CSL 356: Analysis and Design of Algorithms

Ragesh Jaiswal

CSE, IIT Delhi

Techniques

- Greedy Algorithms
- Divide and Conquer
- Dynamic Programming
- Network Flows

Divide and Conquer

You have already looked at many divide and conquer algorithms:

- Binary Search
- Merge Sort
- Quick Sort
- Multiplying two n bit numbers in $O\left(n^{\log_2(3)}\right)$time.

Divide and Conquer: Introduction

- Main Idea:
  - Divide: *Divide the input into smaller parts.*
  - Conquer: *Solve the smaller problems and combine their solution.*

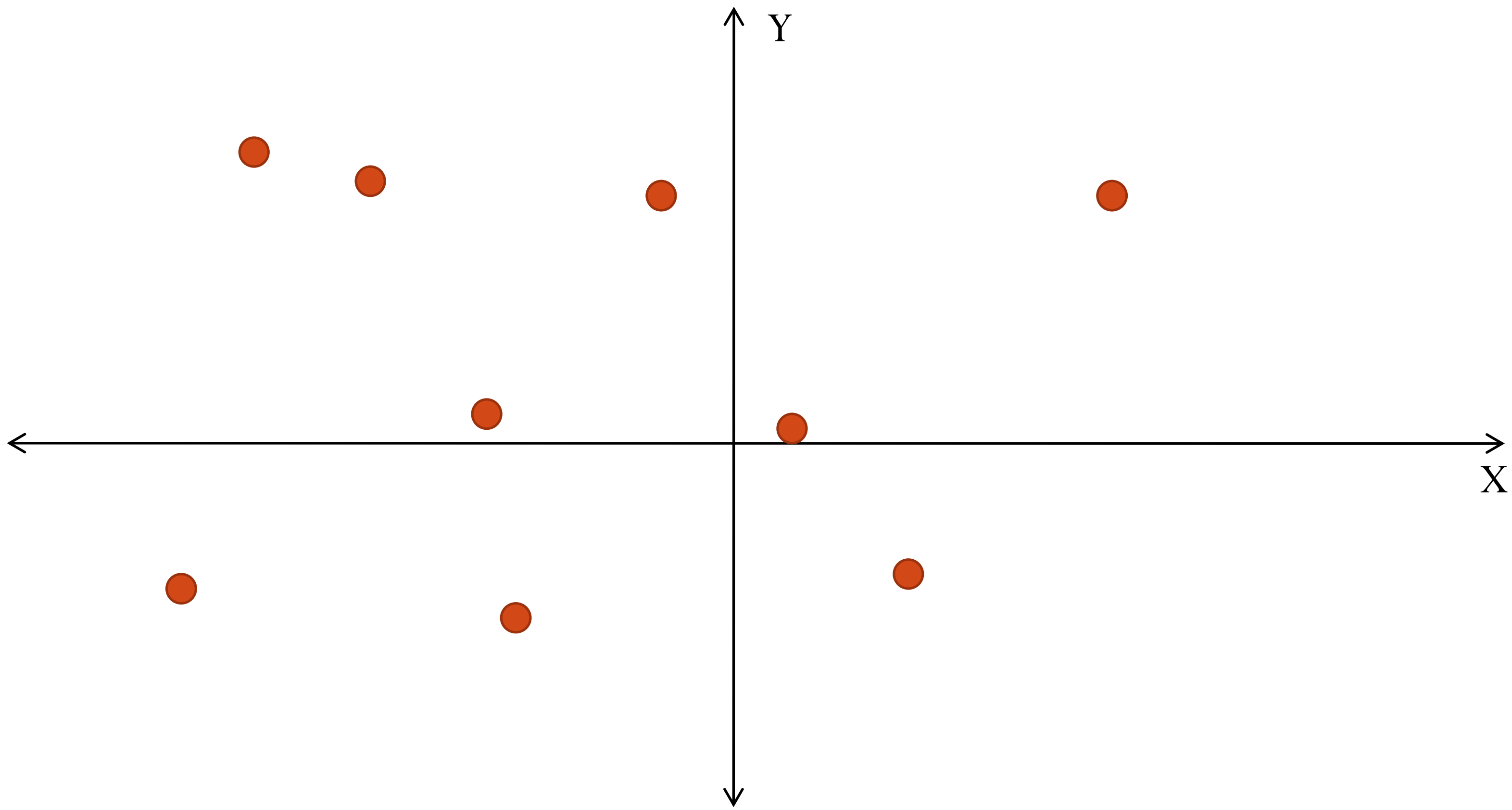
- Example: Merge Sort
  - Divide the input array A into two equal parts A_1 and A_2.
  - Recursively sort the array A_1 and A_2.
  - Merge the arrays A_1 and A_2.
  - Running time:
    - $T(n) = 2 \cdot T(n/2) + O(n)$
    - $T(1) = O(1)$
    - Solving the above recurrence relation, we get $T(n) = O(n \log n)$.

Divide and Conquer: Examples

Closest pair of points on a plane

Divide and Conquer: Examples

- Problem: You are given n points on a two dimensional plane. Each point i is defined by a pair $(x(i), y(i))$ of coordinates. Give an algorithm that outputs the closest pair of points.



Divide and Conquer: Examples

- Problem: You are given n points on a two dimensional plane. Each point i is defined by a pair $(x(i), y(i))$ of coordinates. Give an algorithm that outputs the closest pair of points.

- Brute-force algorithm: Consider all pairs and pick closest.
  - Running Time:

Divide and Conquer: Examples

- Problem: You are given n points on a two dimensional plane. Each point i is defined by a pair $(x(i), y(i))$ of coordinates. Give an algorithm that outputs the closest pair of points.

- Brute-force algorithm: Consider all pairs and pick closest.
  - Running Time: $O(n^2)$

Divide and Conquer: Examples

- Problem: You are given n points on a two dimensional plane. Each point i is defined by a pair $(x(i), y(i))$ of coordinates. Give an algorithm that outputs the closest pair of points.

- Divide and Conquer:
  - Based on X-axis. Consider left-half points P_L and right-half points P_R.

Divide and Conquer: Examples

- Problem: You are given n points on a two dimensional plane. Each point i is defined by a pair $(x(i), y(i))$ of coordinates. Give an algorithm that outputs the closest pair of points.

- Divide and Conquer:
  - Based on X-axis. Consider left-half points P_L and right-half points P_R.
  - Recursively find the closest pair of points in $P_L, (i_L, j_L)$ and $P_R, (i_R, j_R)$.

Divide and Conquer: Examples

- Problem: You are given n points on a two dimensional plane. Each point i is defined by a pair $(x(i), y(i))$ of coordinates. Give an algorithm that outputs the closest pair of points.

- Divide and Conquer:
  - Based on X-axis. Consider left-half points P_L and right-half points P_R.
  - Recursively find the closest pair of points in $P_L, (i_L, j_L)$ and $P_R, (i_R, j_R)$.
  - Consider all pair of points (p, q) such that p belongs to P_L and q belongs to P_R.

Divide and Conquer: Examples

- Problem: You are given n points on a two dimensional plane. Each point i is defined by a pair $(x(i), y(i))$ of coordinates. Give an algorithm that outputs the closest pair of points.

- Divide and Conquer:
  - Based on X-axis. Consider left-half points P_L and right-half points P_R.
  - Recursively find the closest pair of points in $P_L, (i_L, j_L)$ and $P_R, (i_R, j_R)$.
  - Consider all pair of points (p, q) such that p belongs to P_L and q belongs to P_R.
  - Find the pair (p, q) that is closest. Pick the closest pair among (i_L, j_L) , (i_R, j_R) and (p, q).

Divide and Conquer: Examples

- Problem: You are given n points on a two dimensional plane. Each point i is defined by a pair $(x(i), y(i))$ of coordinates. Give an algorithm that outputs the closest pair of points.

- Divide and Conquer:
  - Based on X-axis. Consider left-half points P_L and right-half points P_R.
  - Recursively find the closest pair of points in P_L,(i_L, j_L) and P_R,(i_R, j_R).
  - Consider all pair of points (p, q) such that p belongs to P_L and q belongs to P_R.
  - Find the pair (p, q) that is closest. Pick the closest pair among (i_L, j_L) , (i_R, j_R) and (p, q).
- Running time?

Divide and Conquer: Examples

- Let $x = x^*$ be a line along the Y-axis dividing the points into P_L and P_R
- Let d be the distance between the closest pair of points in P_L and P_R.
- <u>Claim 1</u>: For any pair of points (p, q) such that

  $$x(p) < x^* - d \qquad \text{and} \qquad x(q) \geq x^*$$

  the distance between p and q is at least d.
- <u>Claim 2</u>: For any pair of points (p, q) such that

  $$x(p) \leq x^* \qquad \text{and} \qquad x(q) > x^* + d$$

  the distance between p and q is at least d.

Divide and Conquer: Examples

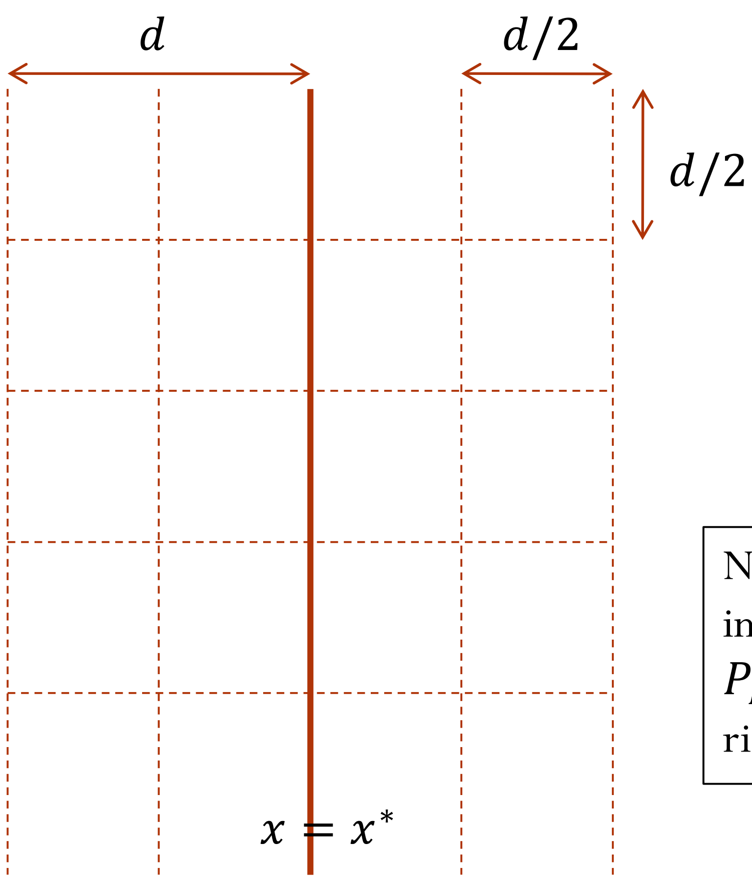
- Let $x = x^*$ be a line along the Y-axis dividing the points into P_L and P_R
- Let d be the distance between the closest pair of points in P_L and P_R.
- Claim 1: For any pair of points (p, q) such that
  $$x(p) < x^* - d \qquad \text{and} \qquad x(q) \geq x^*,$$
  the distance between p and q is at least d.
- Claim 2: For any pair of points (p, q) such that
  $$x(p) \leq x^* \qquad \text{and} \qquad x(q) > x^* + d,$$
  the distance between p and q is at least d.
- This means that for pairs of points across the line, we can throw any point in P_L that has small X-coordinate and any point in P_R that has large X-coordinate.
- Does this claim help in improving the running time?

Divide and Conquer: Examples

- How many points does each “box” contain?

Divide and Conquer: Examples

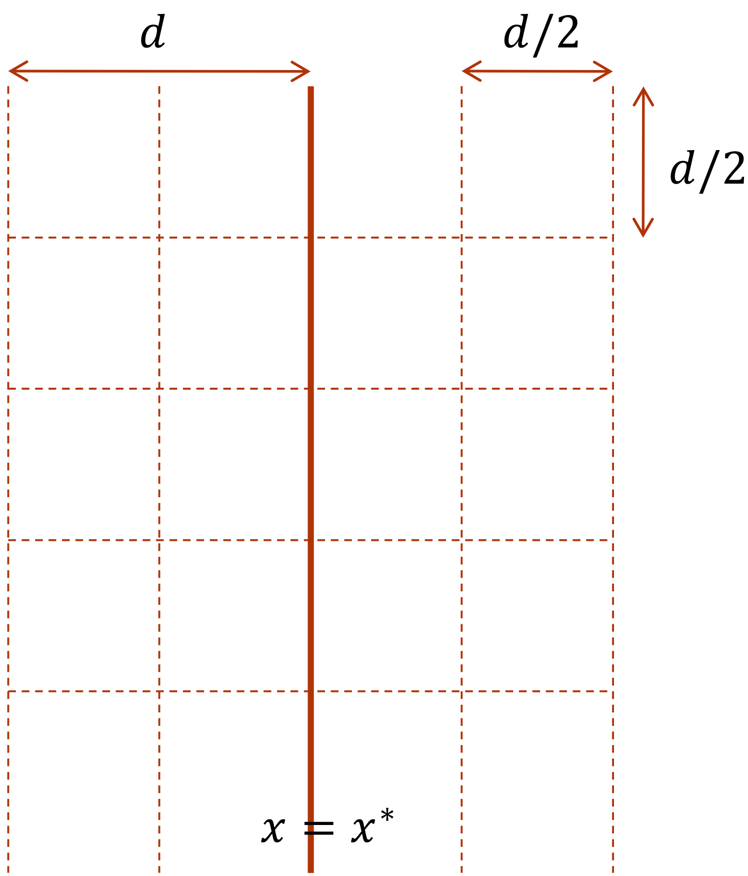
- Claim 3: Let P be all the points that have X-coordinate between $(x^* - d)$ and $(x^* + d)$. Let S be the sorted list of points in P sorted in increasing order of their Y-coordinates. Consider any pair of points (p, q) such that p belongs to P_L and q belongs to P_R and distance between p and q is at most d. There cannot be more than 10 points between p and q in the sorted list S.



Note that in Claim 3, P_L denotes the points in P that are to the left of the $x = x^*$ line and P_R denotes the points in P that are to the right of $x = x^*$.

Divide and Conquer: Examples

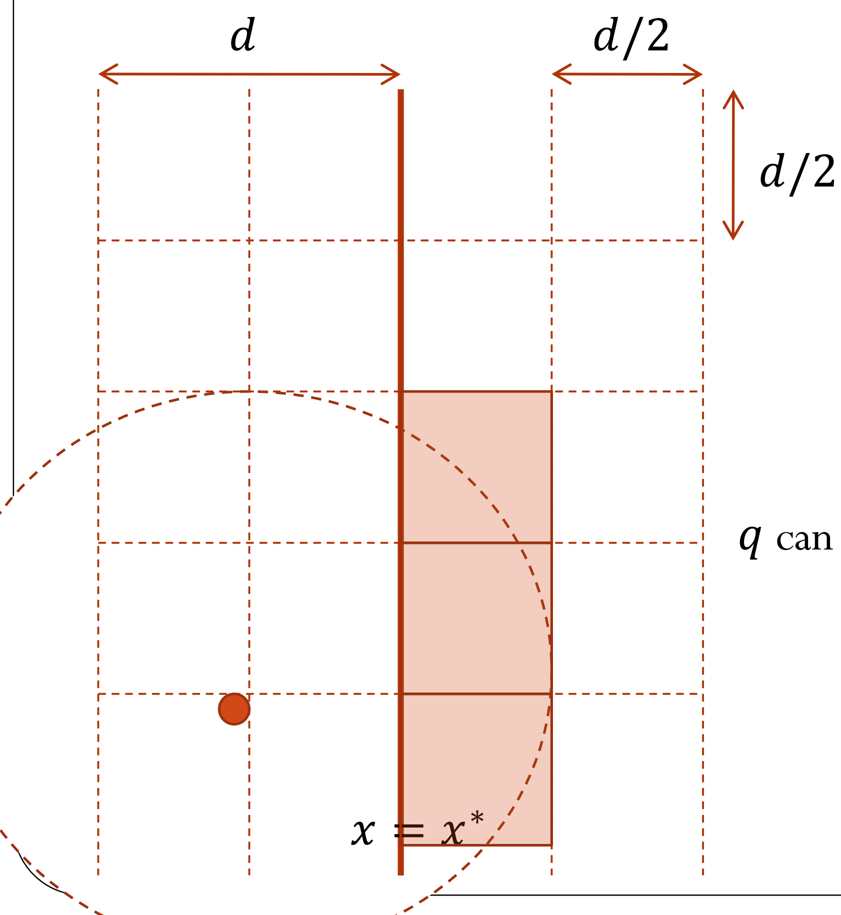
- Claim 3: Let P be all the points that have X-coordinate between $(x^* - d)$ and $(x^* + d)$. Let S be the sorted list of points in P sorted in increasing order of their Y-coordinates. Consider any pair of points (p, q) such that p belongs to P_L and q belongs to P_R and distance between p and q is at most d. There cannot be more than 10 points between p and q in the sorted list S.



- Proof idea: Consider a pair (p, q) such that p belongs to P_L and q belongs to P_R and distance between p and q is at most d. Let $y(p) \leq y(q)$ ($y(q) \leq y(p)$ will be symmetric).

Divide and Conquer: Examples

- Claim 3: Let P be all the points that have X-coordinate between $(x^* - d)$ and $(x^* + d)$. Let S be the sorted list of points in P sorted in increasing order of their Y-coordinates. Consider any pair of points (p, q) such that p belongs to P_L and q belongs to P_R and distance between p and q is at most d. There cannot be more than 10 points between p and q in the sorted list S.

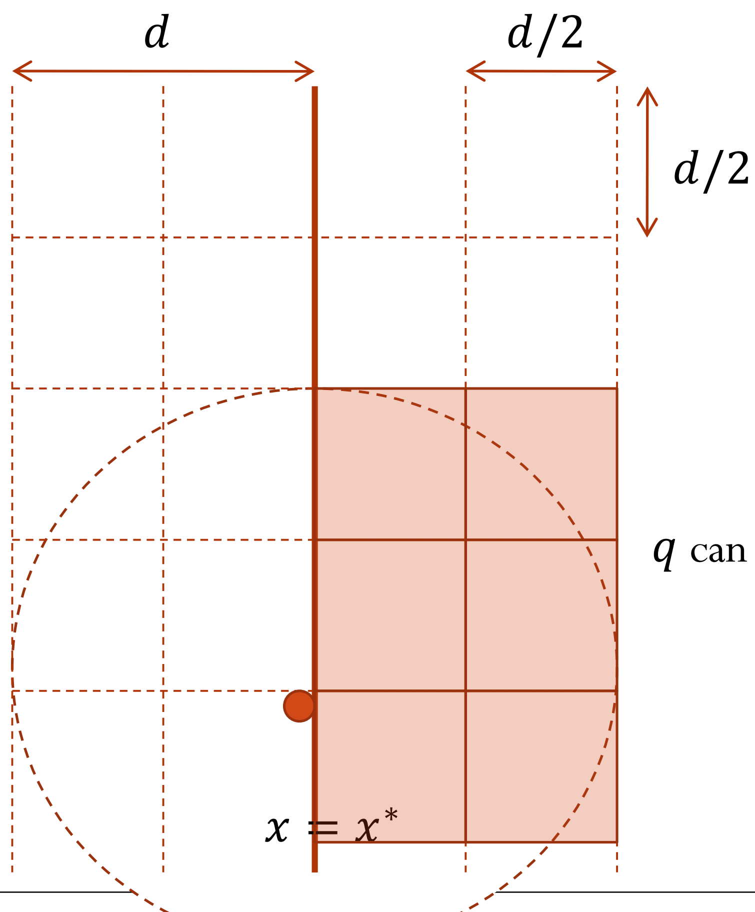


- Proof idea: Consider a pair (p, q) such that p belongs to P_L and q belongs to P_R and distance between p and q is at most d. Let $y(p) \leq y(q)$ ($y(q) \leq y(p)$ will be symmetric).

q can only belong to one of the shaded boxes

Divide and Conquer: Examples

- Claim 3: Let P be all the points that have X-coordinate between $(x^* - d)$ and $(x^* + d)$. Let S be the sorted list of points in P sorted in increasing order of their Y-coordinates. Consider any pair of points (p, q) such that p belongs to P_L and q belongs to P_R and distance between p and q is at most d. There cannot be more than 10 points between p and q in the sorted list S.



q can only belong to one of the shaded boxes

- Proof idea: Consider a pair (p, q) such that p belongs to P_L and q belongs to P_R and distance between p and q is at most d. Let $y(p) \leq y(q)$ ($y(q) \leq y(p)$ will be symmetric).

Divide and Conquer: Examples

ClosestPair(P)

- … *//Base cases*
- Sort the points in increasing order of X-coordinates. Pick the median point (x^*, y)
- Partition P into P_L (all points p with $x(p) < x^*$) and P_R (all points p with $x(p) \geq x^*$)
- $(p_1, q_1) \leftarrow$ ClosestPair(P_L)
- $(p_2, q_2) \leftarrow$ ClosestPair(P_R)
- Let (p, q) denote the pair with smaller distance and d be this distance
- Let S be the sorted list of point with X-coordinate between $(x^* - d)$ and $(x^* + d)$
- For $i = 1\ to\ |S|$
  - For $j = 1\ to\ 11$
    - If the distance between points $S[i]$ and $S[i+j]$ is smaller than d, then set (p, q) to be $(S[i], S[i+j])$ and d to be distance between $S[i]$ and $S[i+j]$
- Output(p, q)

- <u>Running time</u>:

Divide and Conquer: Examples

ClosestPair(P)

- … //*Base cases*
- Sort the points in increasing order of X-coordinates. Pick the median point (x^*, y)
- Partition P into P_L (all points p with $x(p) < x^*$) and P_R (all points p with $x(p) \geq x^*$)
- $(p_1, q_1) \leftarrow$ ClosestPair(P_L)
- $(p_2, q_2) \leftarrow$ ClosestPair(P_R)
- Let (p, q) denote the pair with smaller distance and d be this distance
- Let S be the sorted list of point with X-coordinate between $(x^* - d)$ and $(x^* + d)$
- For $i = 1\ to\ |S|$
  - For $j = 1\ to\ 11$
    - If the distance between points $S[i]$ and $S[i+j]$ is smaller than d, then set (p, q) to be $(S[i], S[i+j])$ and d to be distance between $S[i]$ and $S[i+j]$
- Output(p, q)

- <u>Running time</u>:
  - $T(n) = 2 * T(n/2) + O(n \log n), T(1) = O(1), T(2) = O(1)$
  - $T(n) = O(n\,(\log n)^2)$

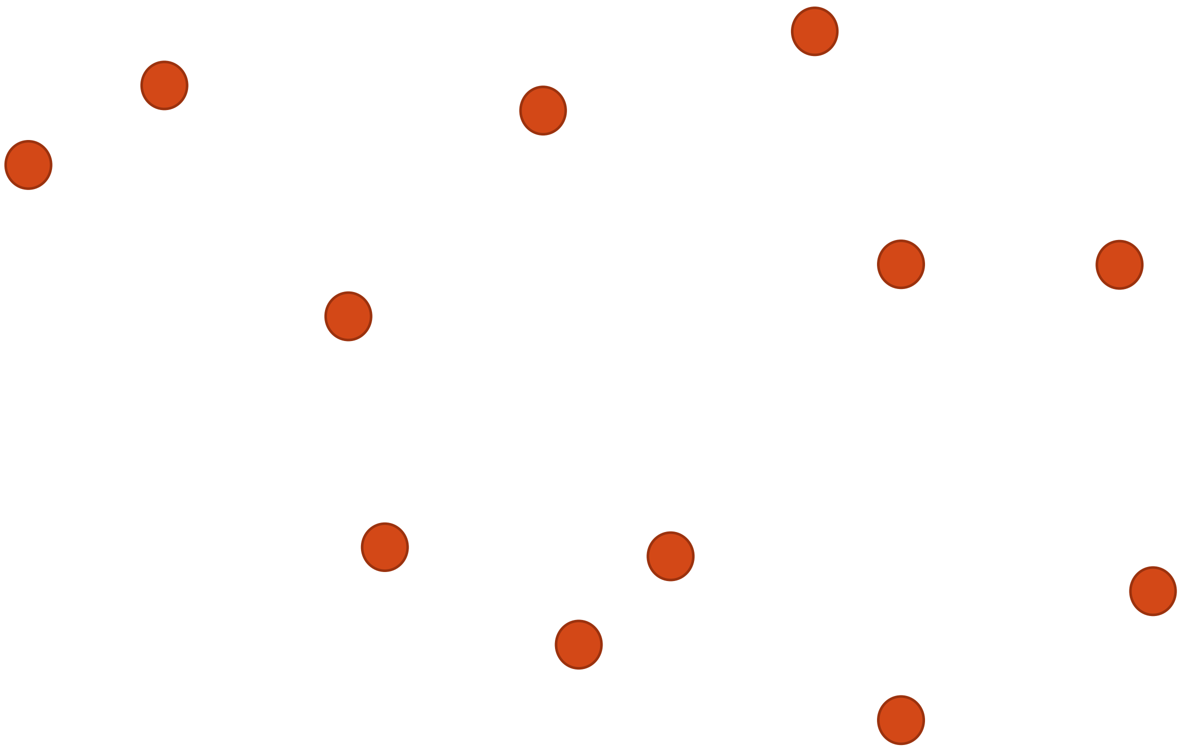
Divide and Conquer: Examples

ClosestPair(P)

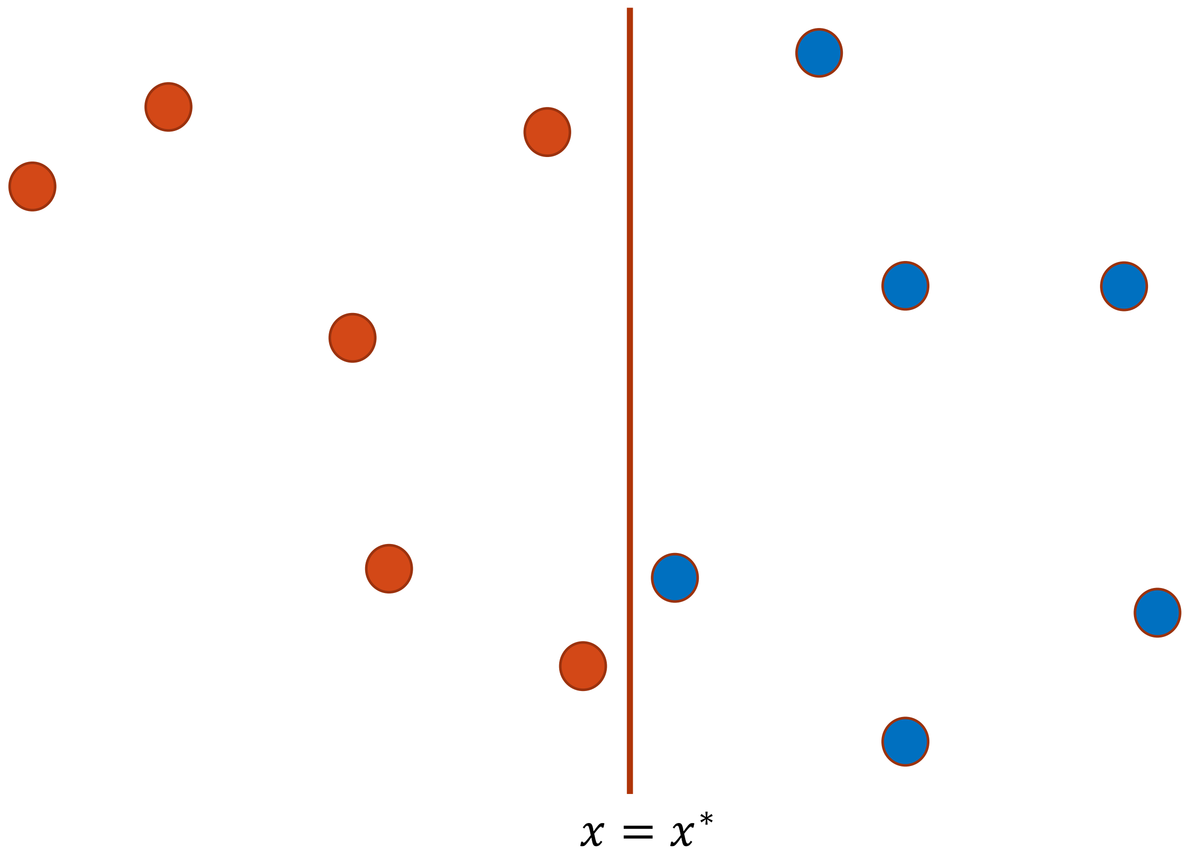
- … *//Base cases*
- Sort the points in increasing order of X-coordinates. Pick the median point (x^*, y)
- Partition P into P_L (all points p with $x(p) < x^*$) and P_R (all points p with $x(p) \geq x^*$)
- $(p_1, q_1) \leftarrow$ ClosestPair(P_L)
- $(p_2, q_2) \leftarrow$ ClosestPair(P_R)
- Let (p, q) denote the pair with smaller distance and d be this distance
- Let S be the sorted list of point with X-coordinate between $(x^* - d)$ and $(x^* + d)$
- For $i = 1\ to\ |S|$
    - For $j = 1\ to\ 11$
        - If the distance between points $S[i]$ and $S[i+j]$ is smaller than d, then set (p, q) to be $(S[i], S[i+j])$ and d to be distance between $S[i]$ and $S[i+j]$
- Output(p, q)

- Can we take the sorting out of the recursion?
- What is the running time we get in doing so?
    - We can get $O(n \log n)$.

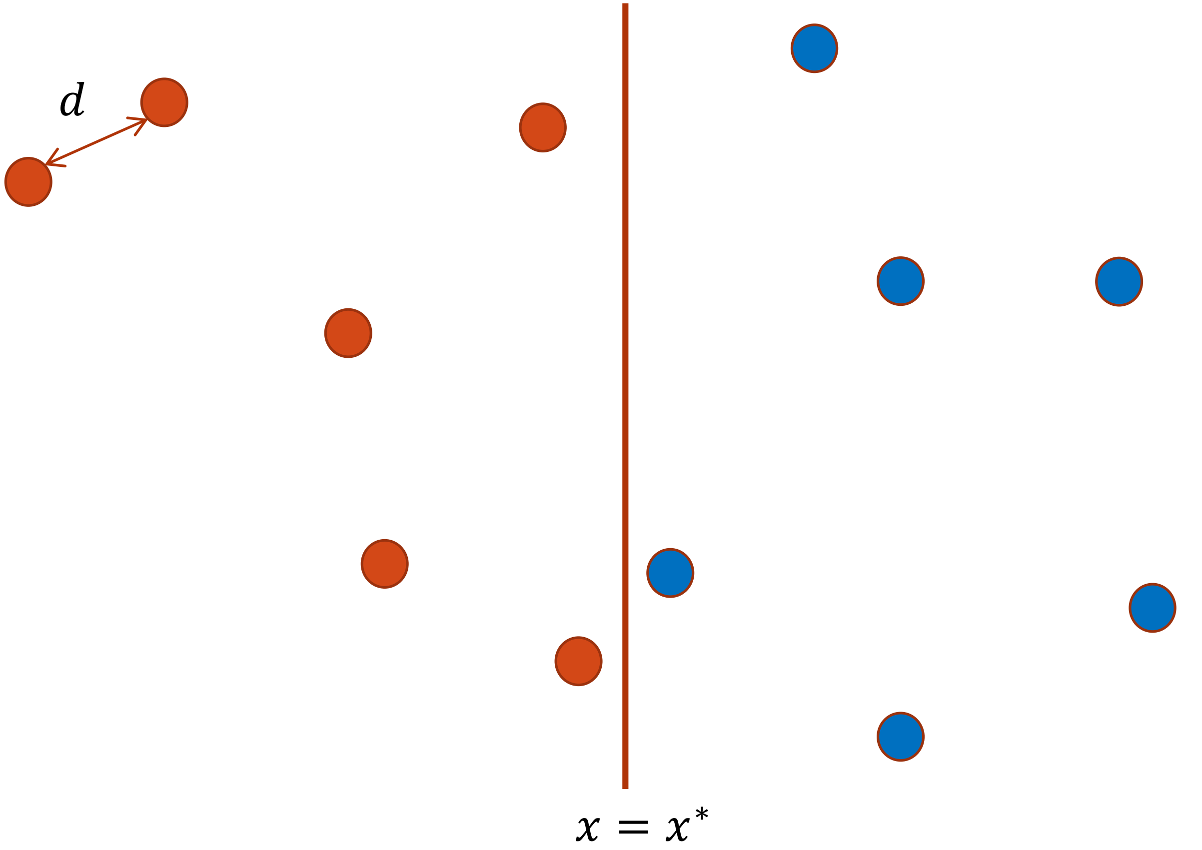
Divide and Conquer: Examples



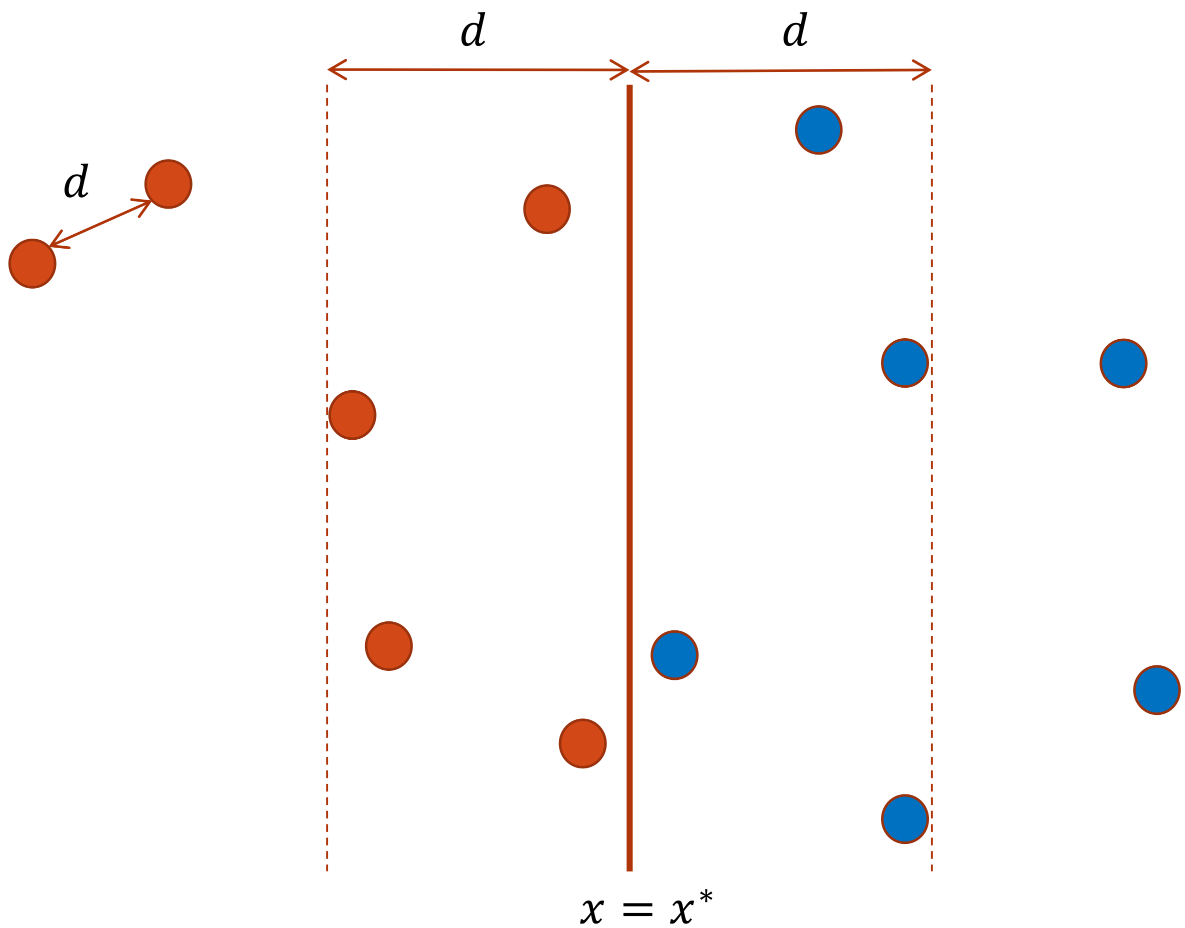
Divide and Conquer: Examples



Divide and Conquer: Examples

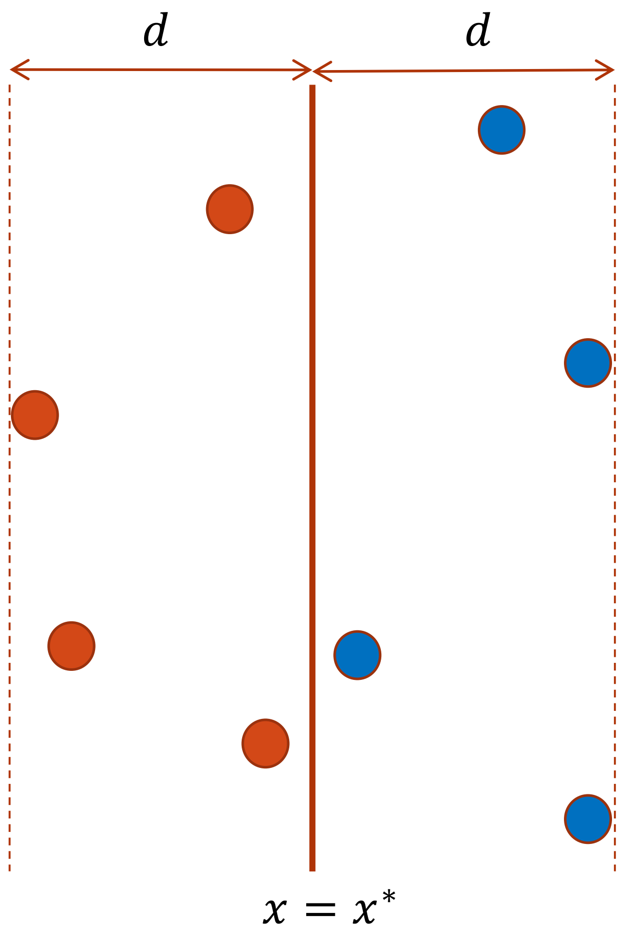


Divide and Conquer: Examples



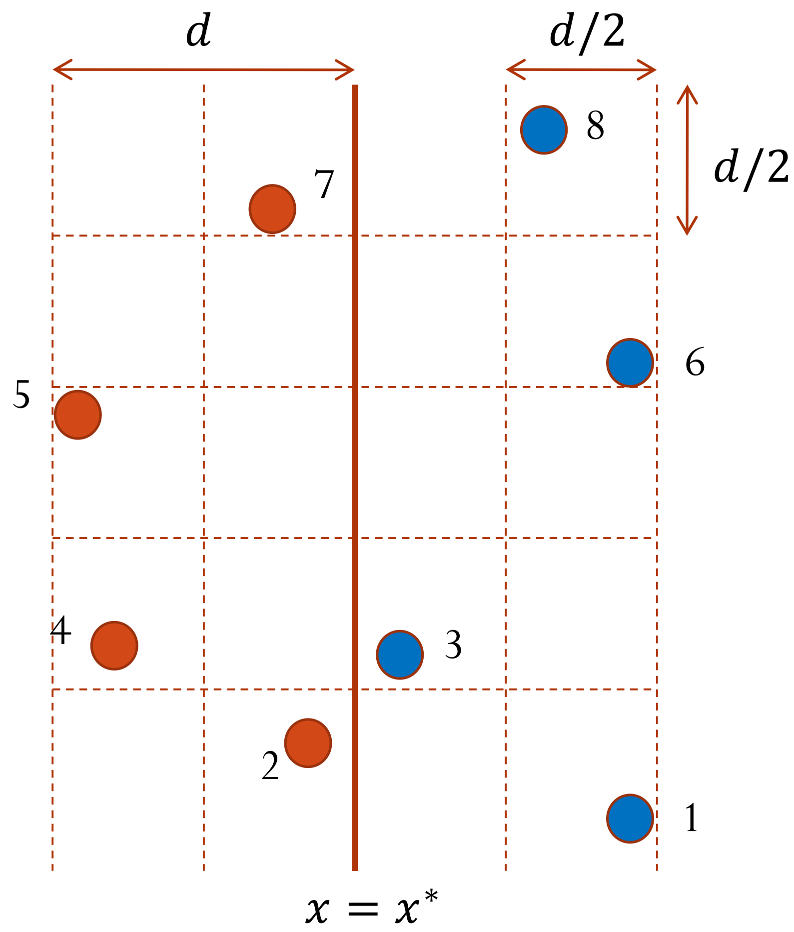
- Throw away points beyond $(x^* - d)$ and $(x^* + d)$

Divide and Conquer: Examples



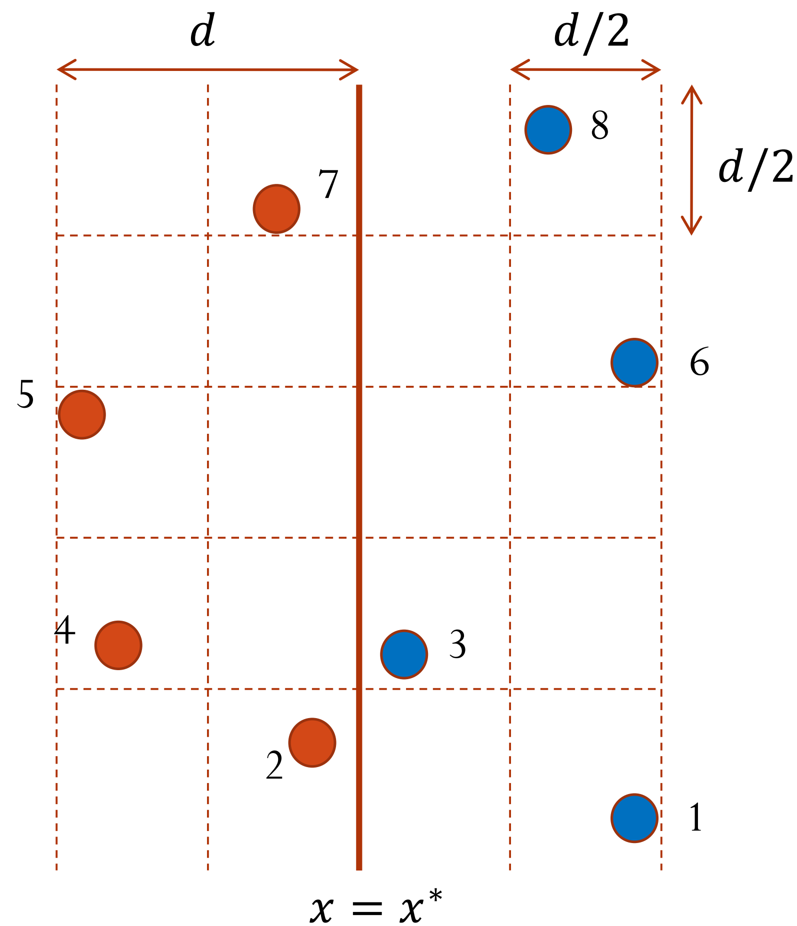
- Throw away points beyond $(x^* - d)$ and $(x^* + d)$

Divide and Conquer: Examples



- Consider the list of points sorted based on Y-coordinate.

Divide and Conquer: Examples



- Consider the list of points sorted based on Y-coordinate.
- Check the distance of a point in the list with the next 11 elements.

End