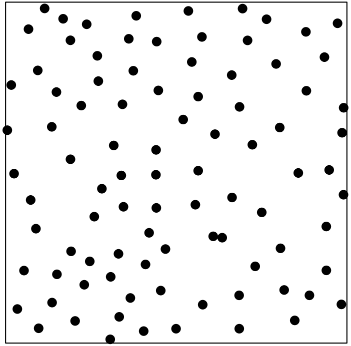
# 10.10

# Supplemental: Divide and conquer for closest pair

# Problem: Closest pair

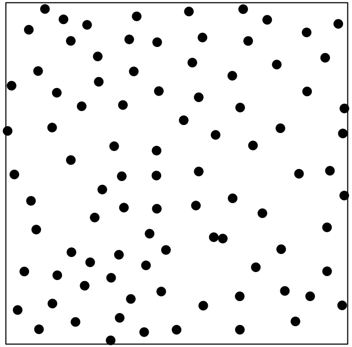
$\boldsymbol{P}$: Set of $\boldsymbol{n}$ distinct points in the plane.
Compute the two points $\boldsymbol{p}, \boldsymbol{q} \in \boldsymbol{P}$ that are closest together. Formally, compute

$$\arg \min_{\boldsymbol{p},\boldsymbol{q} \in \boldsymbol{P} : \boldsymbol{p} \neq \boldsymbol{q}} ||\boldsymbol{p} - \boldsymbol{q}||.$$



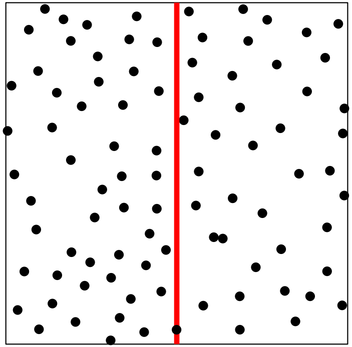
# Closest pair: Divide and conquer leads to a special case

1. $P = P_L \cup P_R$



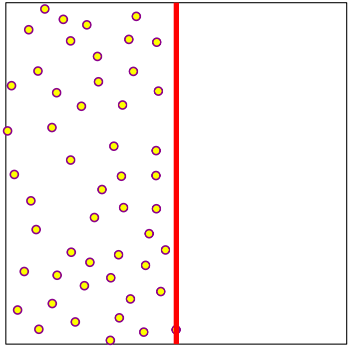
# Closest pair: Divide and conquer leads to a special case

1. $P = P_L \cup P_R$



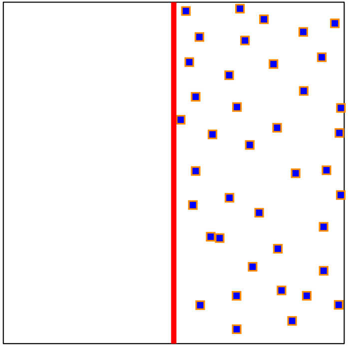
# Closest pair: Divide and conquer leads to a special case

1. $P = P_L \cup P_R$



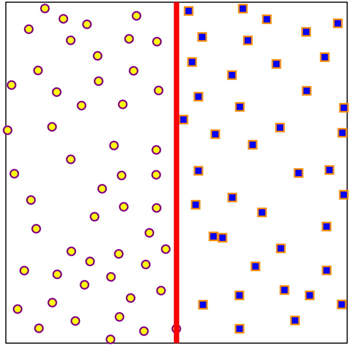
# Closest pair: Divide and conquer leads to a special case

1. $P = P_L \cup P_R$



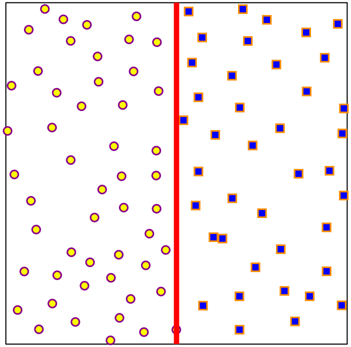
# Closest pair: Divide and conquer leads to a special case

1. $P = P_L \cup P_R$
2. $|P_L| = |P_R| = n/2$.
   $x(P_L) < 0$ and $x(P_R) > 0$.



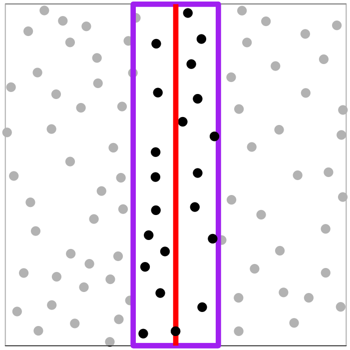
# Closest pair: Divide and conquer leads to a special case

1. $P = P_L \cup P_R$
2. $|P_L| = |P_R| = n/2$.
   $x(P_L) < 0$ and $x(P_R) > 0$.
3. Given $\ell = \min(\mathbf{cp}(P_L), \mathbf{cp}(P_R))$.



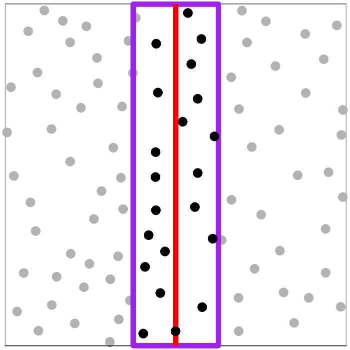
# Closest pair: Divide and conquer leads to a special case

1. $P = P_L \cup P_R$
2. $|P_L| = |P_R| = n/2$.
   $x(P_L) < 0$ and $x(P_R) > 0$.
3. Given $\ell = \min(\mathbf{cp}(P_L), \mathbf{cp}(P_R))$.
4. $P_m = \{p \in P \mid -\ell \leq x(p) \leq \ell\}$



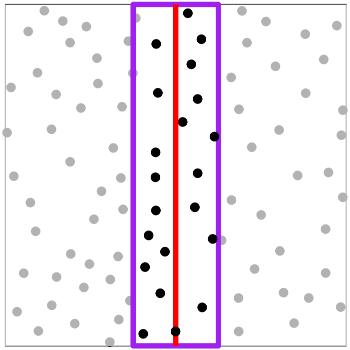
# Closest pair: Divide and conquer leads to a special case

1. $P = P_L \cup P_R$
2. $|P_L| = |P_R| = n/2$.
   $x(P_L) < 0$ and $x(P_R) > 0$.
3. Given $\ell = \min(\mathbf{cp}(P_L), \mathbf{cp}(P_R))$.
4. $P_m = \{p \in P \mid -\ell \leq x(p) \leq \ell\}$
5. Task: compute
   $\mathbf{cp}(P) = \min(\ell, \mathbf{cp}(p_M))$.



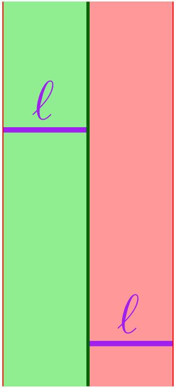
# Closest pair: Divide and conquer leads to a special case

1. $P = P_L \cup P_R$
2. $|P_L| = |P_R| = n/2$.
   $x(P_L) < 0$ and $x(P_R) > 0$.
3. Given $\ell = \min(\mathbf{cp}(P_L), \mathbf{cp}(P_R))$.
4. $P_m = \{p \in P \mid -\ell \leq x(p) \leq \ell\}$
5. Task: compute
   $\mathbf{cp}(P) = \min(\ell, \mathbf{cp}(p_M))$.
6. **Claim:** Closest pair in $P_m$ can be computed in $O(n \log n)$ time.



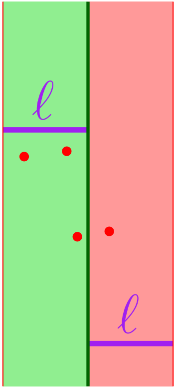
# An elevator can not be too full

## ...or $\mathbf{P_m}$ is well spread



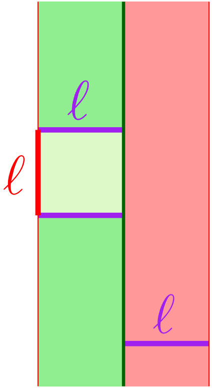
# An elevator can not be too full

## ...or $\mathbf{P_m}$ is well spread



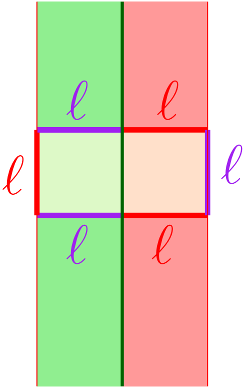
# An elevator can not be too full

## ...or $\mathbf{P_m}$ is well spread



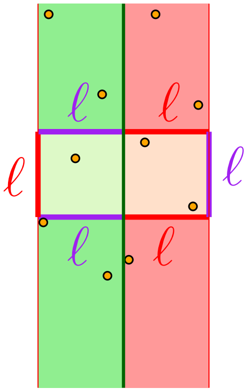
# An elevator can not be too full

## ...or $\mathbf{P_m}$ is well spread



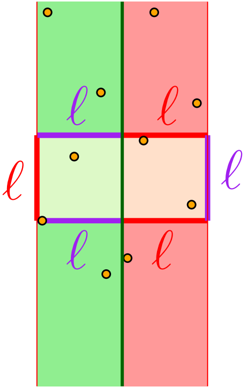
# An elevator can not be too full

## ...or $\mathbf{P_m}$ is well spread



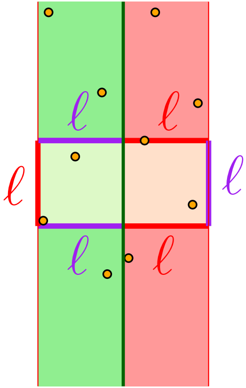
# An elevator can not be too full

...or $\mathbf{P_m}$ is well spread



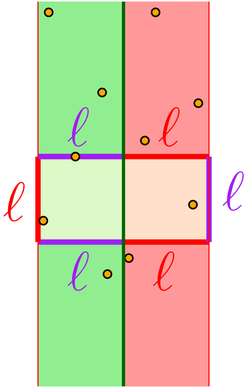
# An elevator can not be too full

## ...or $\mathbf{P_m}$ is well spread



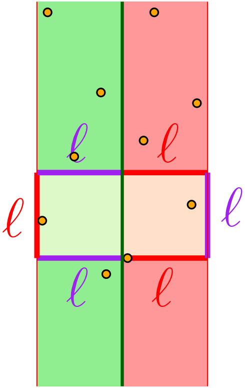
# An elevator can not be too full

...or $\mathbf{P_m}$ is well spread



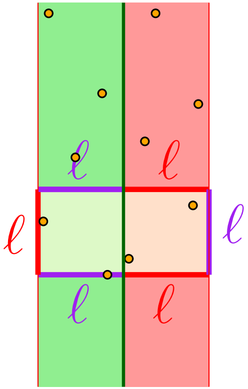
# An elevator can not be too full

## ...or $\mathbf{P_m}$ is well spread



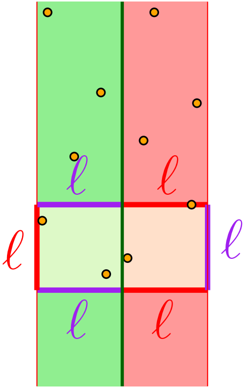
# An elevator can not be too full

## ...or $\mathbf{P_m}$ is well spread



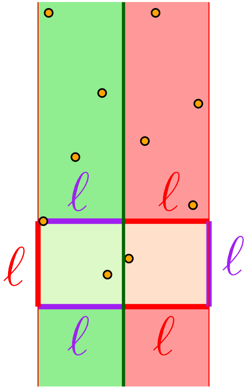
# An elevator can not be too full

## ...or $\mathbf{P_m}$ is well spread



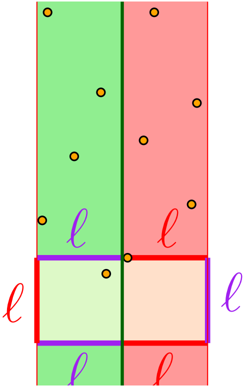
# An elevator can not be too full

...or $\mathbf{P_m}$ is well spread



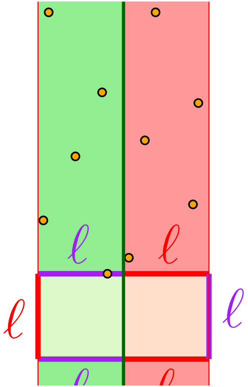
# An elevator can not be too full

## ...or $\mathbf{P_m}$ is well spread



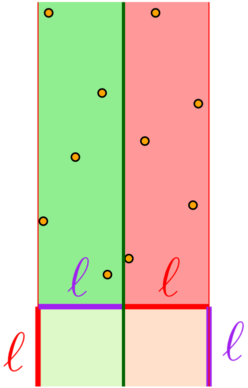
# An elevator can not be too full

...or $\mathbf{P_m}$ is well spread



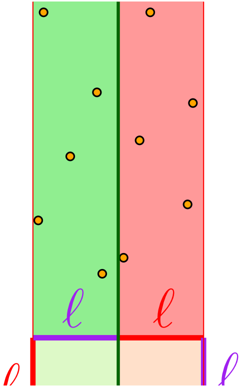
# An elevator can not be too full

...or $\mathbf{P_m}$ is well spread



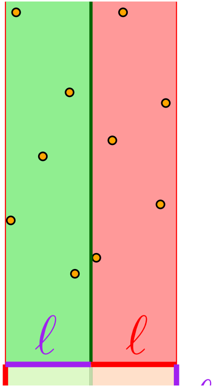
# An elevator can not be too full

## ...or $\mathbf{P_m}$ is well spread



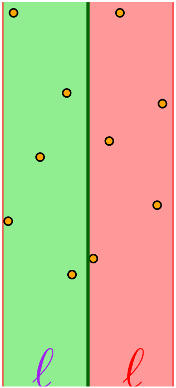
# An elevator can not be too full

## ...or $\mathbf{P_m}$ is well spread



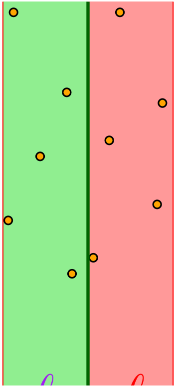
# An elevator can not be too full

## ...or $\mathbf{P_m}$ is well spread



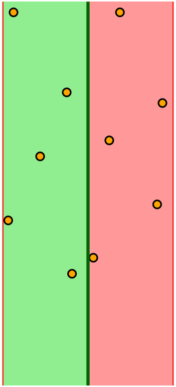
# An elevator can not be too full

## ...or $\mathbf{P_m}$ is well spread



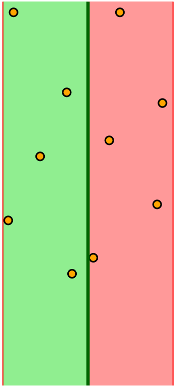
# An elevator can not be too full

## ...or $\mathbf{P_m}$ is well spread



# An elevator can not be too full

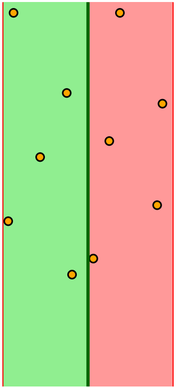
## ...or $\mathbf{P_m}$ is well spread



Closet pair in $P_m$ can be computed in $O(n \log n)$ time.

# An elevator can not be too full

## ...or $\mathbf{P_m}$ is well spread



Closet pair in $P_m$ can be computed in $O(n \log n)$ time.

Closet pair in $P_m$ can be computed in $O(n)$ time, if $P$ is presorted by $y$-order.

# Closest pair: Algorithm

## CPDInner = ClosestPairDistance

```
CPDInner( P = {p_1, ..., p_n} ):
    if |P| = O(1) then compute by brute force
    x* = median(x(p_1), ..., x(p_n)).
    P_L ← {p ∈ P | x(p) ≤ x*}
    P_R ← {p ∈ P | x(p) > x*}
    ℓ_L = CPDInner(P_L)
    ℓ_R = CPDInner(P_R)
    ℓ = min(ℓ_L, ℓ_R).
    P_m = {p ∈ P | x* − ℓ ≤ x(p) ≤ x* + ℓ}
    ℓ_M = call alg. closet-pair distance for special case on P_m.
    return min(ℓ, ℓ_m).
```

```
CPD( P = {p_1, ..., p_n} ):
    return CPDInner(P)
```

# Closest pair algorithm

## Lemma

*Given a set $P$ of $n$ points in the plane, one can compute the closet pair distance in $P$ in $O(n \log^2 n)$ time.*

# Closest pair: Algorithm

**CPDInner = ClosestPairDistance**

```
CPDInner( P = {p_1, ..., p_n} ):
    if |P| = O(1) then compute by brute force
    x* = median(x(p_1), ..., x(p_n)).
    P_L ← {p ∈ P | x(p) ≤ x*}
    P_R ← {p ∈ P | x(p) > x*}
    ℓ_L = CPDInner(P_L)
    ℓ_R = CPDInner(P_R)
    ℓ = min(ℓ_L, ℓ_R).
    P_m = {p ∈ P | x* − ℓ ≤ x(p) ≤ x* + ℓ}
    ℓ_M = call alg. closet-pair distance for special case on P_m.
    return min(ℓ, ℓ_m).
```

```
CPD( P = {p_1, ..., p_n} ):
    Sort P by x-order. Sort P by y-order
    return CPDInner(P)
```

# Closest pair

## Theorem

*Given a set $P$ of $n$ points in the plane, one can compute the closet pair distance in $P$ in $O(n \log n)$ time.*

# Wait wait... one can do better

Rabin showed that if we allow the floor function, and randomization, one can do better:

## Theorem

*Given a set $P$ of $n$ points in the plane, one can compute the closet pair distance in $P$ in $O(n)$ time.*

# THE END

...

# (for now)