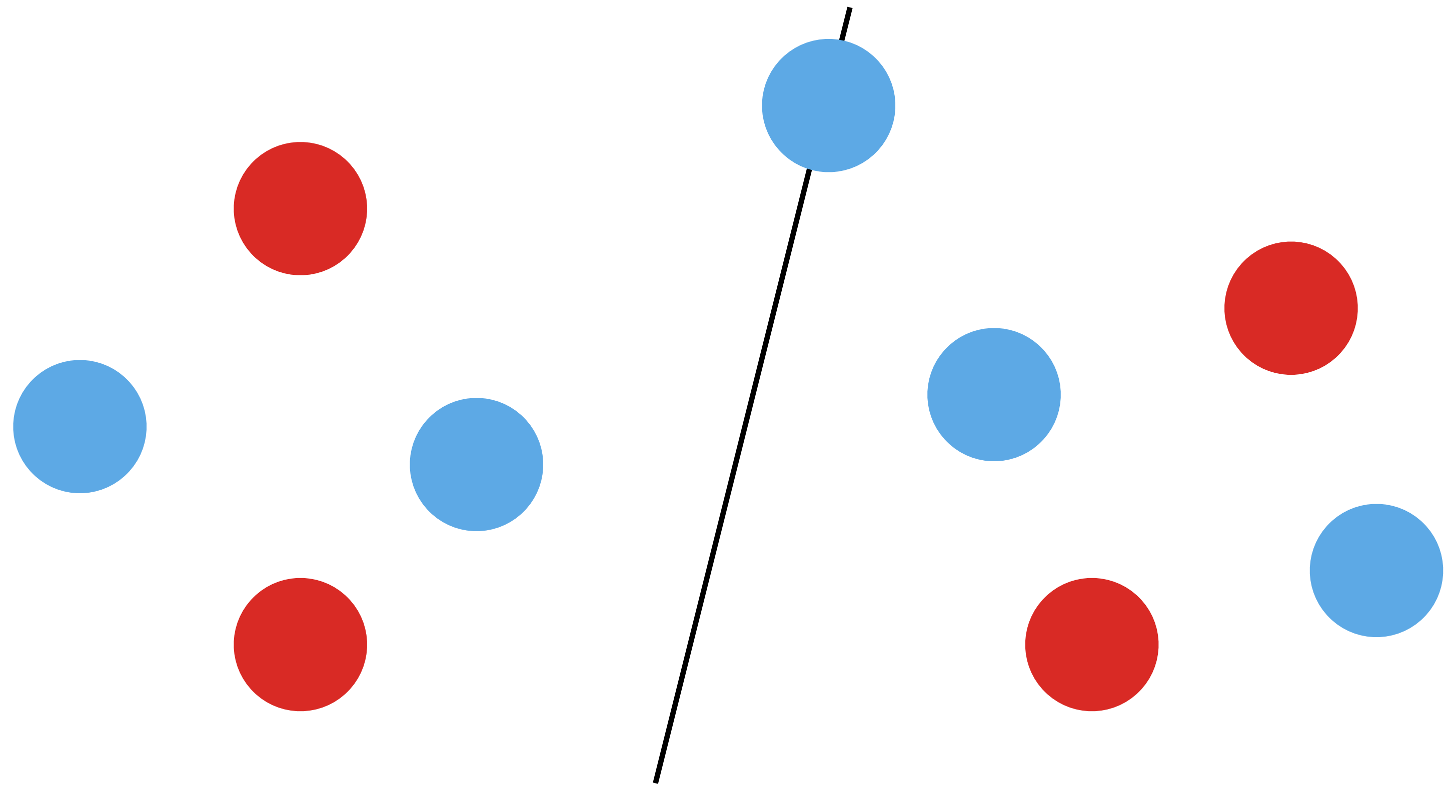


Ham-Sandwich Cut

Ham-Sandwich Cut

Input: a set $B = \{b_1, b_2, ..., b_n\}$ of n blue points in the plane and a set $R = \{r_1, r_2, ..., r_m\}$ of m red points in the plane.

Output: a straight line so that the number of blue points on either side of the line is equal and the number of red points on either side of the line is equal.



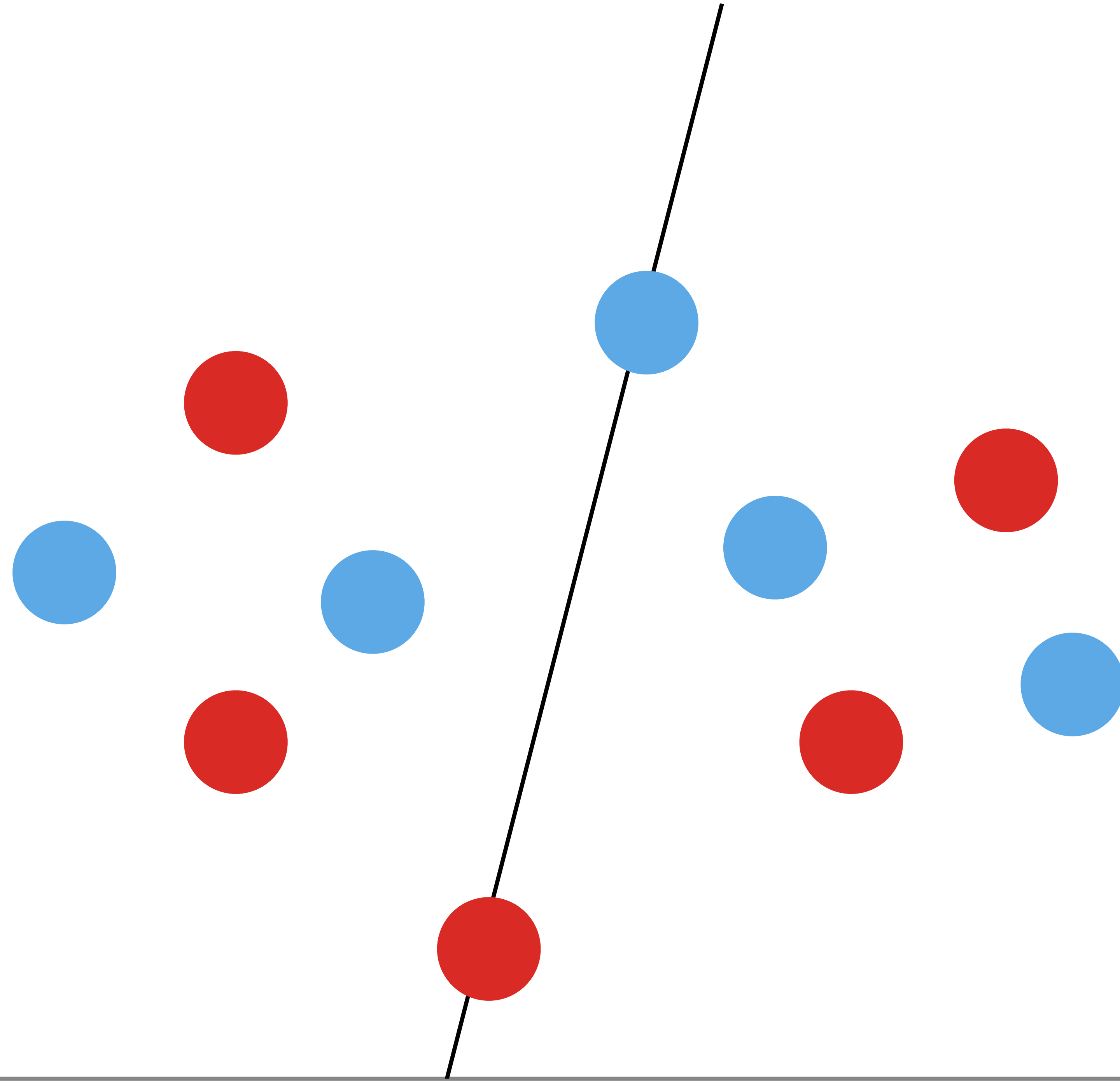
Ham-Sandwich Cut

Input: a set $B = \{b_1, b_2, ..., b_n\}$ of n blue points in the plane and a set $R = \{r_1, r_2, ..., r_m\}$ of m red points in the plane.

Output: a straight line so that the number of blue points on either side of the line is equal and the number of red points on either side of the line is equal.

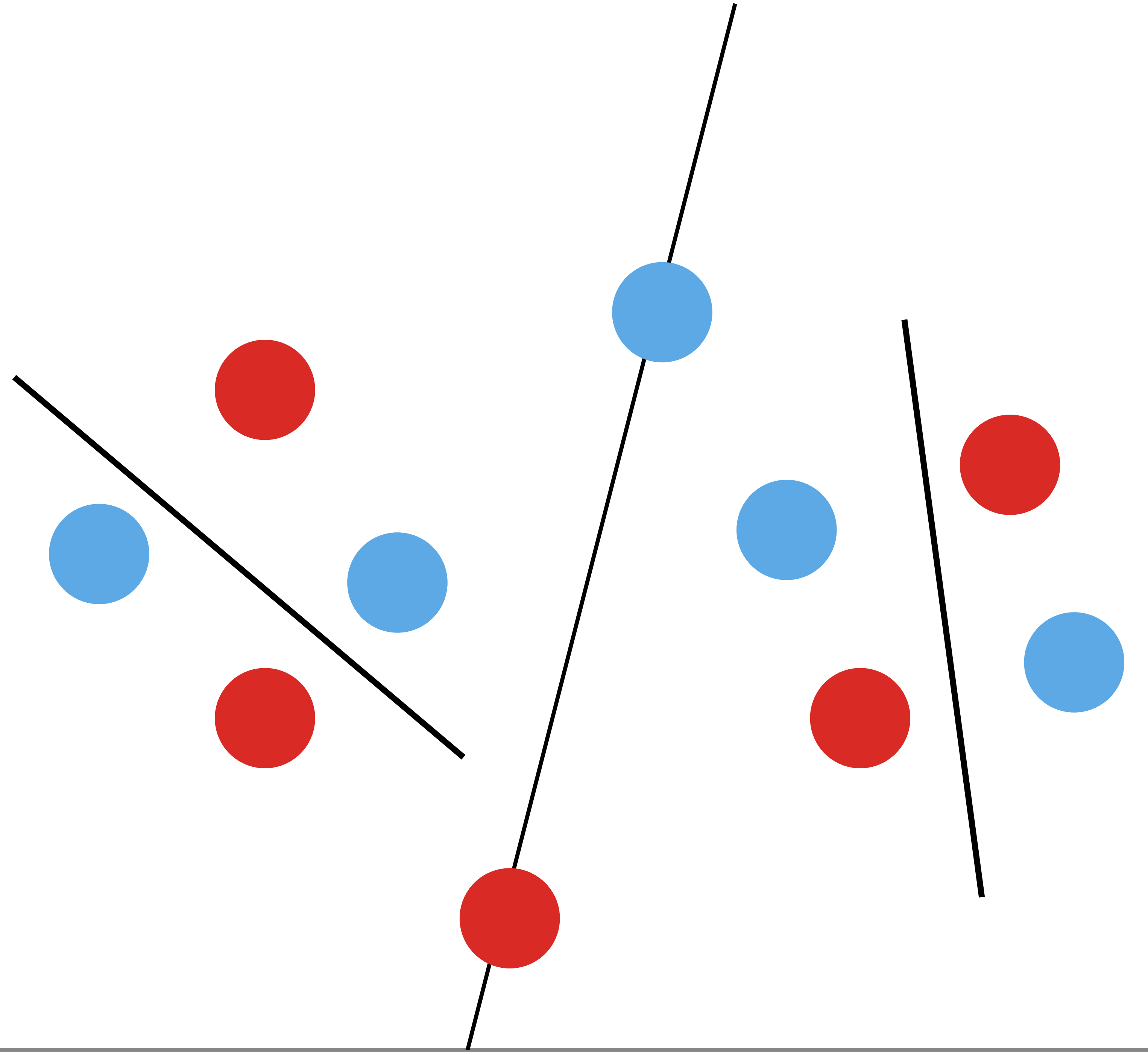
Ham-Sandwich Cut can be found in $O(n+m)$ time.

Solving BMP by Ham-Sandwich Cut



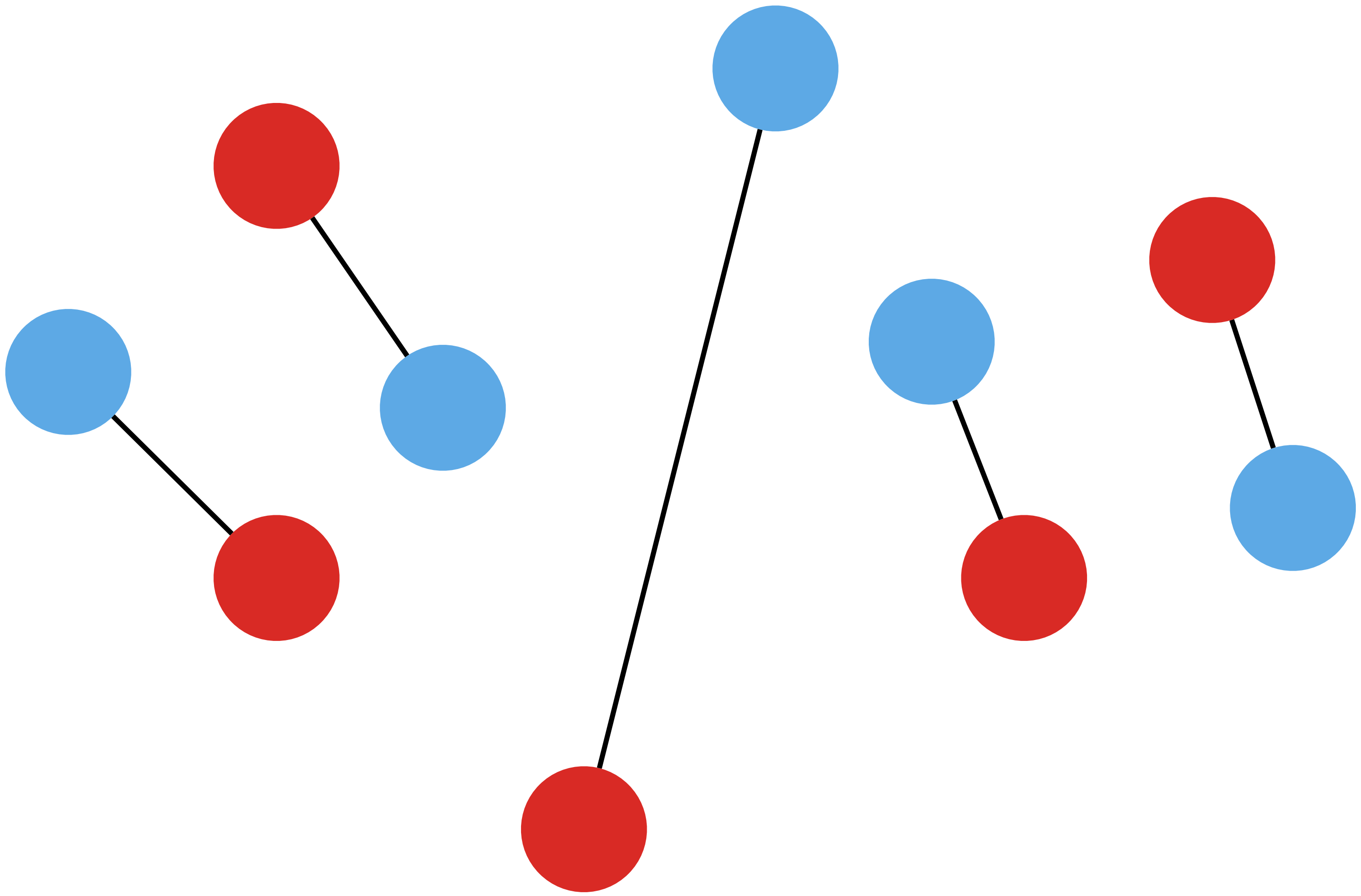
The pairing of points on the left side of the cut cannot interfere with the pairing on the right side.

Solving BMP by Ham-Sandwich Cut



Pair two points if they are both on a seperating line, or a subproblem has exactly the two points as its input.

Solving BMP by Ham-Sandwich Cut



Solving BMP by Ham-Sandwich Cut

$$T(n) = \begin{cases} 2T(\lfloor n/2 \rfloor) + O(n) & \text{if } n > 1 \\ O(1) & \text{if } n = 1 \end{cases}$$

By Master Theorem, T(n) = O(n log n).

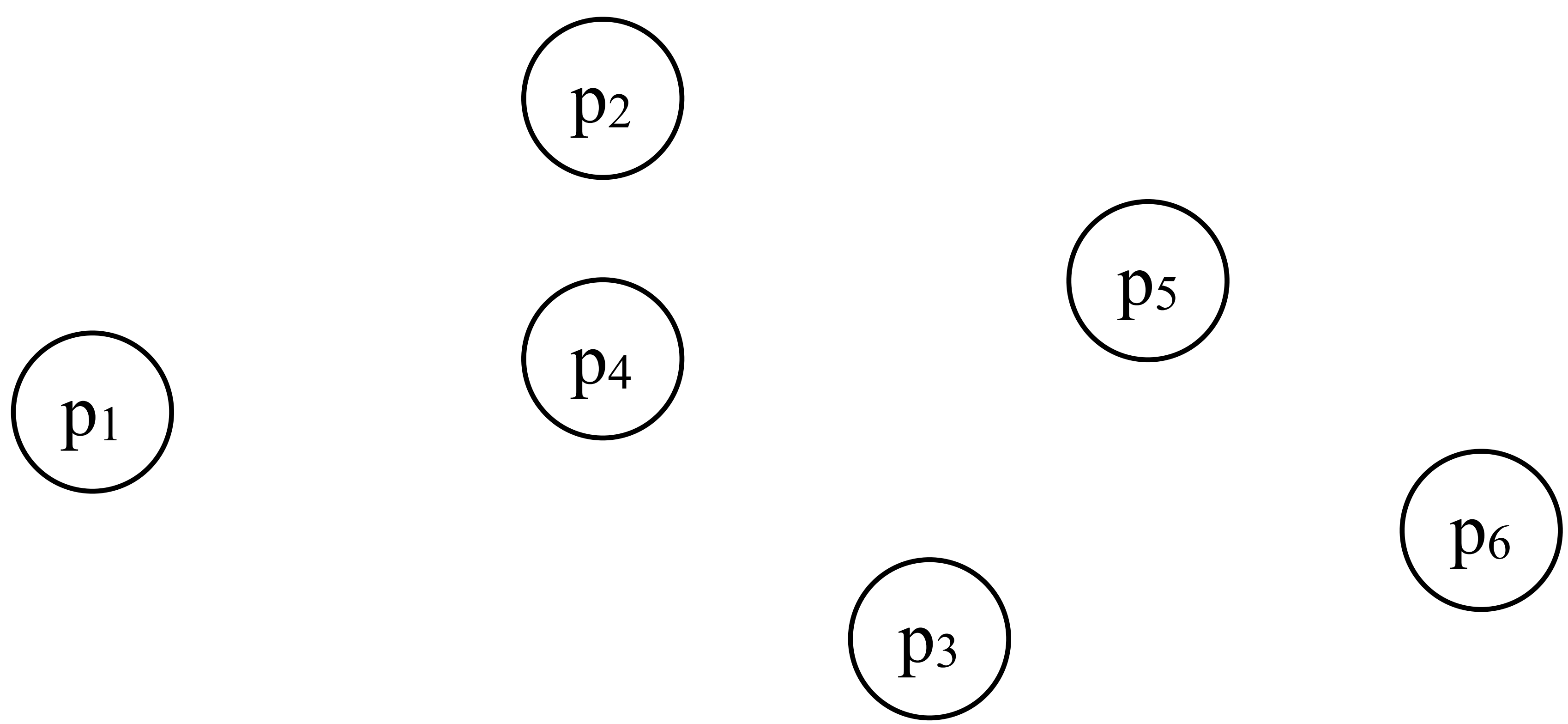
The Closest Pair of Points

The Closest Pair of Points

Input: a set S of n points in the plane, where $S = \{p_1, p_2, \ldots, p_n\}$.

Output: the minimum euclidean distance among all (p_i, p_j) pairs for $1 \leq i < j \leq n$.

Example.

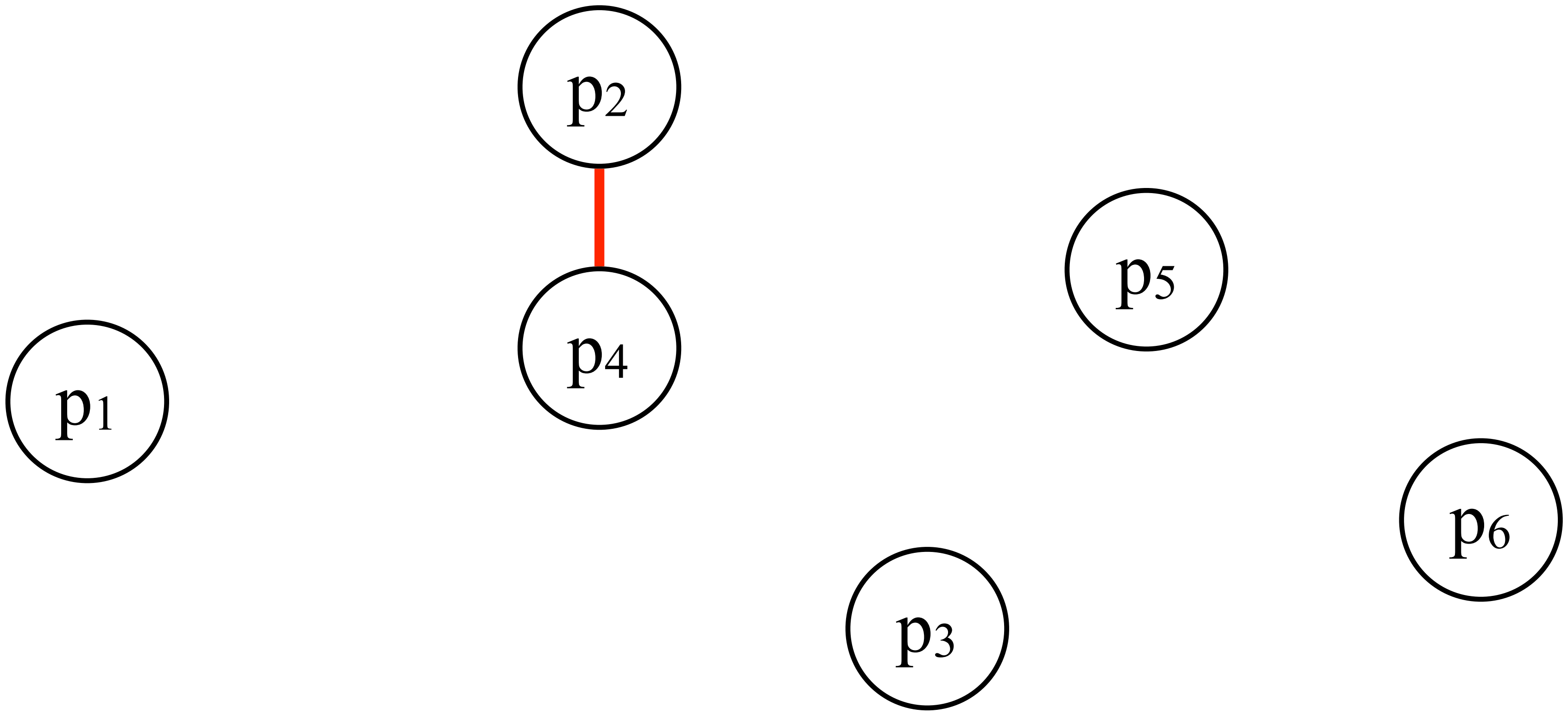


The Closest Pair of Points

Input: a set S of n points in the plane, where S = $\{p_1, p_2, \ldots, p_n\}$.

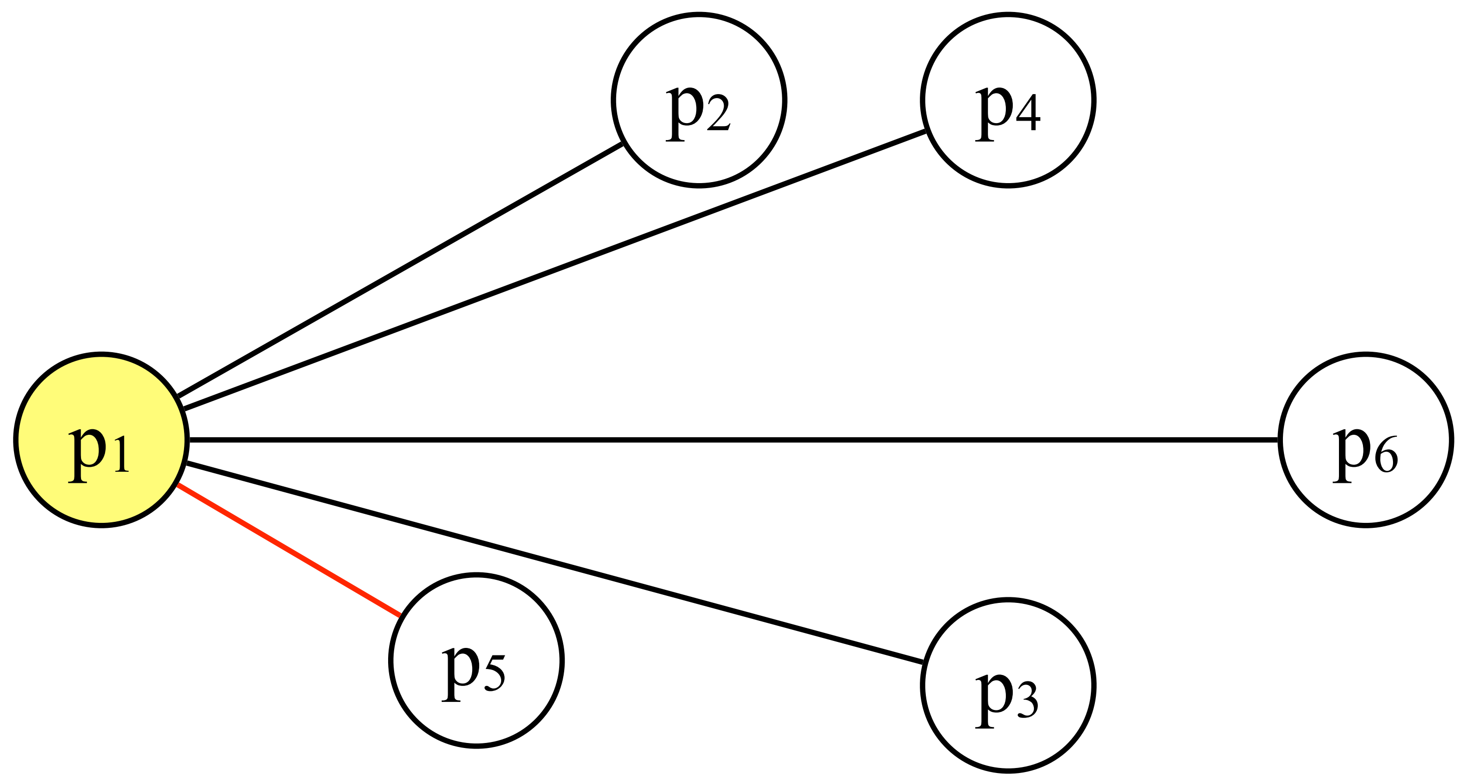
Output: the minimum euclidean distance among all (p_i, p_j) pairs for $1 \le i < j \le n$.

Example.



The Closest Pair of Points

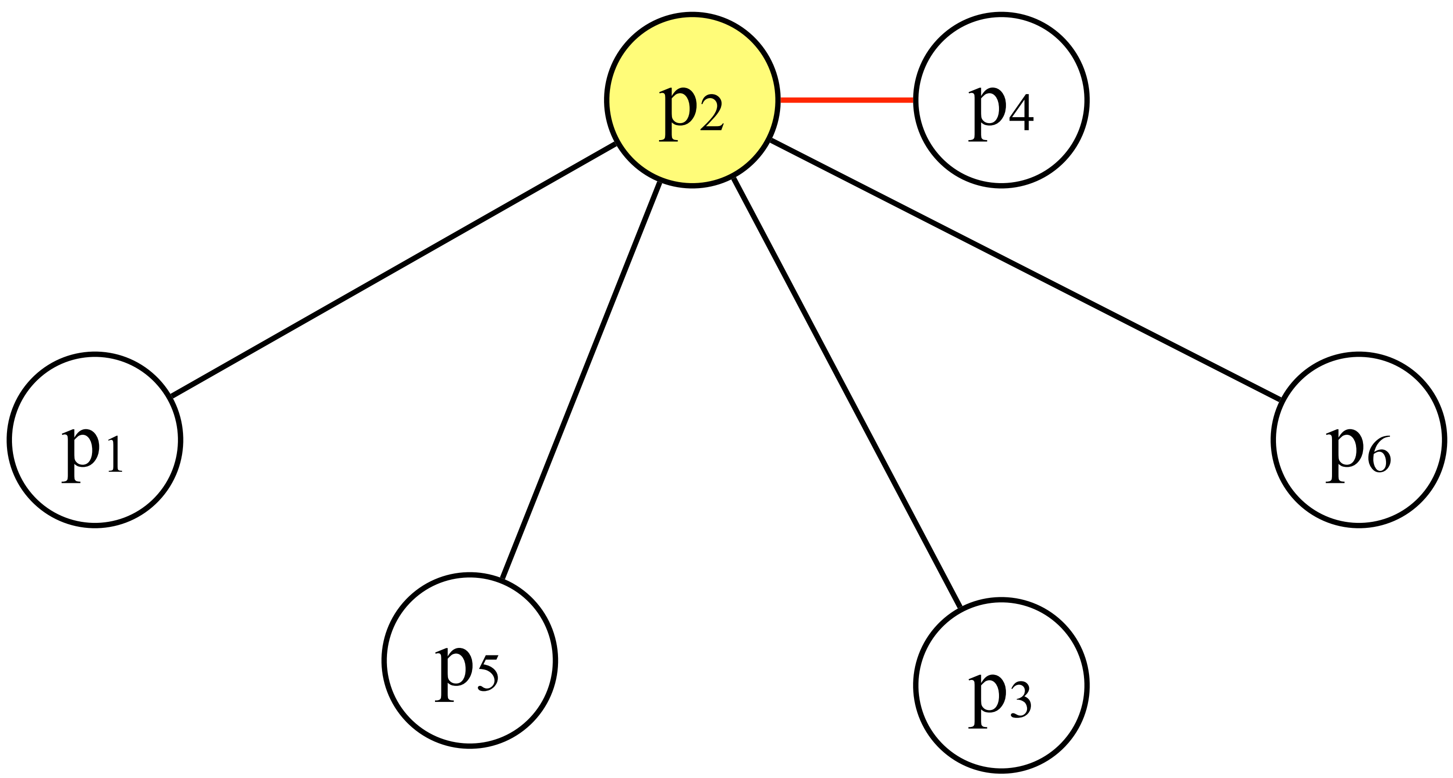
We can solve CP(S) by calculating all pairwise distance. This approach needs $O(n^2)$ time.



Guess one point in the closest pair is p_1.

The Closest Pair of Points

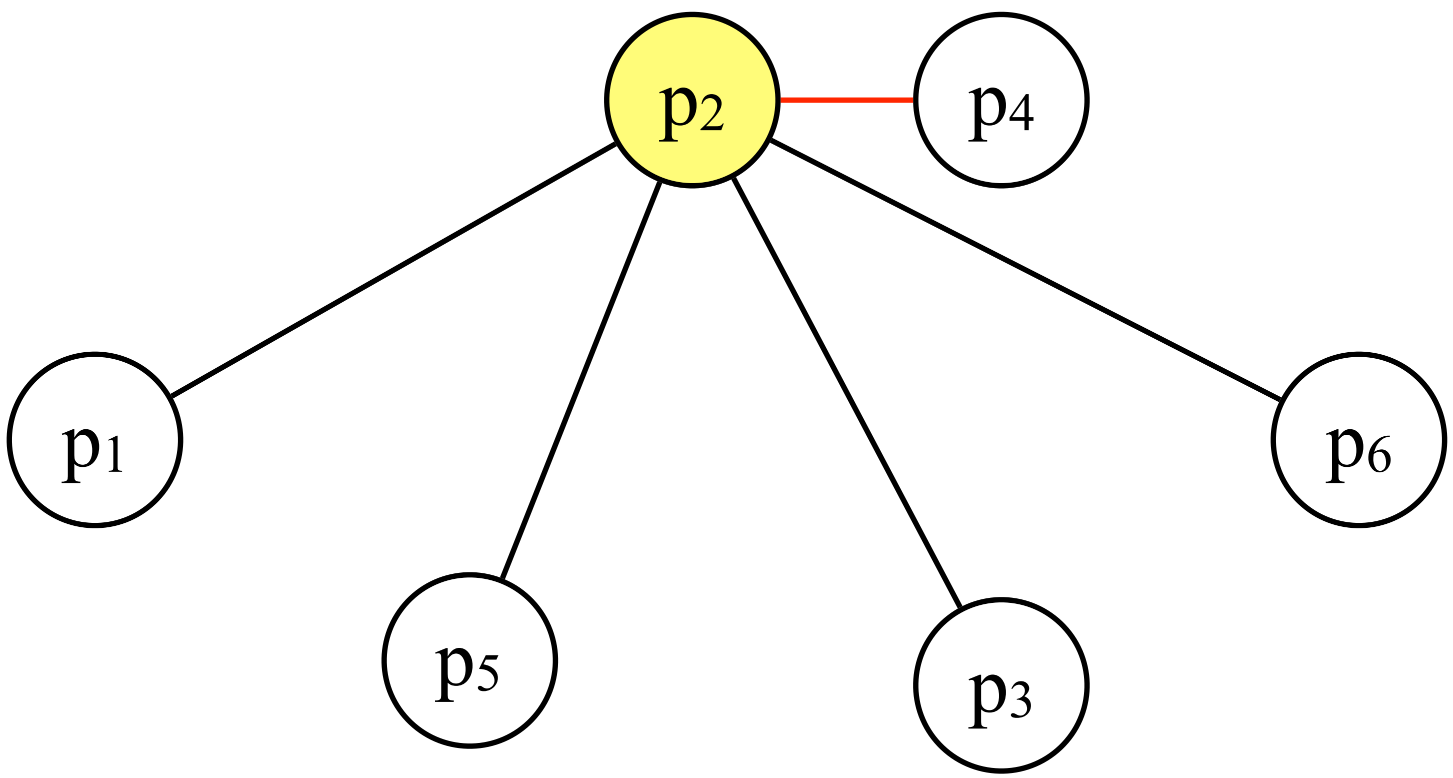
We can solve CP(S) by calculating all pairwise distance. This approach needs $O(n^2)$ time.



Guess one point in the closest pair is p_2.

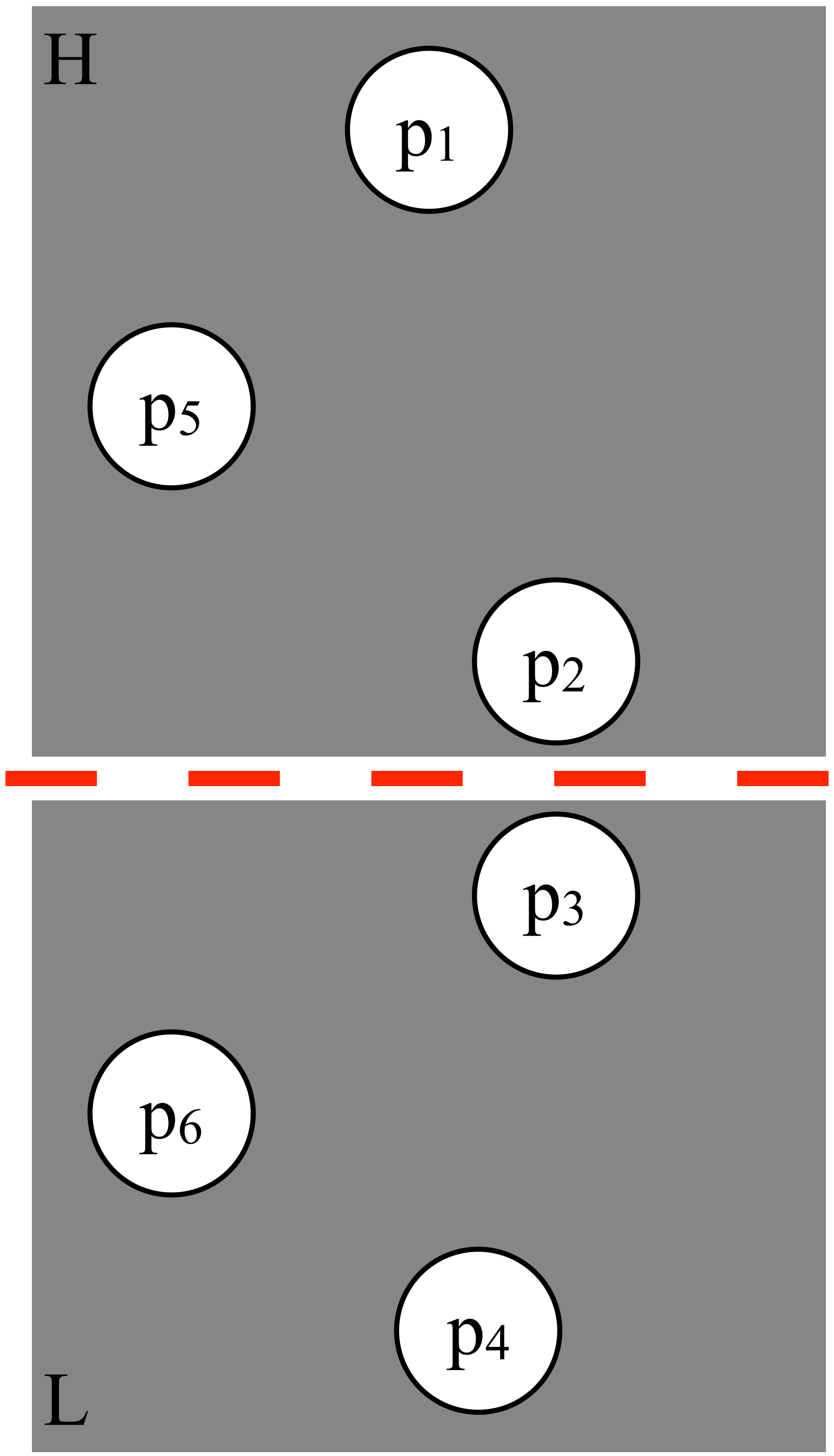
The Closest Pair of Points

We can solve CP(S) by calculating all pairwise distance. This approach needs $O(n^2)$ time.



Trying all possible guesses needs $O(n^2)$ time.

The Closest Pair of Points



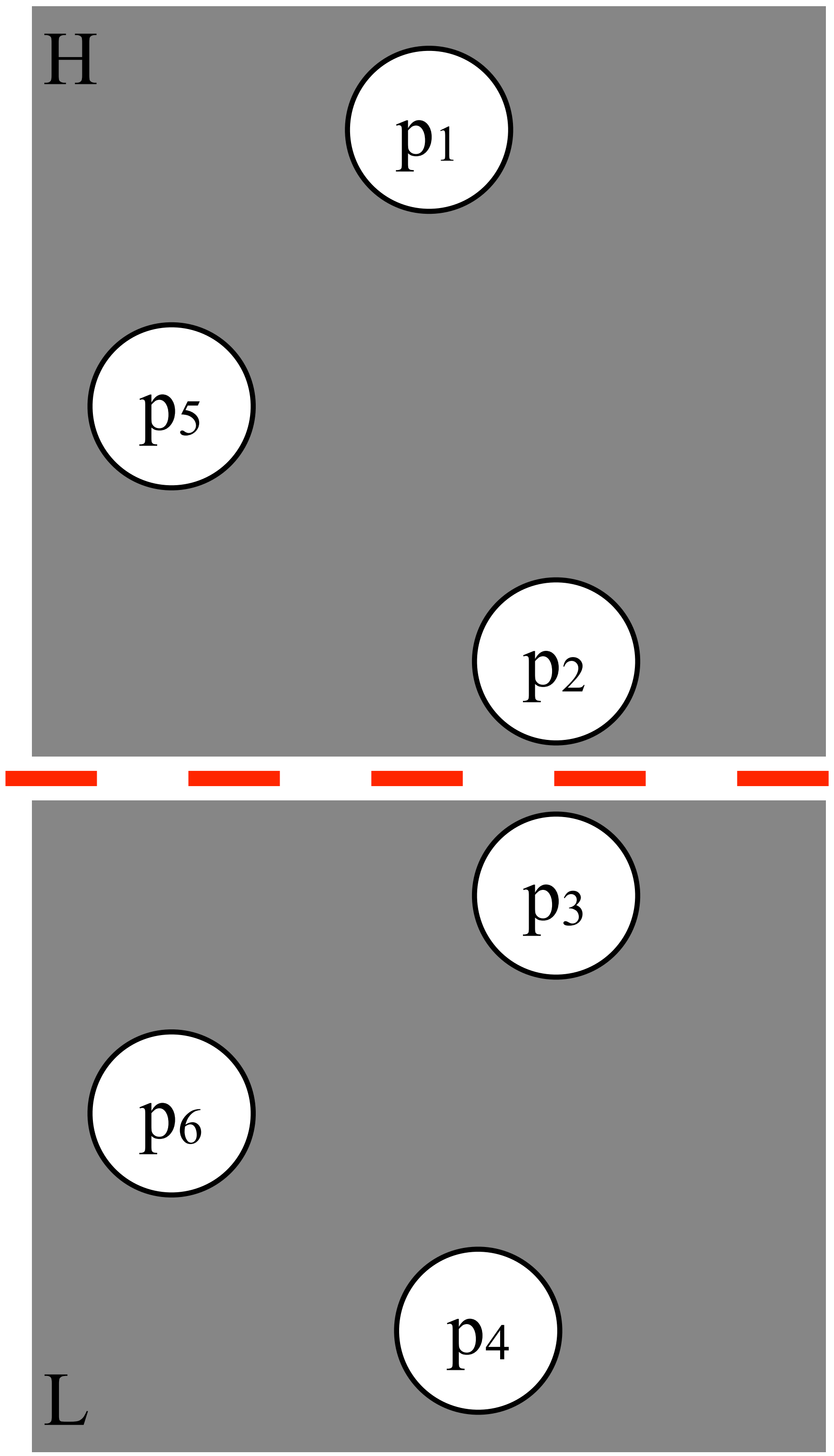
Try the divide-and-conquer approach.

Step 1.
If $|S| = O(1)$, we solve CP(S) by the naive approach in $O(1)$ time.

Otherwise, divide S into two subsets L and H where $||L|-|H|| \leq 1$.

Finding a horizontal line that partition S evenly needs $O(|S|)$ time by the **median selection**.

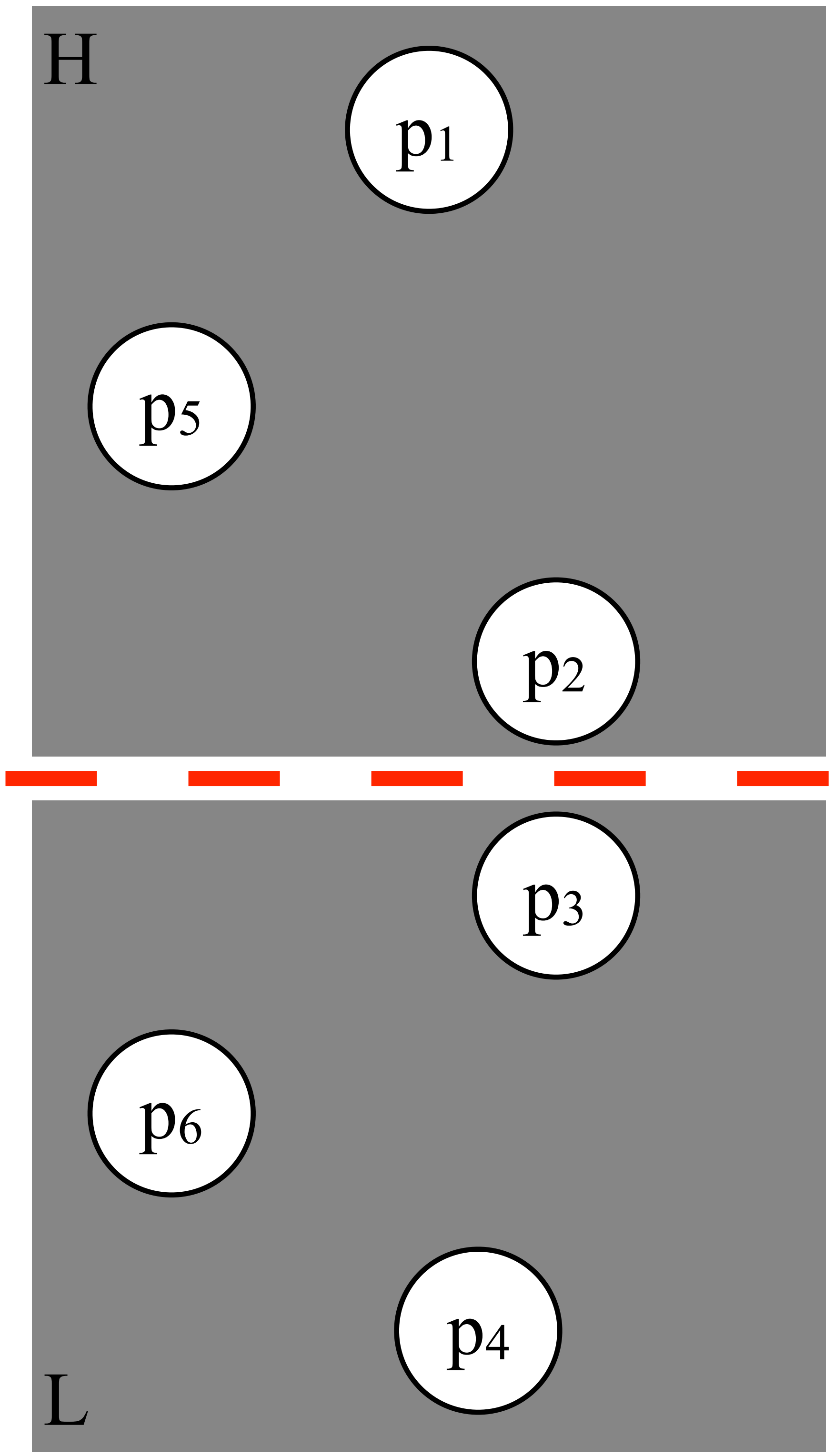
The Closest Pair of Points



Try the divide-and-conquer approach.

Step 2.
Recurse on the subproblems CP(L) and CP(H). Let $\boldsymbol{\delta}_1$ = CP(L) and $\boldsymbol{\delta}_2$ = CP(H).

The Closest Pair of Points

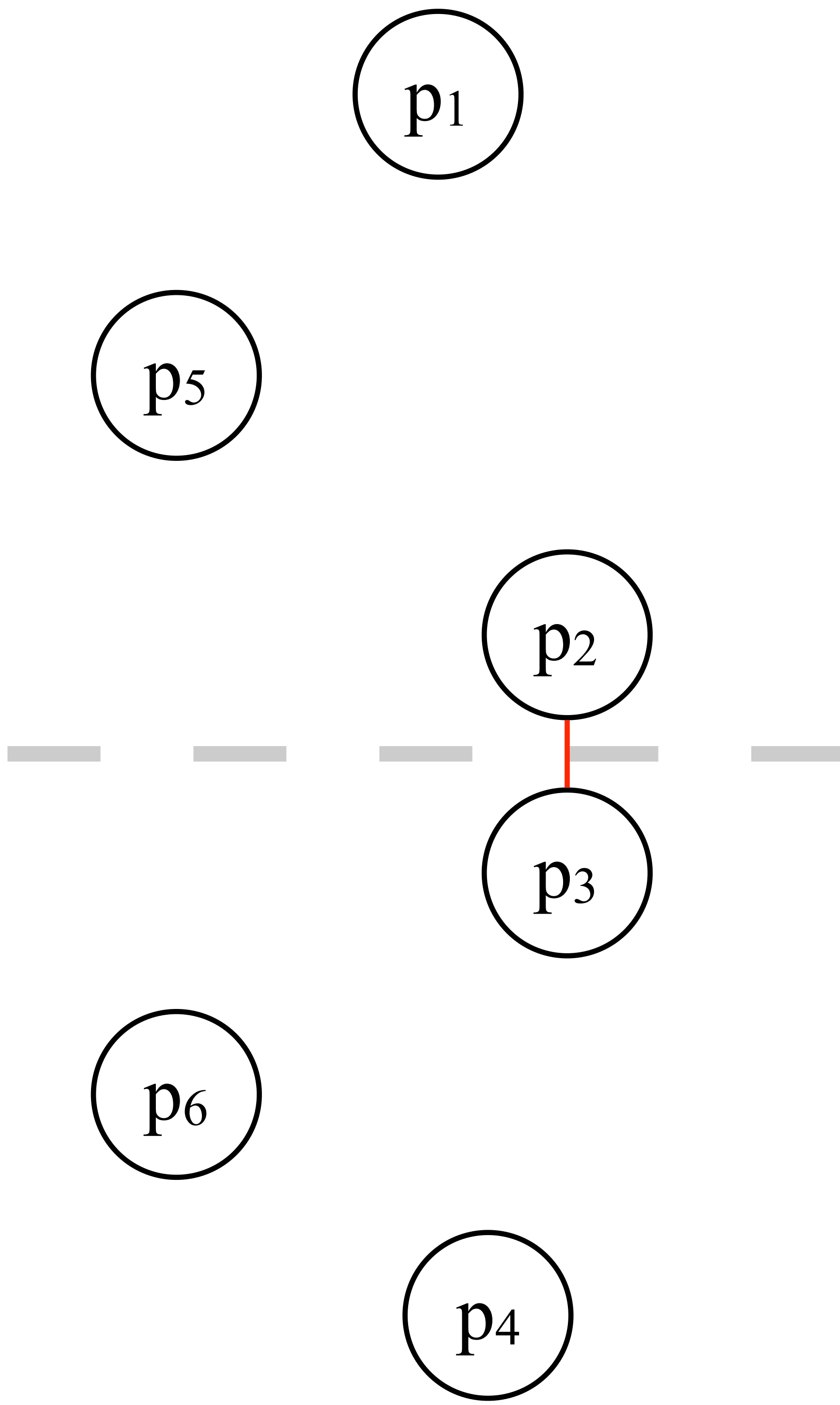


Try the divide-and-conquer approach.

Step 3.
Combine the results of the subproblems. Let $\boldsymbol{\delta} = \min(\boldsymbol{\delta}_1, \boldsymbol{\delta}_2)$.

Is $\boldsymbol{\delta}$ the answer?

The Closest Pair of Points

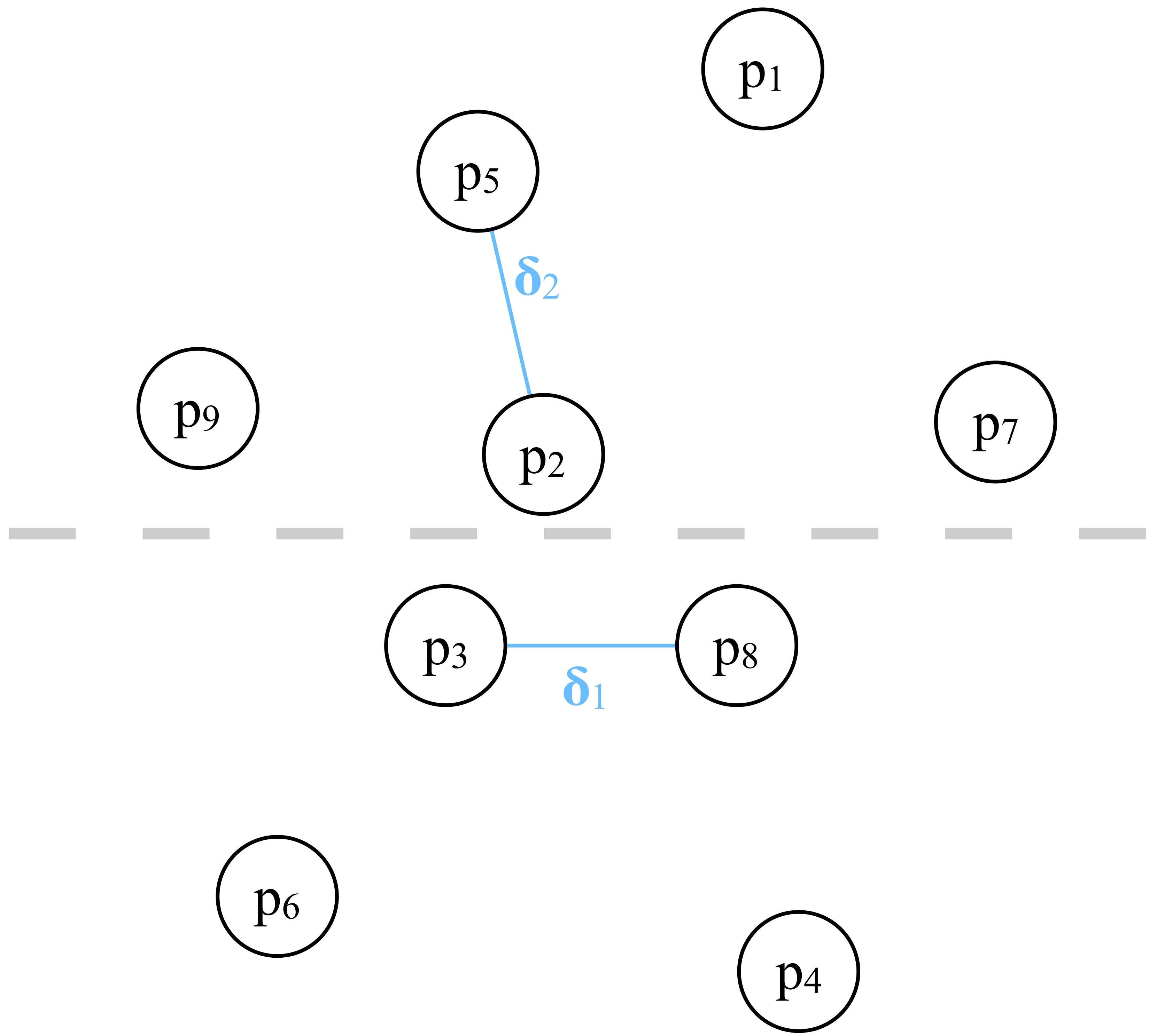


Try the divide-and-conquer approach.

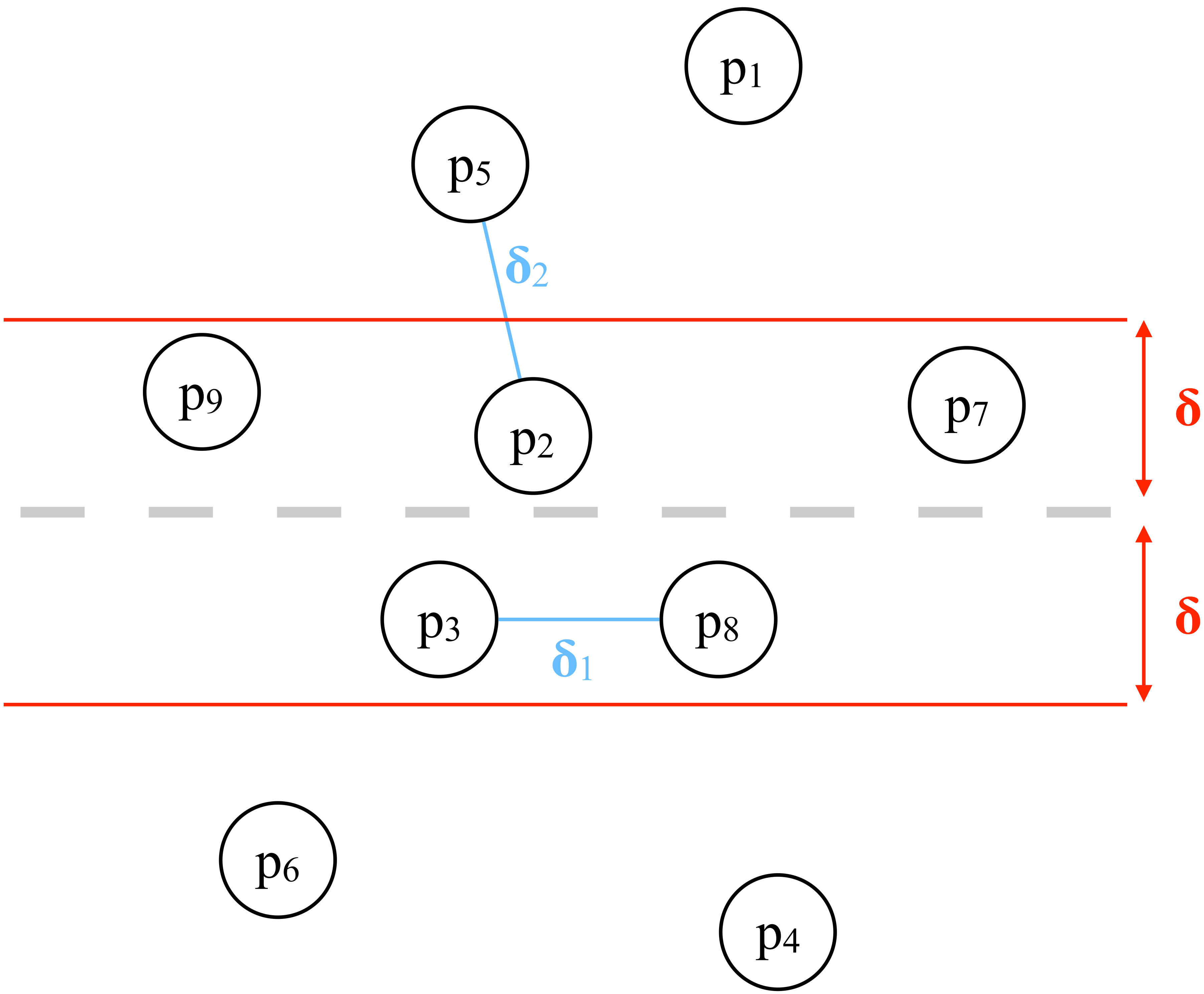
<u>Step 3</u>.
Combine the results of the subproblems. Let $\boldsymbol{\delta} = \min(\boldsymbol{\delta}_1, \boldsymbol{\delta}_2)$.

$\boldsymbol{\delta}$ is a wrong answer if the closest pair is separated by the partition line.

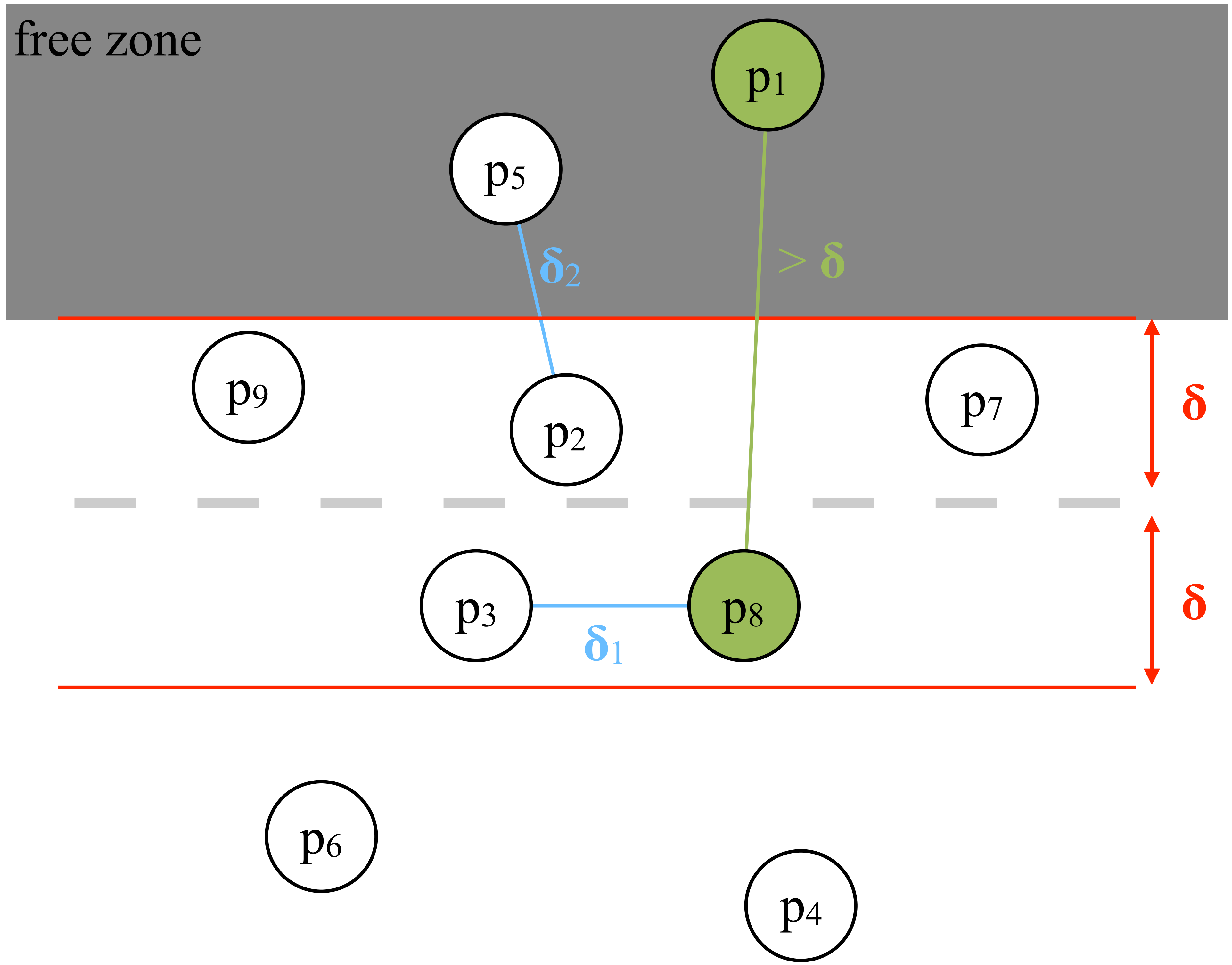
The Closest Pair of Points



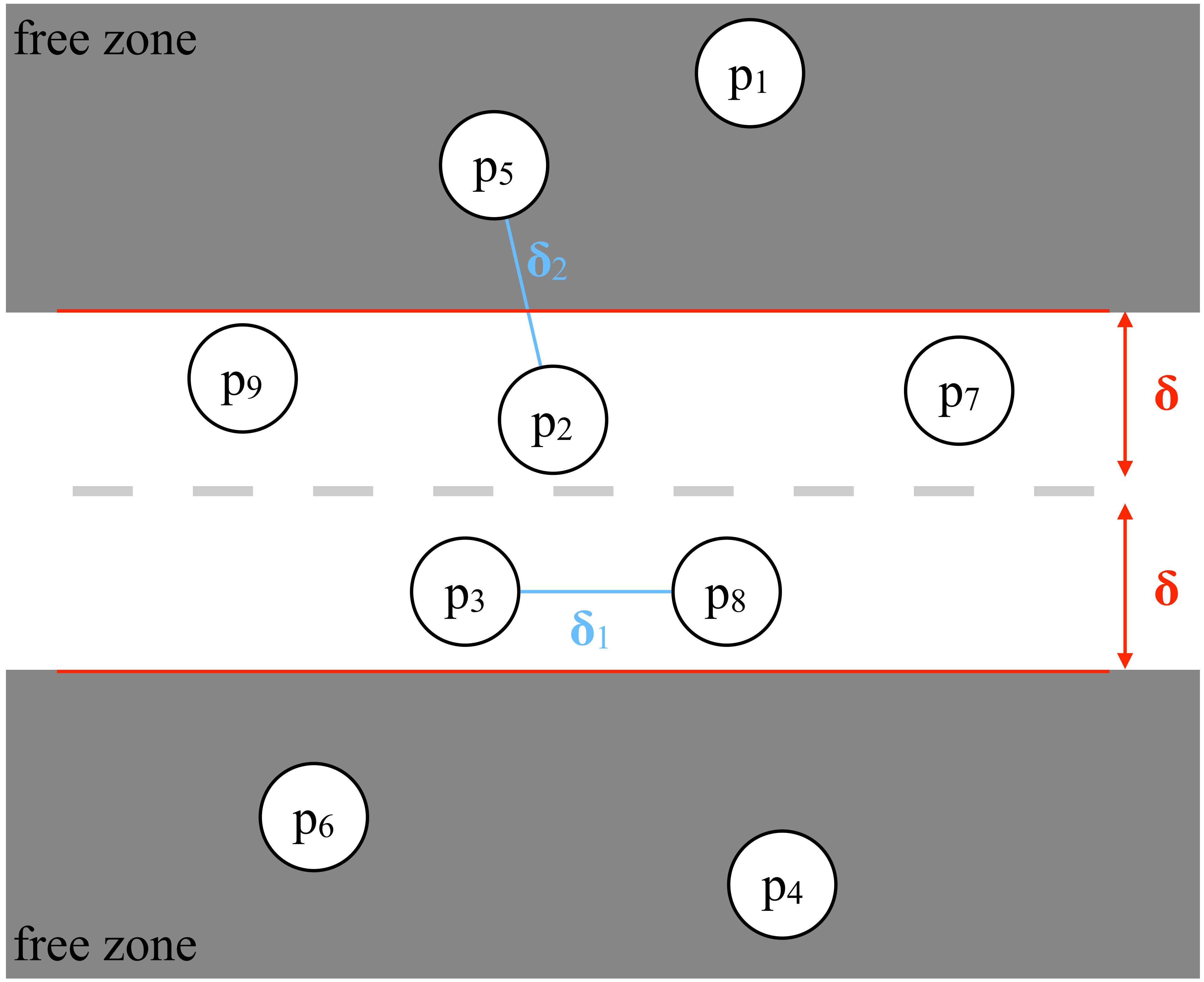
The Closest Pair of Points



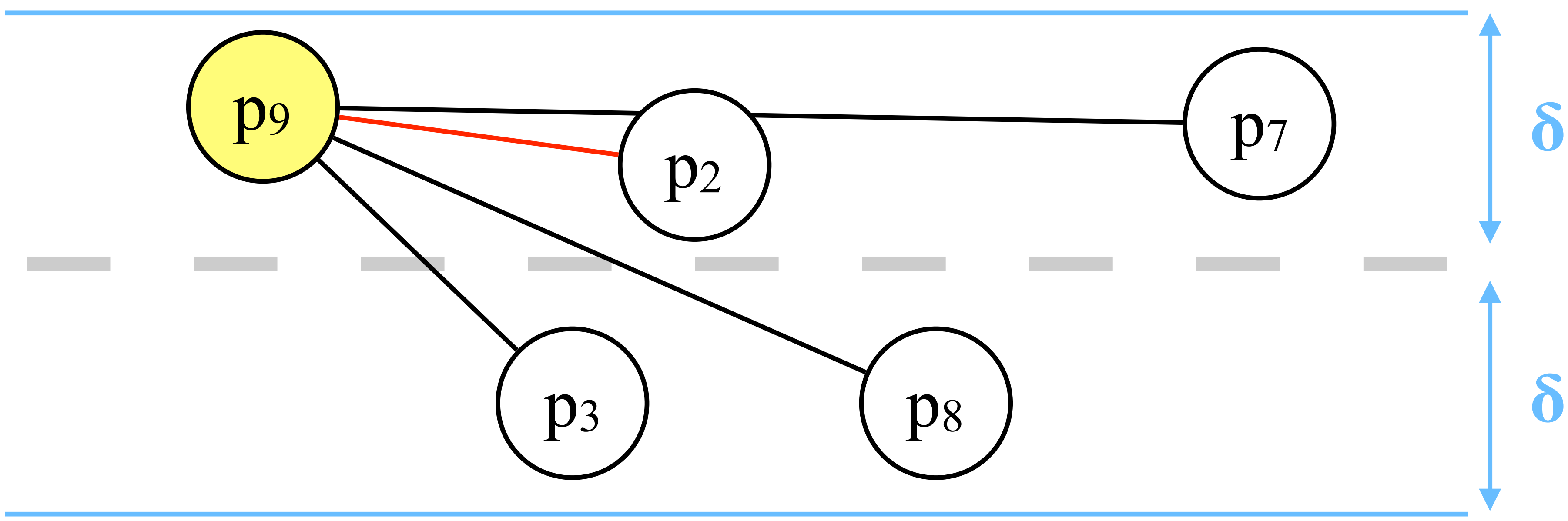
The Closest Pair of Points



The Closest Pair of Points

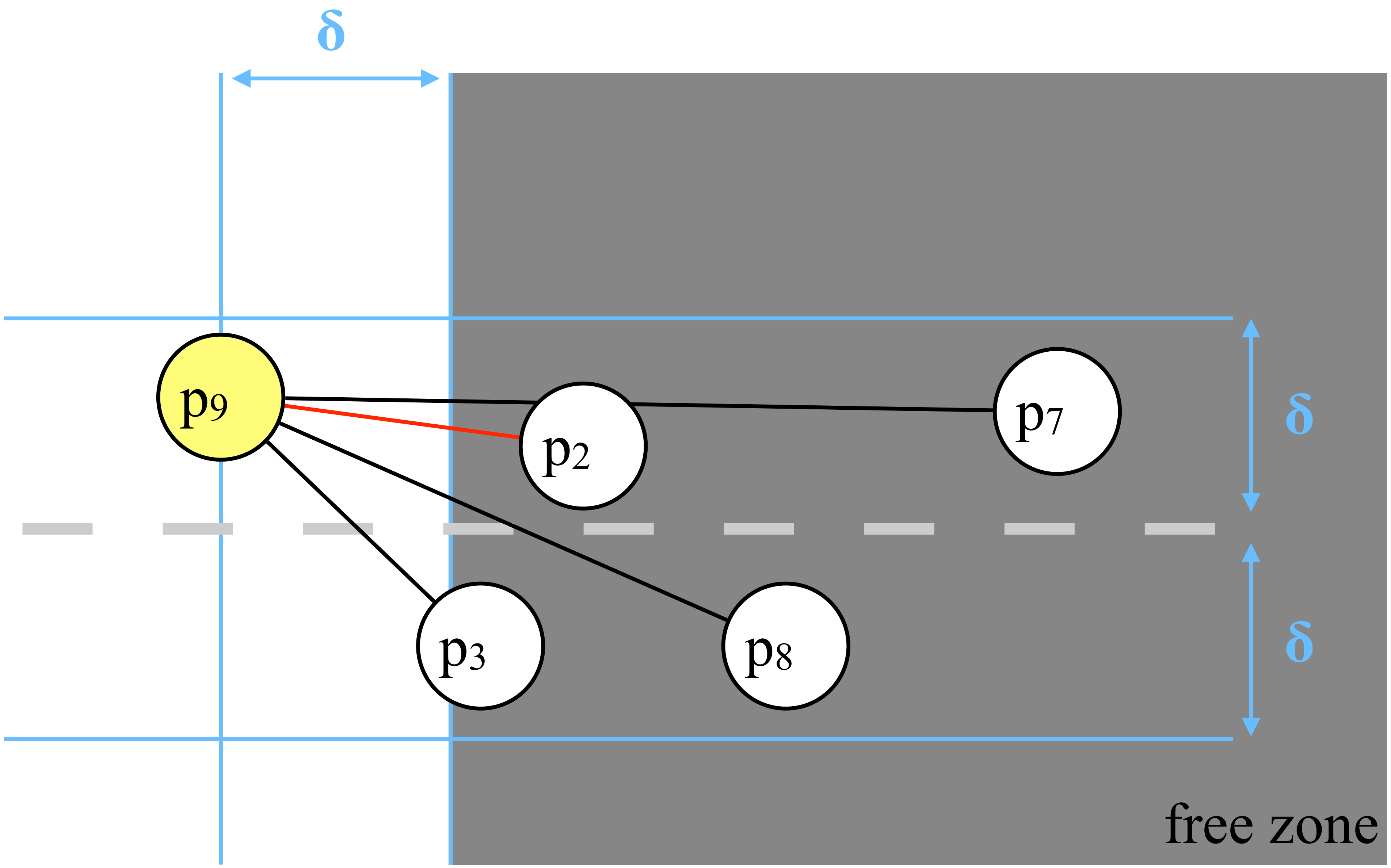


The Closest Pair of Points



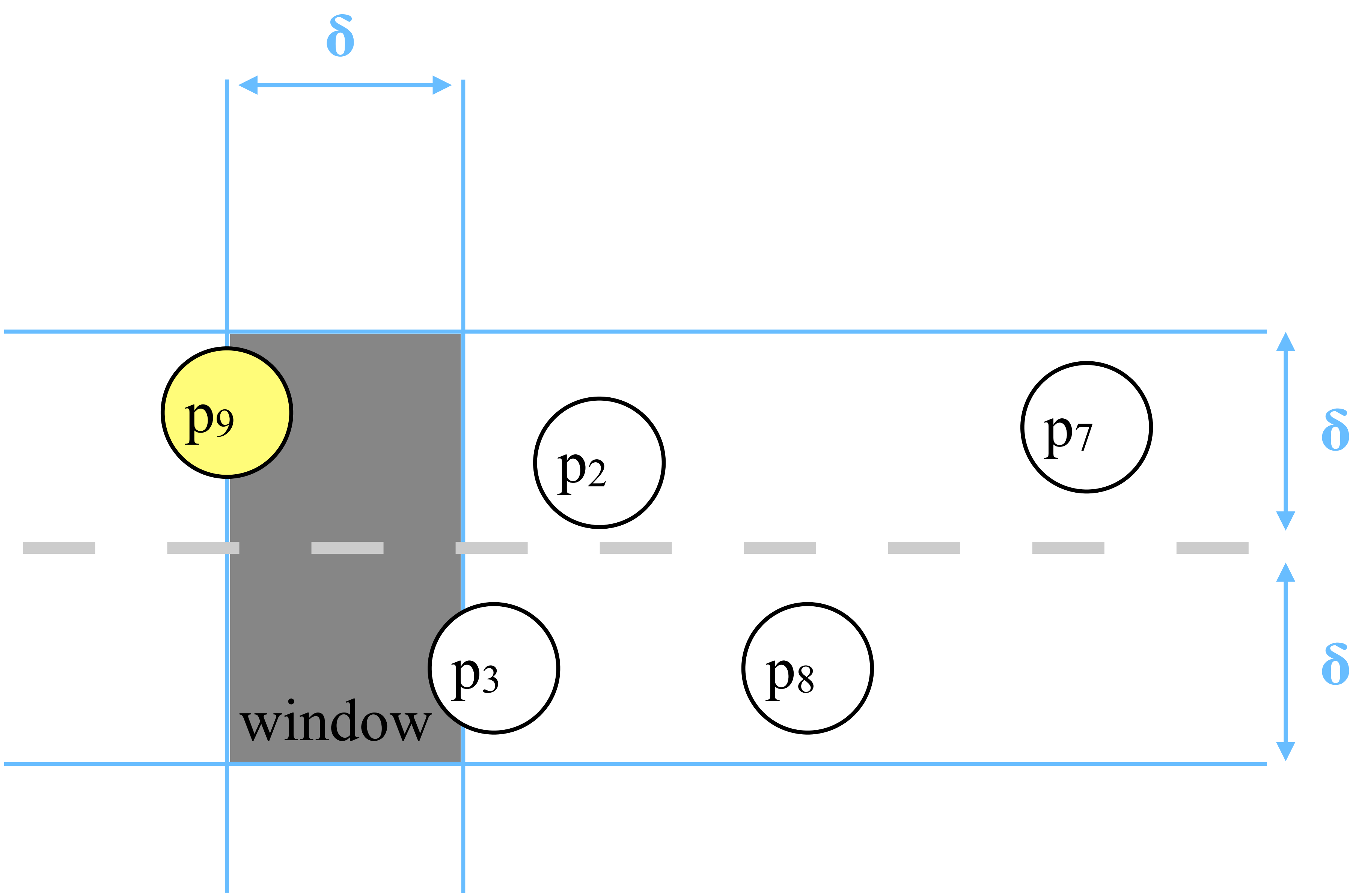
Guess one point in the pair is p_9.

The Closest Pair of Points



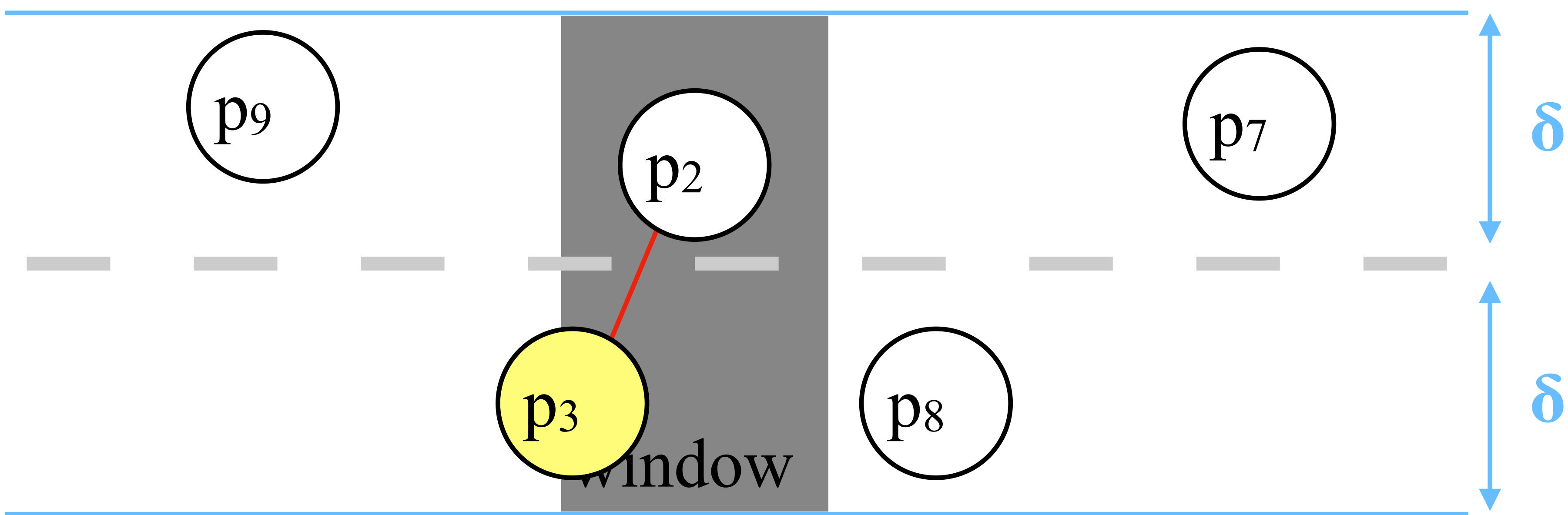
A lot of trials are unnecessary.

The Closest Pair of Points



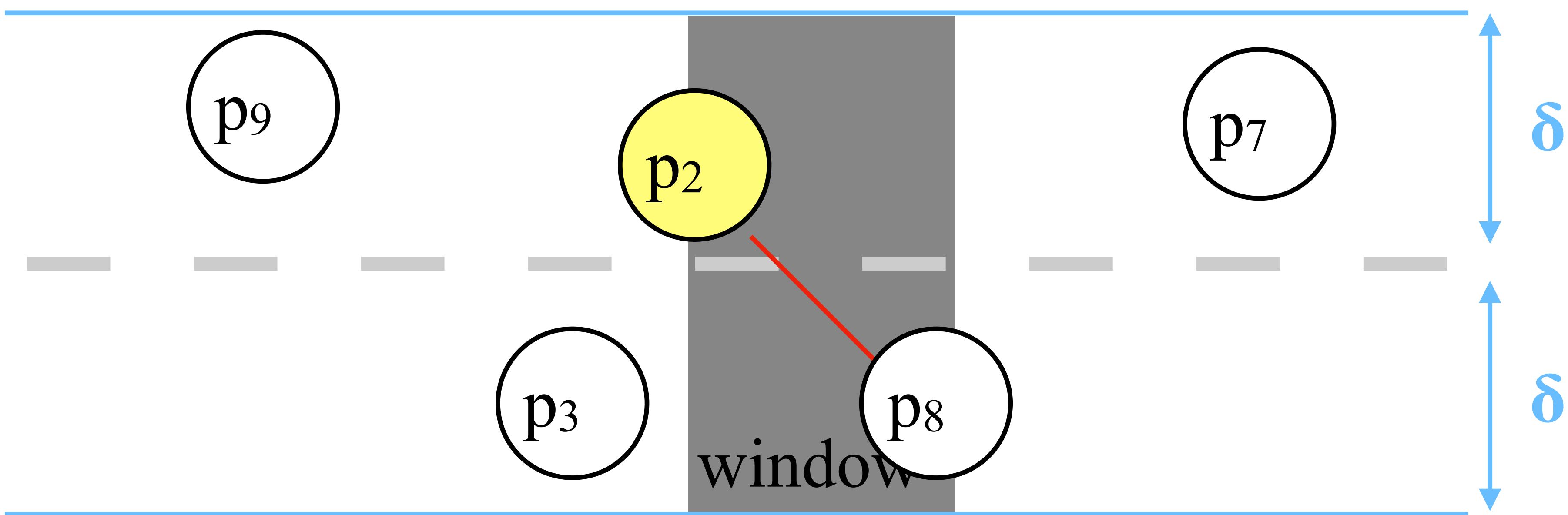
We only need to check the points in the window.

The Closest Pair of Points



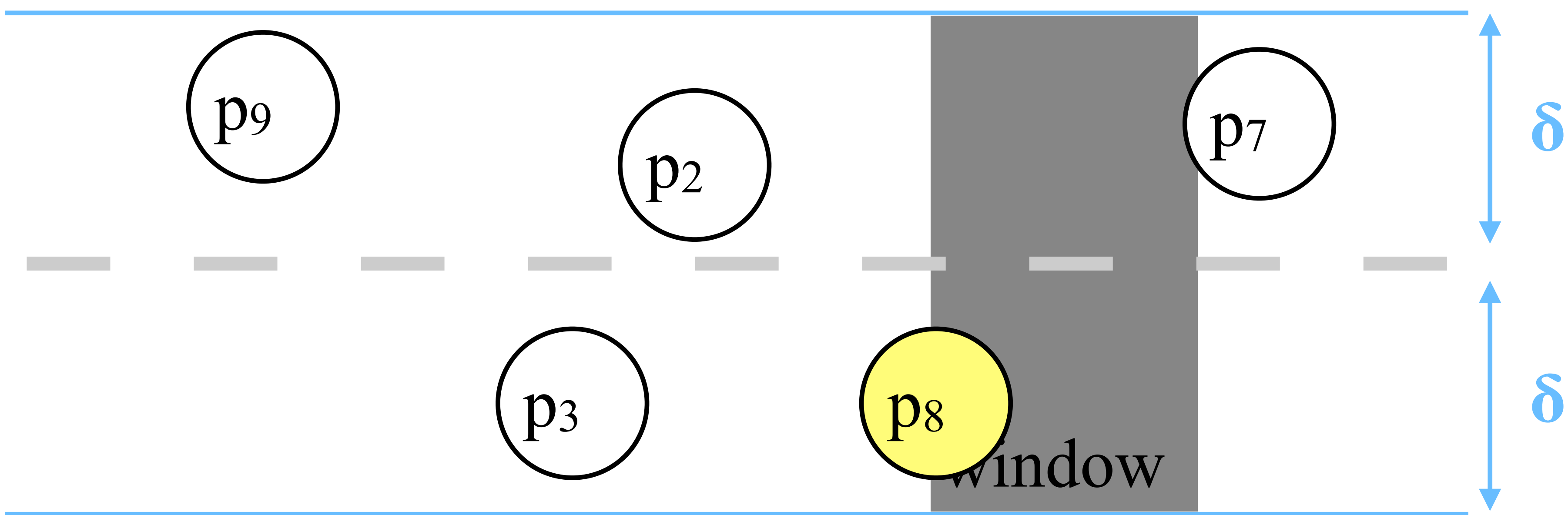
We only need to check the points in the window.

The Closest Pair of Points



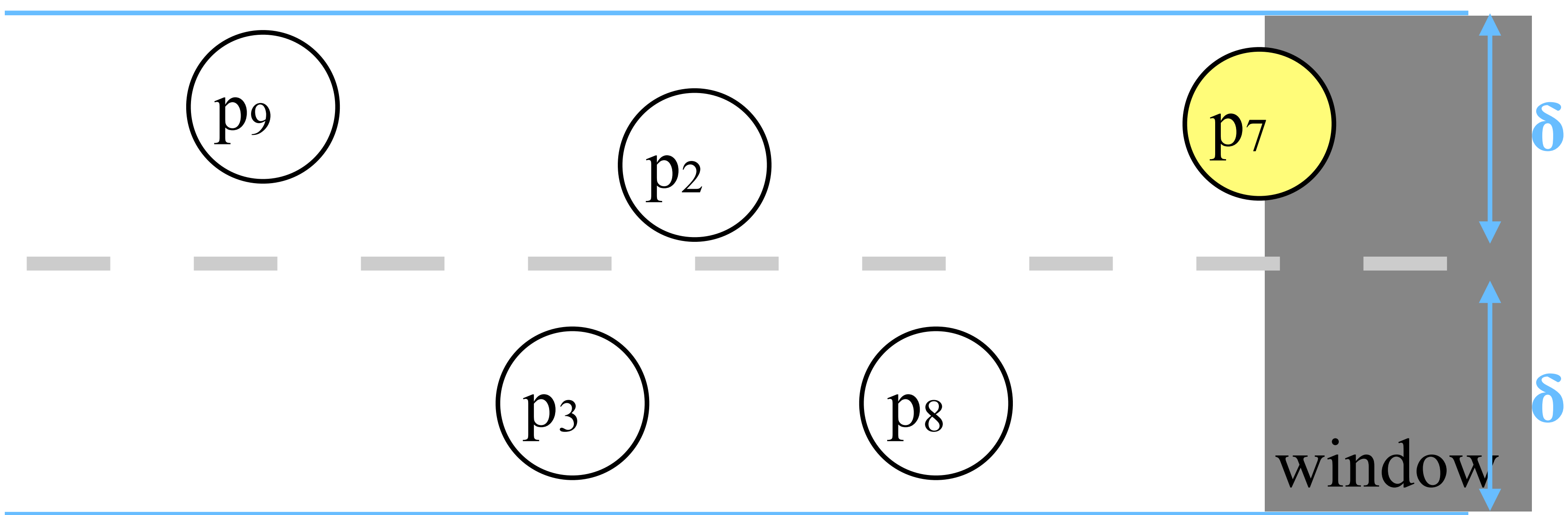
We only need to check the points in the window.

The Closest Pair of Points



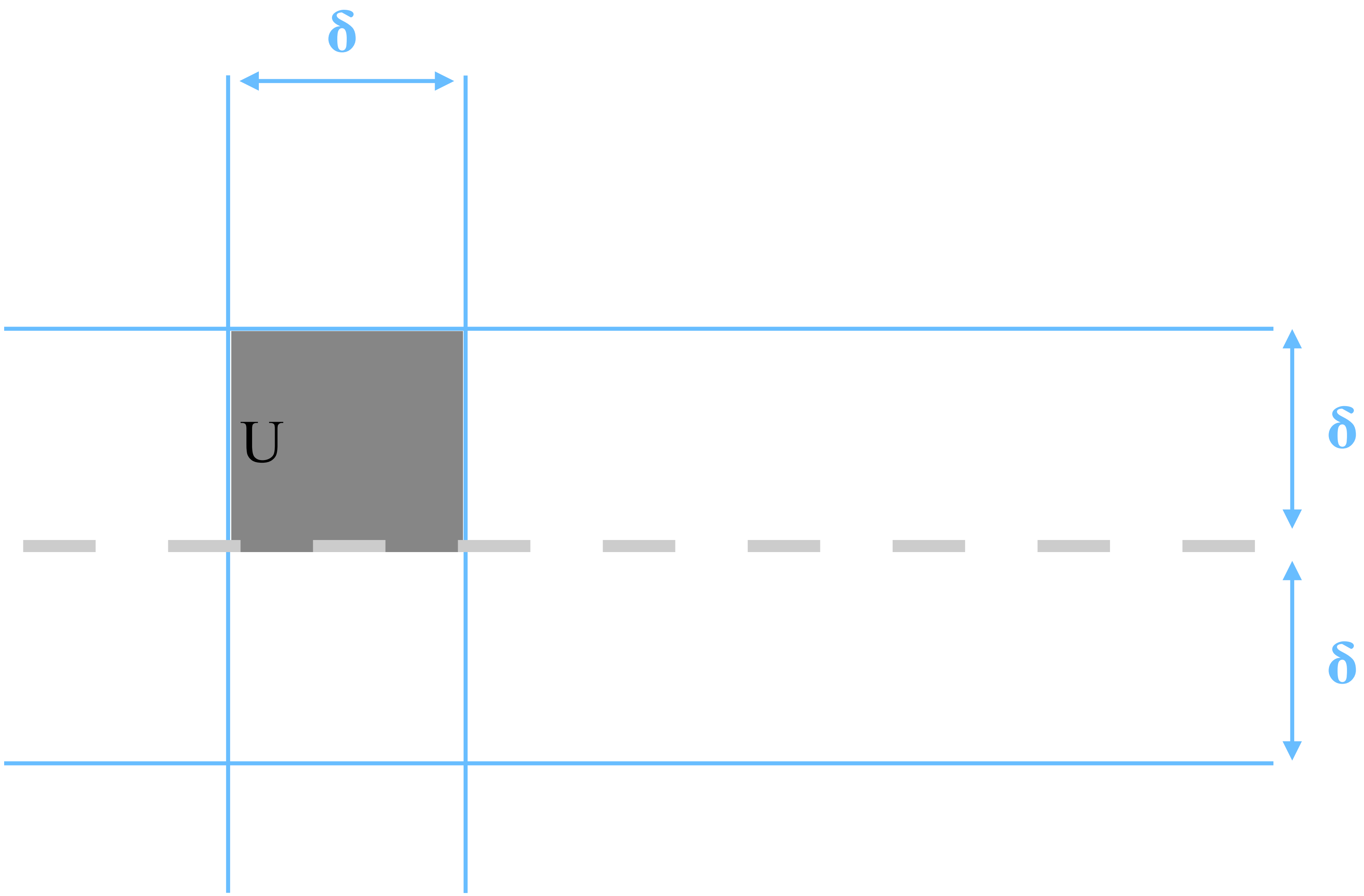
We only need to check the points in the window.

The Closest Pair of Points



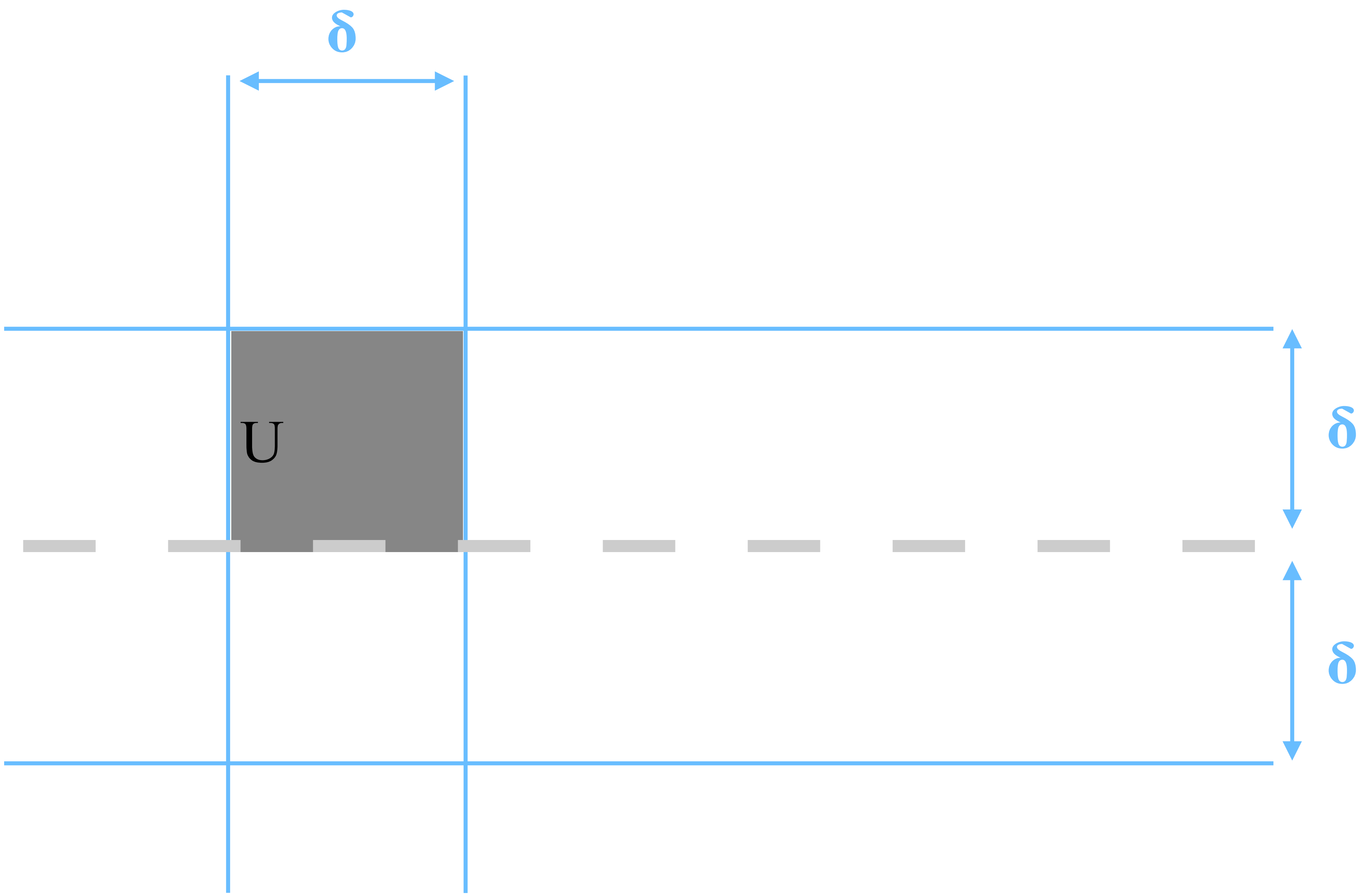
We only need to check the points in the window.

The Closest Pair of Points



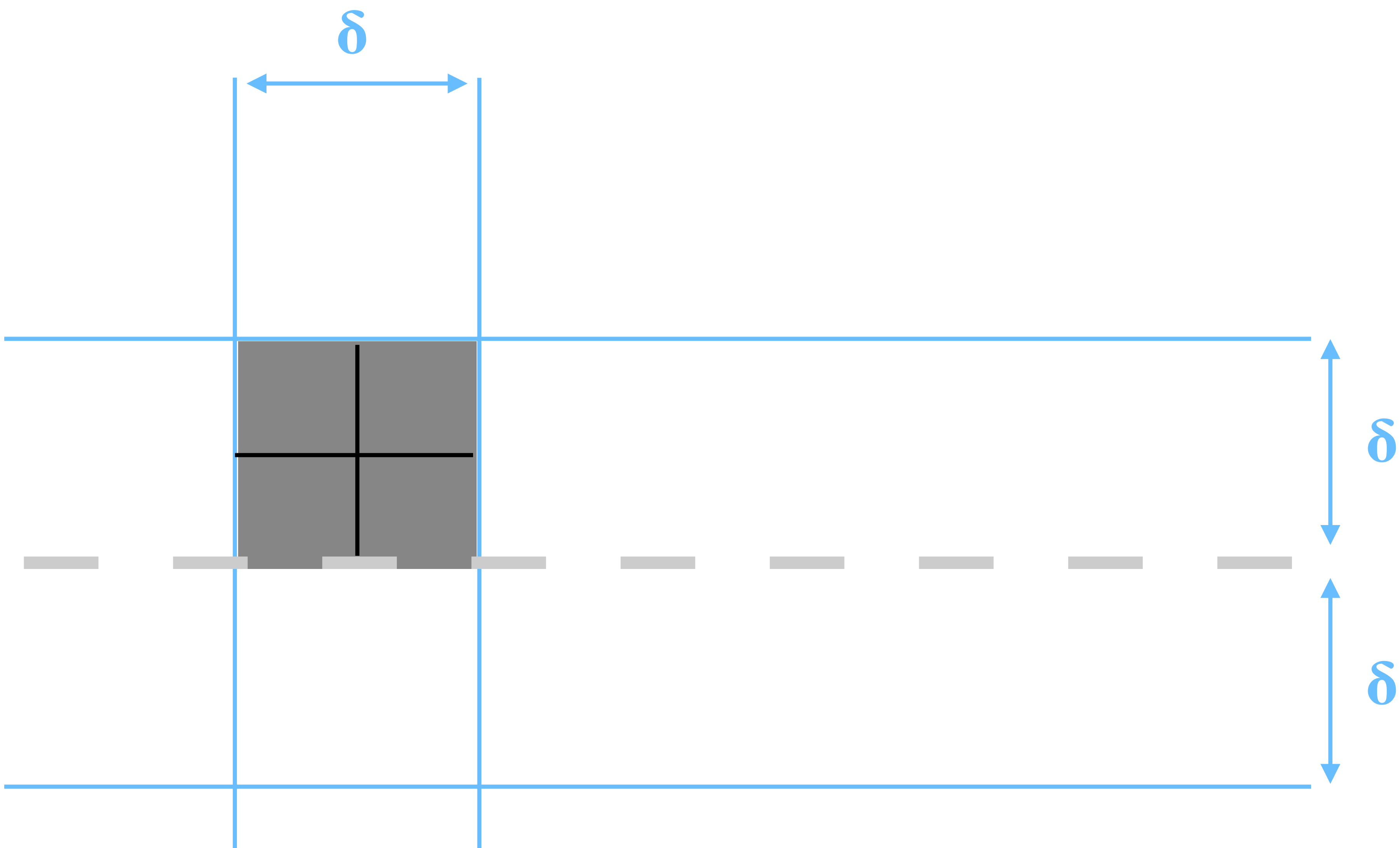
How many points can be in U so that every pair of points in U has distance $\geq \boldsymbol{\delta}_2 \geq \boldsymbol{\delta}$?

The Closest Pair of Points



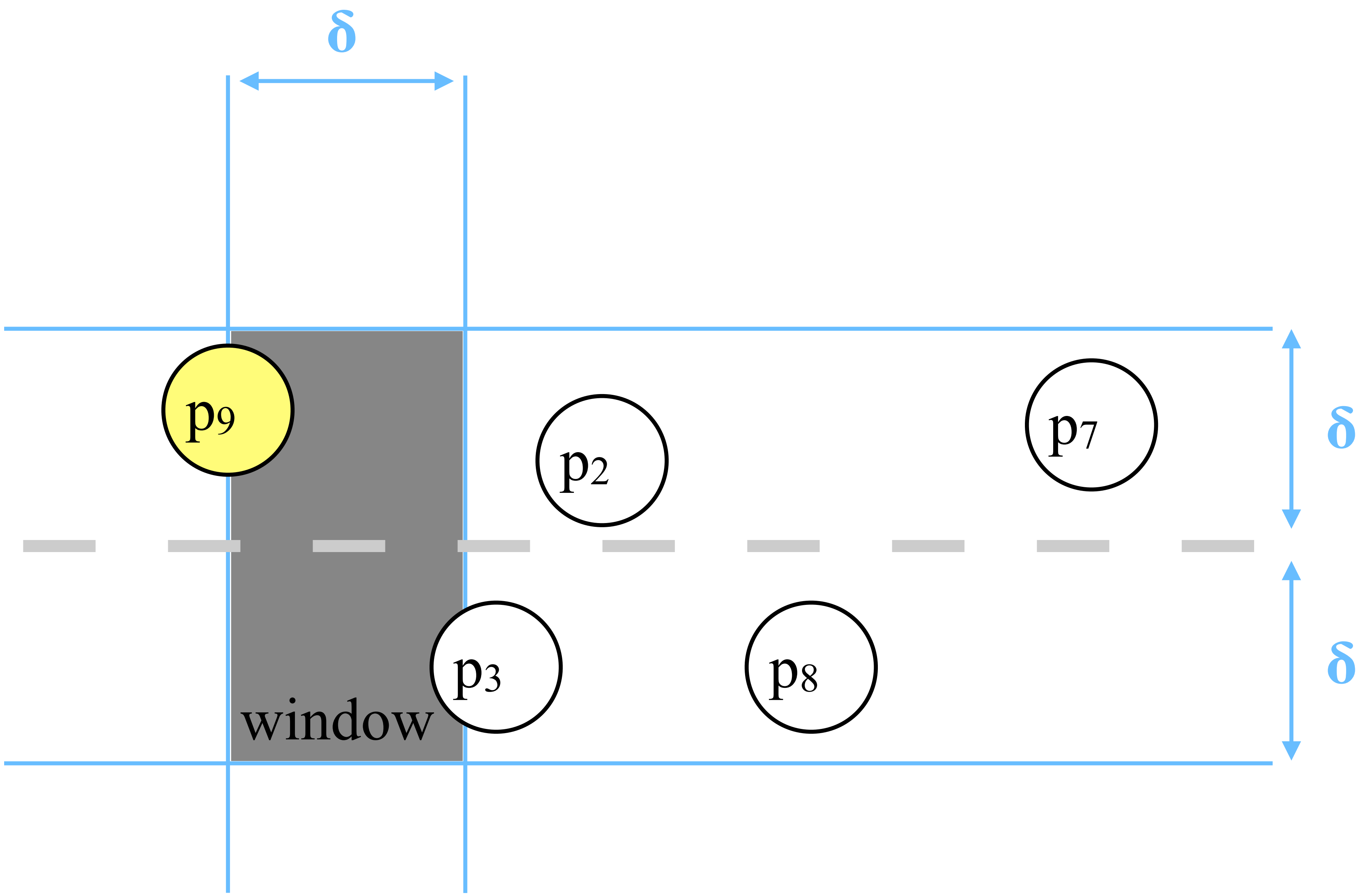
Of course it is 4, but why?

The Closest Pair of Points



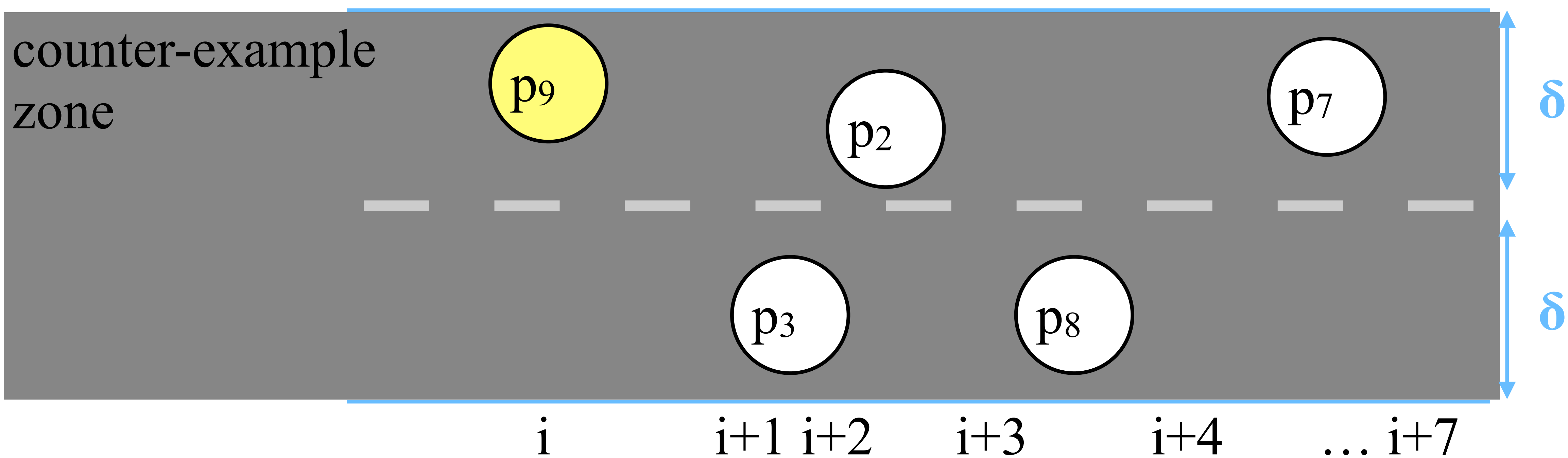
Suppose the answer ≥ 5. We can divide the region U into 4 smaller ones, one of which shall contain ≥ 2 points by the pigeon-hole principle. This gives a contradiction because these two points have distance at most $\frac{1}{\sqrt{2}}\delta < \delta$.

The Closest Pair of Points



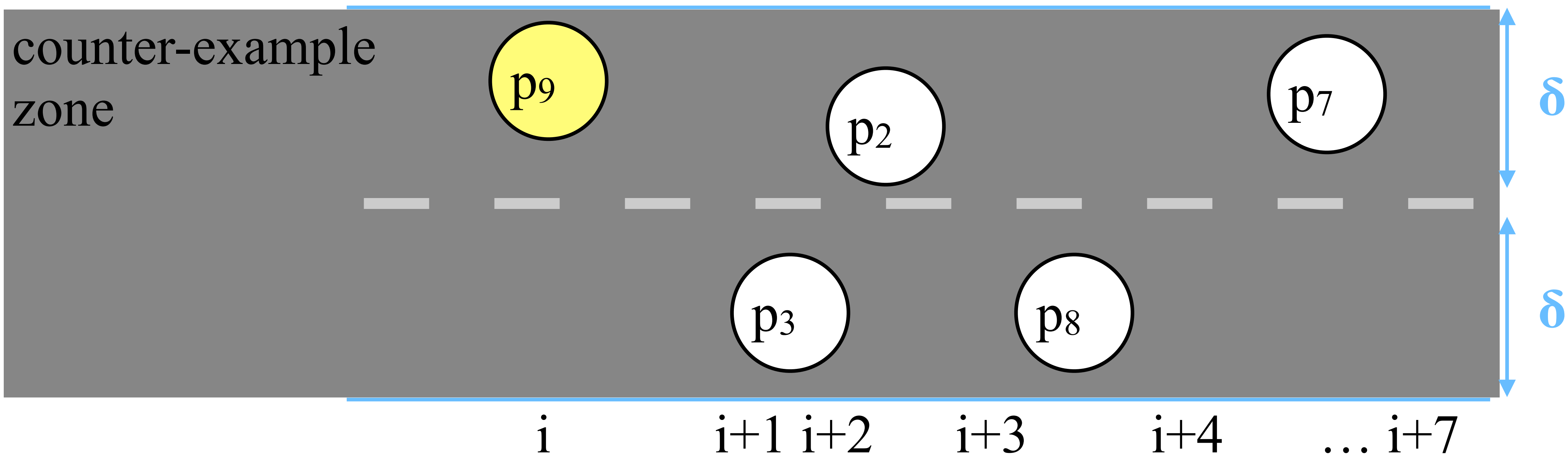
Hence, there are at most 8 points
in the window.

The Closest Pair of Points



If we sort the points in the counter-example zone by their x-coordinate, for every guess we only need to check the next 7 successor points.

The Closest Pair of Points



Hence, checking every guess in the counter-example zone needs at most 7 * |S| = O(|S|) comparisons, assuming that the points are sorted by x-coordinate.

The Closest Pair of Points

CP(S){

If $|S| \leq 3$, compute CP(S) naively and return S sorted by the x-coordinates.

If $|S| > 3$, find the horizontal line by the median selection, based on which we divide S into L and H.

Recurse on CP(L) and CP(H). Let $\boldsymbol{\delta}_1$ = CP(L) and $\boldsymbol{\delta}_2$ = CP(L).
Obtain the sorted S by merging the returned (sorted) L and H.

Get $\boldsymbol{\delta}_{fix}$ by checking the counter-example zone.

Return $\boldsymbol{\delta}_{fix}$ and the sorted S.

}

The Closest Pair of Points

$$T(n) = \begin{cases} T(\lfloor n/2 \rfloor) + T(\lceil n/2 \rceil) + O(n) & \text{if } n > 3 \\ O(1) & \text{if } n \leq 3 \end{cases}$$

By Master Theorem, we have T(n) = O(n log n).