



## Datenstrukturen, Algorithmen und Programmierung 2 (DAP2)

## Teile & Herrsche

### *Teile & Herrsche (Divide & Conquer)*

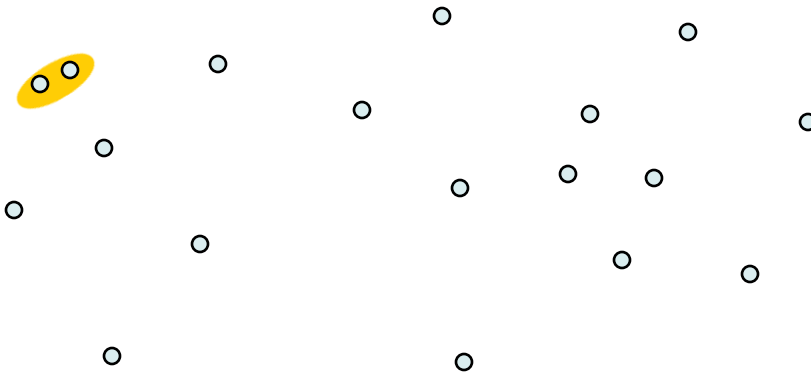
- Teile Eingabe in mehrere Teile auf
- Löse das Problem rekursiv auf den Teilen
- Füge die Teillösungen zu einer Gesamtlösung zusammen

## Teile & Herrsche

### *Nächste Paare*

- Problem: Finde nächstes Paar von Punkten in der Ebene
- Eingabe: Menge  $P$  von  $n$  Punkten in der Ebene
- Ausgabe: Paar  $(q,r) \in P \times P$ ,  $q \neq r$ , mit geringstem Abstand

### *Beispiel*

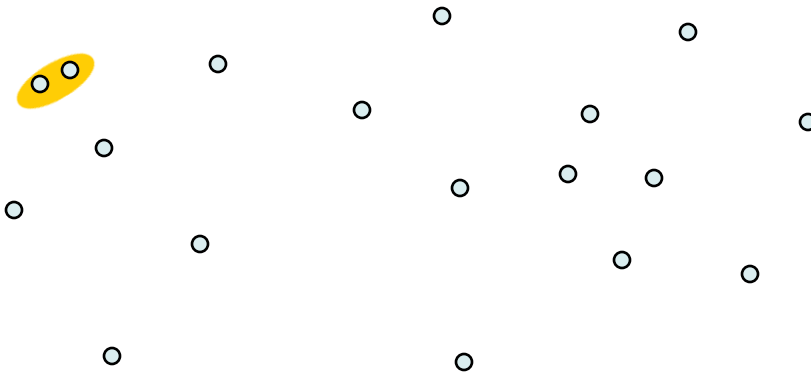


## Teile & Herrsche

### *Nächste Paare*

- $\|q-r\|$  bezeichne den (euklidischen) Abstand zwischen  $q$  und  $r$

### *Beispiel*

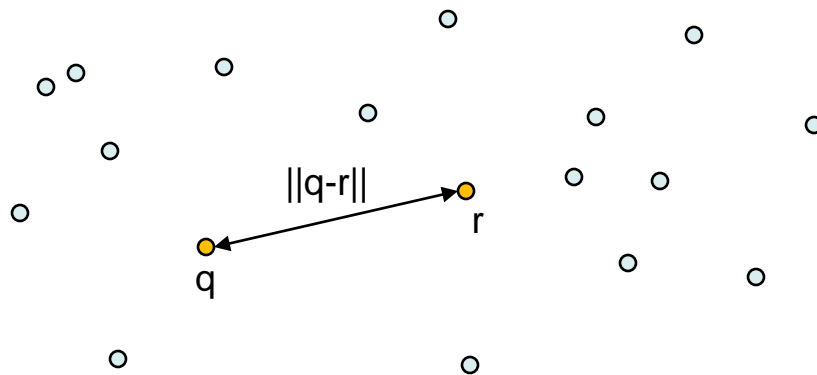


## Teile & Herrsche

### Nächste Paare

- $\|q-r\|$  bezeichne den (euklidischen) Abstand zwischen  $q$  und  $r$
- $\|q-r\|^2 = (q_1 - r_1)^2 + (q_2 - r_2)^2$ , wobei  $q=(q_1, q_2)$  und  $r=(r_1, r_2)$  die Positionsvektoren der Punkte  $q$  und  $r$  sind

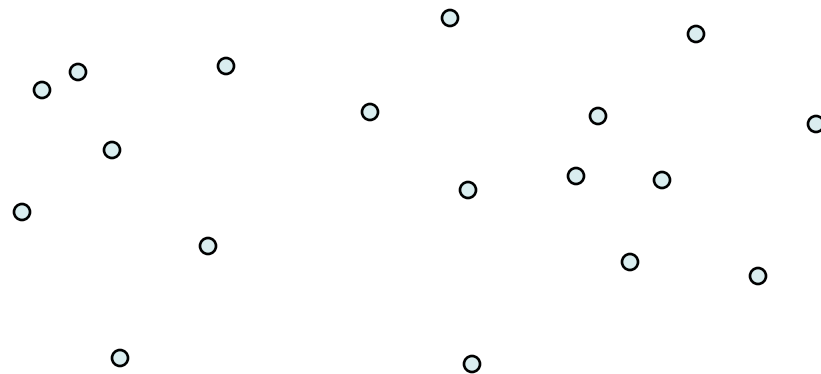
### Beispiel



## Teile & Herrsche

### *Grober Plan (Details siehe Vollversion)*

1. Sortiere Punkte nach x-Koordinate
2. Teile P in die Menge Q der  $n/2$  Punkte mit kleinsten x-Koordinaten und die Menge R der  $n/2$  Punkte mit größten x-Koordinaten auf
3. Löse das Problem rekursiv
4. Füge zusammen



## Teile & Herrsche

ClosestPair1 (A, a, b)

1. **if**  $b-a < 4$  **then return** nächstes Paar aus  $A[a..b]$ 
  - Löse „brute-force“
2. MergeSort(A, a, b)
  - Sortiere nach x-Koordinate
3.  $m \leftarrow \lfloor a+b/2 \rfloor$
4.  $(q_1^*, q_2^*) \leftarrow \text{ClosestPair1}(A, a, m)$ 
  - Löse rekursiv für Q
5.  $(r_1^*, r_2^*) \leftarrow \text{ClosestPair1}(A, m+1, b)$ 
  - Löse rekursiv für R
6. **if**  $\|q_1^* - q_2^*\| < \|r_1^* - r_2^*\|$  **then**  $(p_1^*, p_2^*) \leftarrow (q_1^*, q_2^*)$ 
  - Bestimme bessere der
7. **else**  $(p_1^*, p_2^*) \leftarrow (r_1^*, r_2^*)$ 
  - beiden Lösungen
8. **for**  $i \leftarrow a$  **to**  $m$  **do**
  - Überprüfe Paare
9.     **for**  $j \leftarrow m+1$  **to**  $b$  **do**
  - aus  $Q \times R$
10.       **if**  $\|A[i] - A[j]\| < \|p_1^* - p_2^*\|$  **then**  $(p_1^*, p_2^*) \leftarrow (A[i], A[j])$
11. **return**  $(p_1^*, p_2^*)$

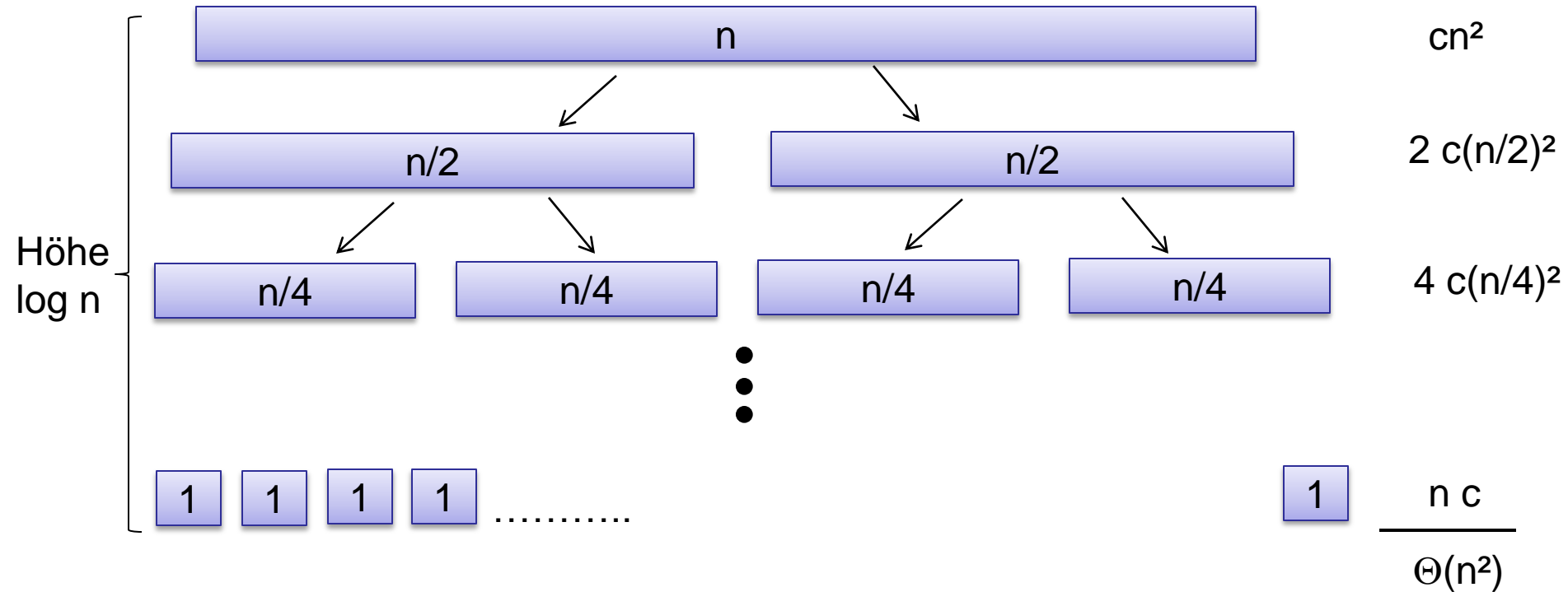
## Teile & Herrsche

### *Laufzeit ClosestPair1*

$$T(n) = \begin{cases} c & , \text{ falls } n < 4 \\ 2 T(n/2) + cn^2 & , \text{ falls } n \geq 4 \end{cases}$$

## Teile & Herrsche

Auflösen von  $T(n) = 2 T(n/2) + cn^2$



## Teile & Herrsche

### *Können wir den Algorithmus verbessern?*

- Wo ist der Flaschenhals?
  - Das Überprüfen der Paare aus  $Q \in R$  dauert  $\Theta(n^2)$
  - alle anderen Schritte brauchen  $O(n \log n)$  Zeit

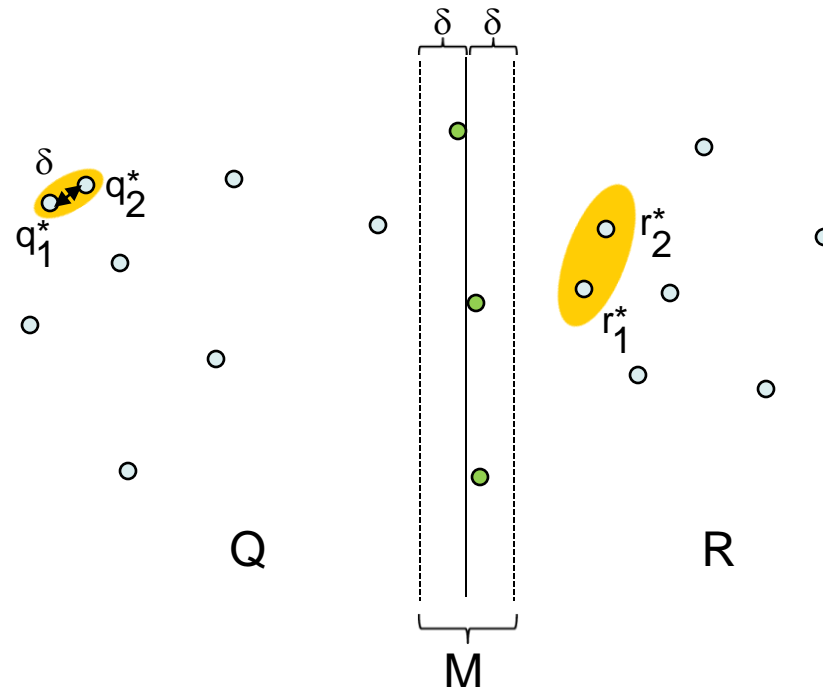
### *Müssen wir alle Paare überprüfen?*

- Nein!
- Aber die Frage ist, ob wir die Anzahl der überprüften Paare auf  $o(n^2)$  reduzieren können und vielleicht sogar  $O(n)$  erreichen können

## Teile & Herrsche

### Erste Verbesserung

- Ist ein Punkt weiter als  $\delta = \min \{ \|q_1^* - q_2^*\|, \|r_1^* - r_2^*\| \}$  von der „Trennlinie zwischen Q und R“ entfernt, so müssen wir ihn nicht testen
- (Die Trennlinie ist an der x-Koordinate von A[m])



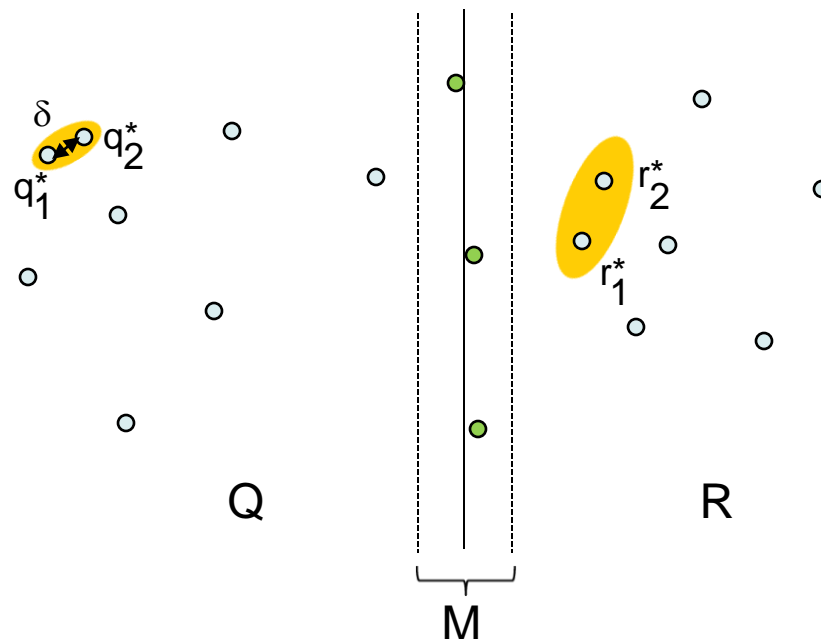
Sei M die Menge der Punkte, deren Abstand zur Trennlinie höchstens  $\delta$  ist

Hoffnung: Es kann nicht sehr viele Punkte in  $M$  geben, die zu einem vorgegeben  $p \in M$  geringen Abstand in  $y$ -Koordinate haben. (ansonsten finden wir zwei Punkte in  $Q$  oder  $R$  mit geringerem Abstand als  $\delta$ ).  
Widerspruch!

## Teile & Herrsche

### Zweite Verbesserung

- Wir müssen nur Punkte in  $M$  miteinander vergleichen, deren Abstand in  $y$ -Koordinate  $\leq \delta$  ist
- Idee: Sortiere Punkte in  $M$  zuerst nach  $y$ -Koordinate und vergleiche jeden Punkt mit seinen Vorgängern, bis der erste Punkt Abstand  $> \delta$  hat



## Teile & Herrsche

ClosestPair2 (A, a, b)

1. **if**  $b-a < 4$  **then return** nächstes Paar aus  $A[a..b]$
2. MergeSort(A, a, b) ➤ Sortiere nach x-Koordinate
3.  $m \leftarrow \lfloor a+b/2 \rfloor$
4.  $(q_1^*, q_2^*) \leftarrow \text{ClosestPair2}(A, a, m)$
5.  $(r_1^*, r_2^*) \leftarrow \text{ClosestPair2}(A, m+1, b)$
6. **if**  $\|q_1^* - q_2^*\| < \|r_1^* - r_2^*\|$  **then**  $(p_1^*, p_2^*) \leftarrow (q_1^*, q_2^*)$
7. **else**  $(p_1^*, p_2^*) \leftarrow (r_1^*, r_2^*)$
8. \*\*\*\* Merge \*\*\*\* ➤ nächste Seite

## Teile & Herrsche

ClosestPair2 (A, a, b)

1. **if**  $b-a < 4$  **then return** nächstes Paar aus  $A[a..b]$
2. MergeSort(A, a, b) ➤ Sortiere nach x-Koordinate
3.  $m \leftarrow \lfloor a+b/2 \rfloor$
4.  $(q_1^*, q_2^*) \leftarrow \text{ClosestPair2}(A, a, m)$
5.  $(r_1^*, r_2^*) \leftarrow \text{ClosestPair2}(A, m+1, b)$
6. **if**  $\|q_1^* - q_2^*\| < \|r_1^* - r_2^*\|$  **then**  $(p_1^*, p_2^*) \leftarrow (q_1^*, q_2^*)$
7. **else**  $(p_1^*, p_2^*) \leftarrow (r_1^*, r_2^*)$
8. \*\*\*\* Merge \*\*\*\* ➤ nächste Seite

## Teile & Herrsche

\*\*\*\* Merge \*\*\*\*

1.  $w \leftarrow A[m].x$  ➤  $w$  ist x-Koordinate von  $A[m]$
2. Sei  $M$  die Menge der Punkte aus  $A[a\dots b]$  deren x-Koordinate im Intervall  $[w-\delta, w+\delta]$  liegt ➤ Annahme:  $M$  ist Feld
3. MergeSort( $M, 1, \text{length}[M]$ ) ➤ Sortiere nach y-Koordinate
4. **for**  $i \leftarrow 1$  **to**  $\text{length}[M]$  **do**
5.      $j \leftarrow i+1$
6.     **while**  $M[j].y \leq M[i].y + \delta$  and  $j \leq \text{length}[M]$  **do**
7.         **if**  $\|M[i] - M[j]\| < \|p_1^* - p_2^*\|$  **then**  $(p_1^*, p_2^*) \leftarrow (M[i], M[j])$
8.          $j \leftarrow j+1$
9. **return**  $(p_1^*, p_2^*)$

## Teile & Herrsche

### *Lemma 16*

- Algorithmus  $\text{ClosestPair2}(A,a,b)$  gibt das nächste Paar aus  $A[a..b]$  zurück.

### *Beweis:*

- Induktion über den Abstand  $n$  zwischen  $a$  und  $b$

- Induktionsanfang:

$$n=b-a<4$$

In diesem Fall wird in Zeile 1 das nächste Paar aus  $A[a..b]$  zurückgegeben.

- Induktionsvoraussetzung:

Für alle  $a, b$  mit  $b-a < n$  berechnet  $\text{ClosestPair2}(A,a,b)$  das nächste Paar aus  $A[a,b]$ .

## Teile & Herrsche

### Lemma 16

- Algorithmus ClosestPair2(A,a,b) gibt das nächste Paar aus A[a..b] zurück.

### Beweis:

- Induktionsvoraussetzung:

Für alle a, b mit  $b-a < n$  berechnet ClosestPair2(A,a,b) das nächste Paar aus A[a,b].

- Induktionsschluss: Betrachte a, b mit  $a-b=n$

Der Algorithmus sortiert zunächst die Punkte nach x-Koordinate. Dann wird

für  $m = \left\lfloor \frac{a+b}{2} \right\rfloor$  rekursiv ClosestPair2 aufgerufen. Da  $a \leq m < b$  gilt, greift

(I.V.) und der Algorithmus berechnet die nächsten Paare A[a...m] und A[m+1...b] korrekt. In Zeile 4 und 7 wird  $(p_1^*, p_2^*)$  dann auf das nächste der beiden Paare gesetzt.

## Teile & Herrsche

- Induktionsschluss (Fortsetzung 1):

Sind beide Punkte des nächsten Paares von  $A[a\dots b]$  aus  $A[a\dots m]$  oder beide aus  $A[m+1\dots b]$ , so ist  $(p_1^*, p_2^*)$  bereits das nächste Paar von  $A[a\dots b]$  und wird am Ende des Algorithmus korrekt ausgegeben, da sich  $(p_1^*, p_2^*)$  nur dann ändert, wenn man tatsächlich ein nächstes Paar findet.

Betrachten wird also den Fall, dass o.B.d.A. für das nächste Paar  $(p, q)$   $p \in A[a\dots m]$  und  $q \in A[m+1\dots b]$  gilt. Zunächst einmal wird in Zeile 2 von von Menge  $M$  auf die Menge der Punkte gesetzt, deren  $x$ -Koordinate im Intervall  $[w - \delta, w + \delta]$  liegt mit  $w = A[m] \cdot x$ . Alle Punkte aus  $A[a\dots m]$  haben  $x$ -Koordinaten  $\leq w$  und alle Punkte aus  $A[m+1\dots b]$  haben  $x$ -Koordinaten  $\geq w$ . Ist nun mind. Ein Punkt des nächsten Paares  $(p, q)$  aus  $A[a\dots b]$  nicht in  $M$ , so schneidet ihre direkte Verbindung die Vertikale  $x=w$ .

## Teile & Herrsche

- Induktionsschluss (Fortsetzung 2):

Sei nun o.B.d.A.  $p \notin M$ . Dann ist der Abstand von  $p$  zum Schnittpunkt mit dieser Vertikale bereits größer als  $\delta$ , dem Abstand von  $(p_1^*, p_2^*)$ . Also kann  $p, q$  nicht nächstes Paar sein. Sind nun  $p, q$  in  $M$ , so werden sie nach  $y$ -Koordinaten sortiert. Dann werden alle Paare aus  $M$  mit  $(p_1^*, p_2^*)$  verglichen, deren Abstand in der  $y$ -Koordinate  $\leq \delta$  ist. Auch hier gilt, dass man nur solche Paare vergleichen muss, da für alle anderen Paare der Abstand  $> \delta$  ist. Somit folgt die Korrektheit des Algorithmus.

## Teile & Herrsche

### *Lemma 17*

- Algorithmus `ClosestPair2(A,a,b)` hat eine Laufzeit von  $O(n \log^2 n)$ , wobei  $n=b-a+1$  ist.

Wir zeigen, dass die Laufzeit eines Aufrufs von `ClosestPair2(A,a,b)` ohne den Aufwand für die Rekursion mit  $n=b-a+1$  durch  $O(n \log n)$  beschränkt ist. Dies gilt offensichtlich für alle Aufrufe bis auf die Zeilen 4-8 des Menge-Teils. Wir zeigen nun folgende Behauptung aus der folgt, dass die Laufzeit für diesen Teil  $O(n)$  ist.

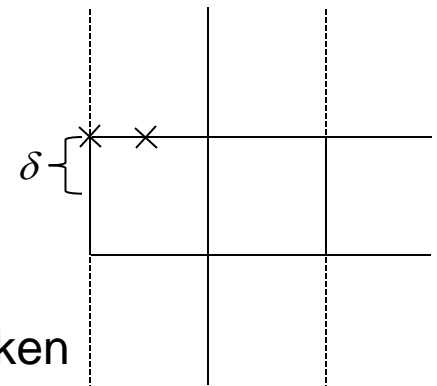
## Teile & Herrsche

### Behauptung

- Für jeden Punkt  $p \in M$  gibt es maximal 8 Punkte aus  $M$ , deren  $y$ -Koordinate aus  $[p \cdot y, p \cdot y + \delta]$  ist.

### Beweis

- Wir betrachten das Rechteck, das durch die Eckpunkte  $(w - \delta, p \cdot y)$ ,  $(w + \delta, p \cdot y)$ ,  $(w - \delta, p \cdot y + \delta)$ ,  $(w + \delta, p \cdot y + \delta)$  definiert wird. Dieses unterteilen wir in 8 Quadrate mit Seitenlängen  $\frac{\delta}{2}$ . Sind nun in  $M$  mehr als 8 Punkte, deren  $y$ -Koordinate aus  $[p \cdot y, p \cdot y + \delta]$  ist, so liegen mindestens zwei dieser innerhalb eines solchen Quadrates mit Seitenlänge  $\frac{\delta}{2}$ . Hat ein Punkt  $p$   $x$ -Koordinate genau  $w$ , so liegt er im linken Quadrat, wenn  $p \in A[a\dots m]$  ist, und ansonsten im rechten Quadrat.



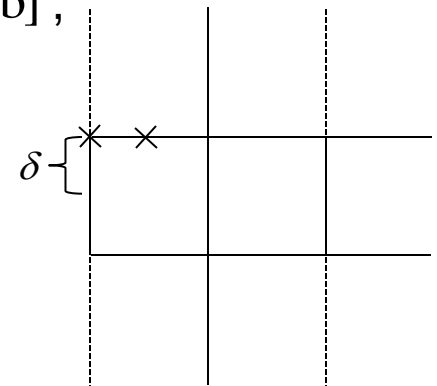
## Teile & Herrsche

### Behauptung

- Für jeden Punkt  $p \in M$  gibt es maximal 8 Punkte aus  $M$ , deren  $y$ -Koordinate aus  $[p \cdot y, p \cdot y + \delta]$  ist.

### Beweis (Fortsetzung)

- Alle anderen sind Tie-breaks beliebig. Dann ist aber ihr Abstand höchstens  $\frac{\delta}{2} \cdot \sqrt{2} < \delta$  und sie liegen beide in  $A[a \dots m]$  oder  $A[m+1 \dots b]$ , da die Quadrate jeweils auf einer der Vertikalen  $x=w$  liegen.  
Widerspruch zu Definition von  $\delta$ .



## Teile & Herrsche

Es ergibt sich also als Laufzeitrekursion für eine geeignete Konstante  $c$ :

$$T(n) \leq 2 \cdot T\left(\frac{n}{2}\right) + c$$

$$T(4) = c$$

z.z.:  $T(n) \leq c \cdot n \log^2 n$

(I.A.)  $T(4) = c \leq 16c$

(I.V.)  $\forall m < n : T(m) \leq c \cdot m \log^2 m$

(I.S.)  $T(n) \leq 2T\left(\frac{n}{2}\right) + c \cdot n \log n$

$$\begin{aligned} &\stackrel{\text{(I.V.)}}{\leq} 2c \frac{n}{2} \cdot \log^2\left(\frac{n}{2}\right) + c \cdot n \log n \\ &\leq cn \cdot \log n (\log n - 1) + c \cdot n \log n \\ &= cn \log^2 n \end{aligned}$$

## Teile & Herrsche

### Satz 18

- Das nächste Paar einer Menge von  $n$  Punkten in der Ebene kann in  $O(n \log^2 n)$  Zeit berechnet werden.