

# Vorlesung Algorithmische Geometrie

## Well-Separated Pair Decomposition

INSTITUT FÜR THEORETISCHE INFORMATIK · FAKULTÄT FÜR INFORMATIK

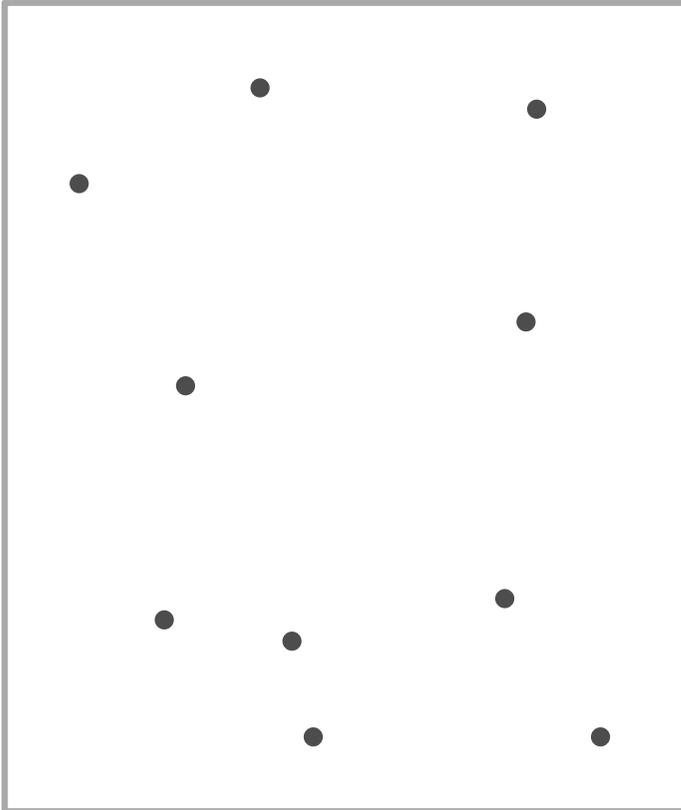
Martin Nöllenburg  
03.07.2012



# Motivation: Spanner

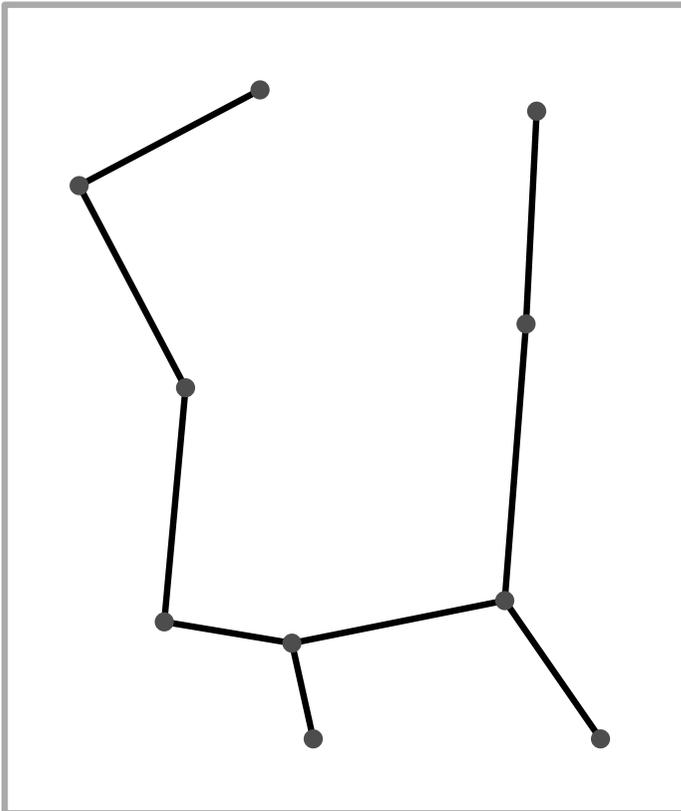
## Aufgabe:

Eine Menge von Städten soll über ein neues Straßennetz miteinander verbunden werden.



## Aufgabe:

Eine Menge von Städten soll über ein neues Straßennetz miteinander verbunden werden.

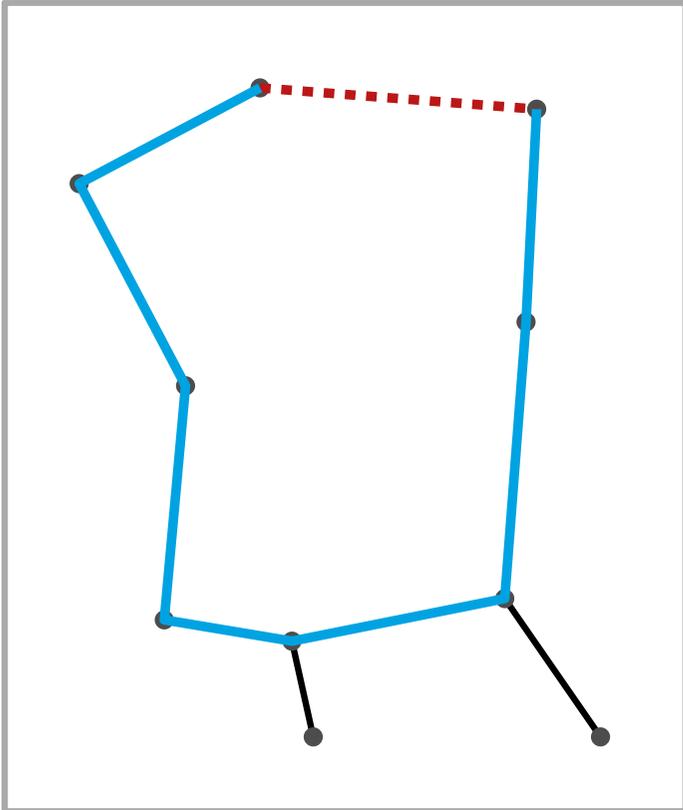


1. Idee: Euklid. min. Spannbaum

## Aufgabe:

Eine Menge von Städten soll über ein neues Straßennetz miteinander verbunden werden.

Allerdings soll für kein Paar  $(x, y)$  der Weg im Straßennetz viel länger als die Distanz  $\|xy\|$  sein.

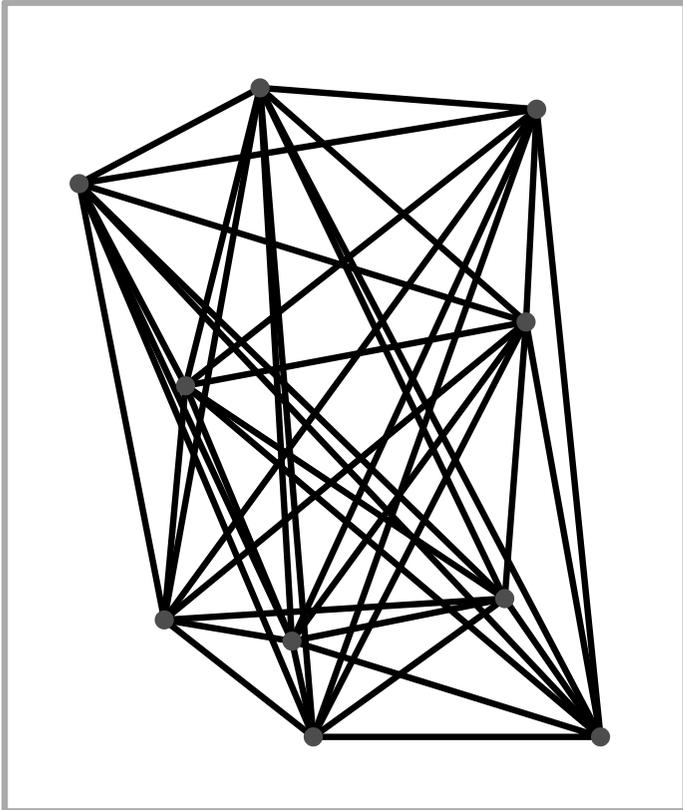


1. Idee: Euklid. min. Spannbaum

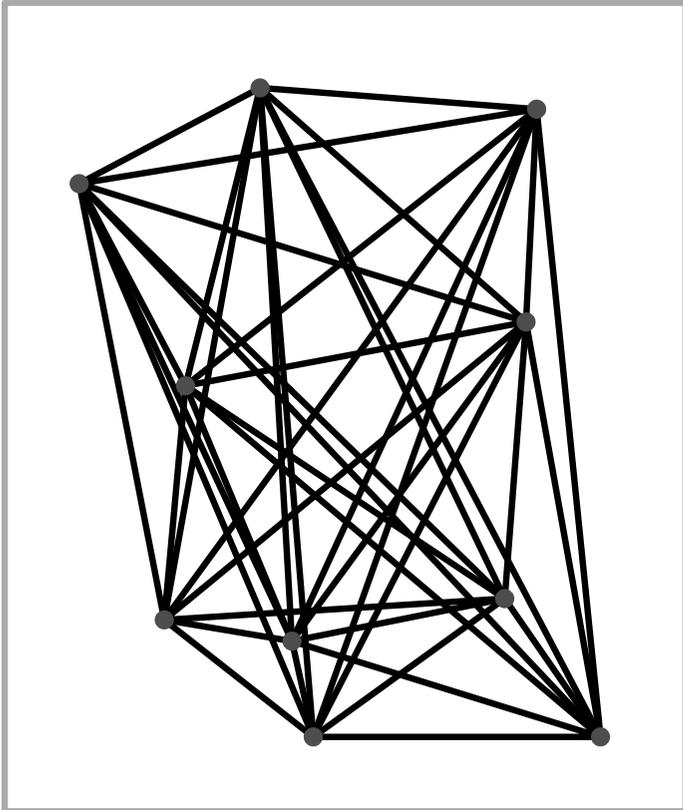
## Aufgabe:

Eine Menge von Städten soll über ein neues Straßennetz miteinander verbunden werden.

Allerdings soll für kein Paar  $(x, y)$  der Weg im Straßennetz viel länger als die Distanz  $\|xy\|$  sein.



1. Idee: Euklid. min. Spannbaum
2. Idee: vollständiger Graph



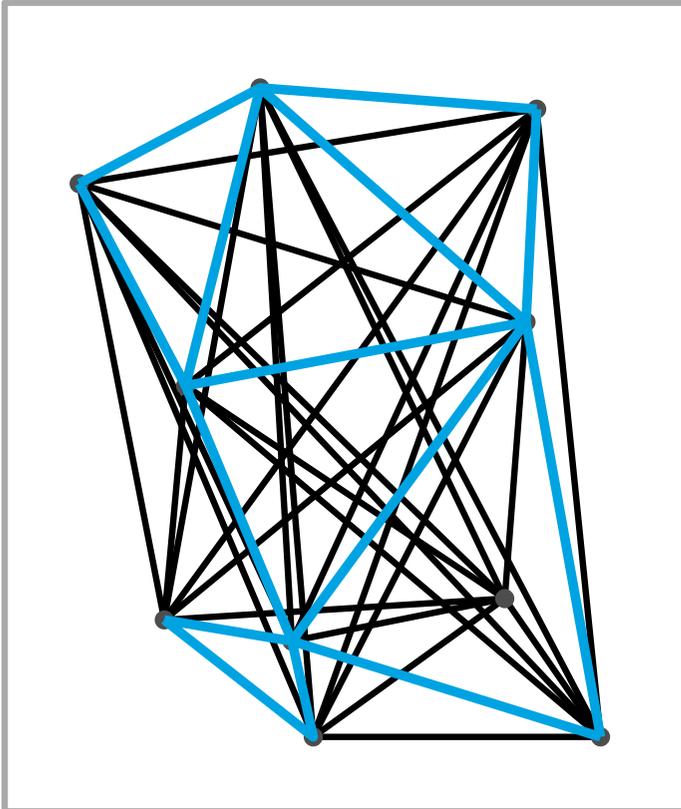
## Aufgabe:

Eine Menge von Städten soll über ein neues Straßennetz miteinander verbunden werden.

Allerdings soll für kein Paar  $(x, y)$  der Weg im Straßennetz viel länger als die Distanz  $\|xy\|$  sein.

Die Baukosten sollen im Rahmen bleiben, also z.B. nur  $O(n)$  Kanten.

1. Idee: Euklid. min. Spannbaum
2. Idee: vollständiger Graph



## Aufgabe:

Eine Menge von Städten soll über ein neues Straßennetz miteinander verbunden werden.

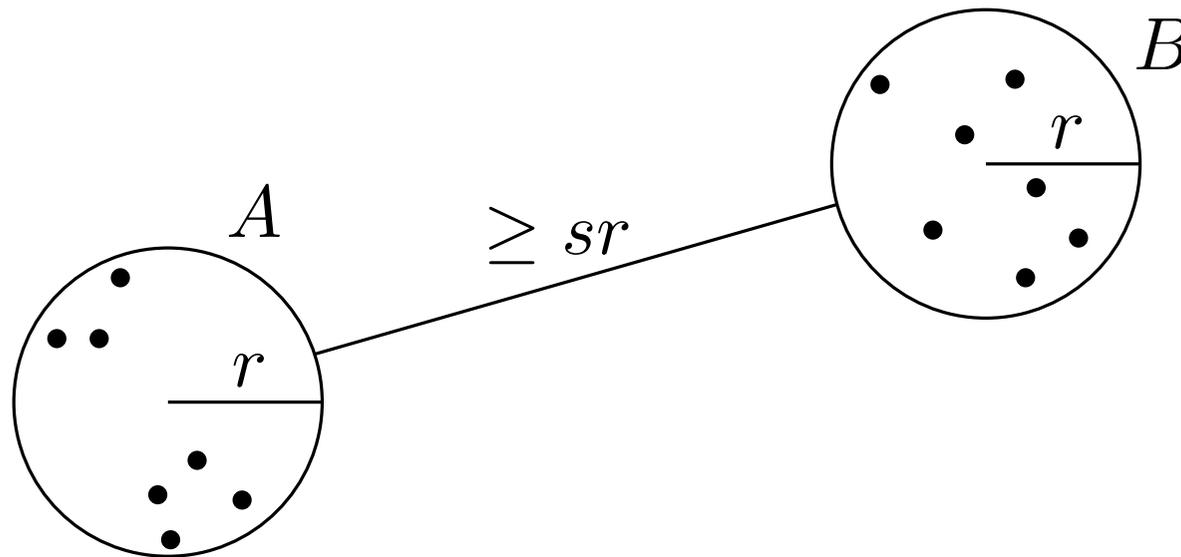
Allerdings soll für kein Paar  $(x, y)$  der Weg im Straßennetz viel länger als die Distanz  $\|xy\|$  sein.

Die Baukosten sollen im Rahmen bleiben, also z.B. nur  $O(n)$  Kanten.

1. Idee: Euklid. min. Spannbaum
2. Idee: vollständiger Graph
3. Idee: **sparse  $t$ -Spanner**

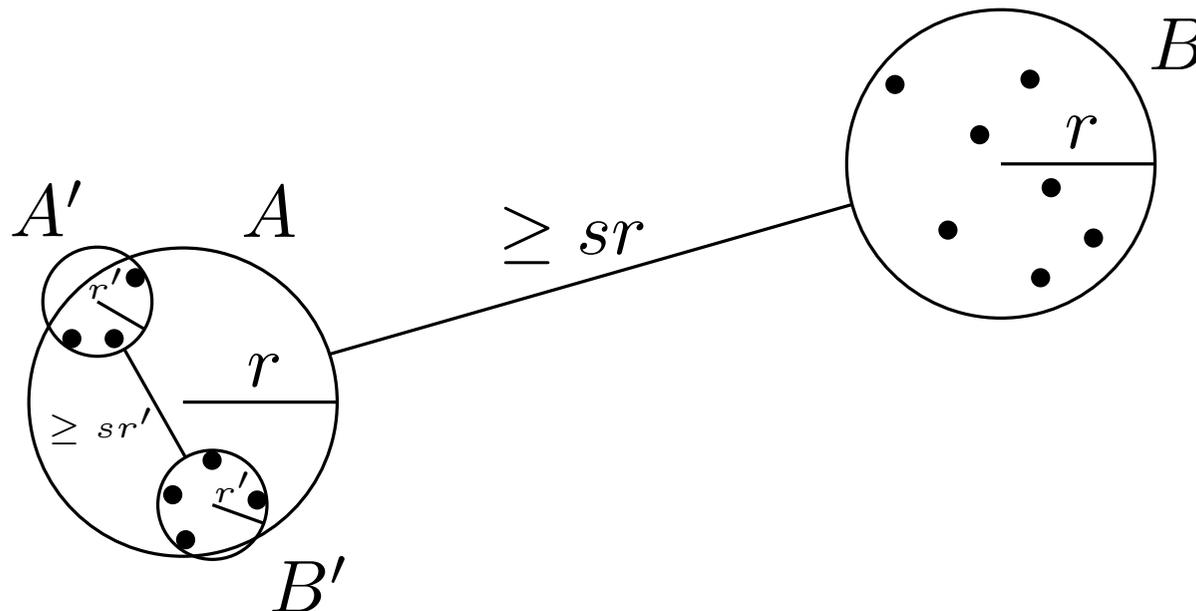
# Well-Separated Pairs

**Def.:** Ein Paar disjunkter Punktmenge  $A$  und  $B$  im  $\mathbb{R}^d$  heißt  **$s$ -well separated** für ein  $s > 0$ , falls  $A$  und  $B$  jeweils von einer Kugel mit Radius  $r$  überdeckt werden und der Abstand der beiden Kugeln mindestens  $sr$  ist.



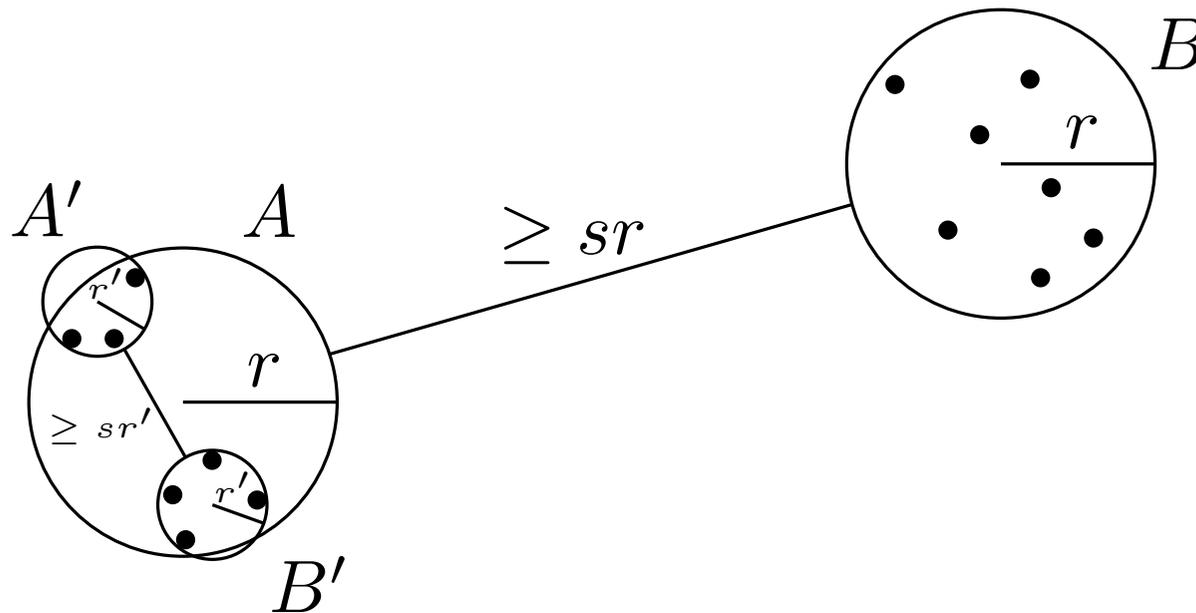
# Well-Separated Pairs

**Def.:** Ein Paar disjunkter Punktmenge  $A$  und  $B$  im  $\mathbb{R}^d$  heißt  **$s$ -well separated** für ein  $s > 0$ , falls  $A$  und  $B$  jeweils von einer Kugel mit Radius  $r$  überdeckt werden und der Abstand der beiden Kugeln mindestens  $sr$  ist.



# Well-Separated Pairs

**Def.:** Ein Paar disjunkter Punktmenge  $A$  und  $B$  im  $\mathbb{R}^d$  heißt  **$s$ -well separated** für ein  $s > 0$ , falls  $A$  und  $B$  jeweils von einer Kugel mit Radius  $r$  überdeckt werden und der Abstand der beiden Kugeln mindestens  $sr$  ist.



**Beob.:**

- $s$ -well separated  $\Rightarrow$   $s'$ -well separated für alle  $s' \leq s$
- Singletons  $\{a\}$  und  $\{b\}$  sind  $s$ -well separated für alle  $s > 0$

# Well-Separated Pair Decomposition (WSPD)



Für wohlsepariertes Paar  $\{A, B\}$  gilt, dass der Abstand für alle Punktpaare in  $A \otimes B = \{\{a, b\} \mid a \in A, b \in B, a \neq b\}$  ähnlich ist.

**Ziel:**  $o(n^2)$ -Datenstruktur, die den Abstand aller  $\binom{n}{2}$  Paare von Punkten einer Menge  $P = \{p_1, \dots, p_n\}$  approximiert.

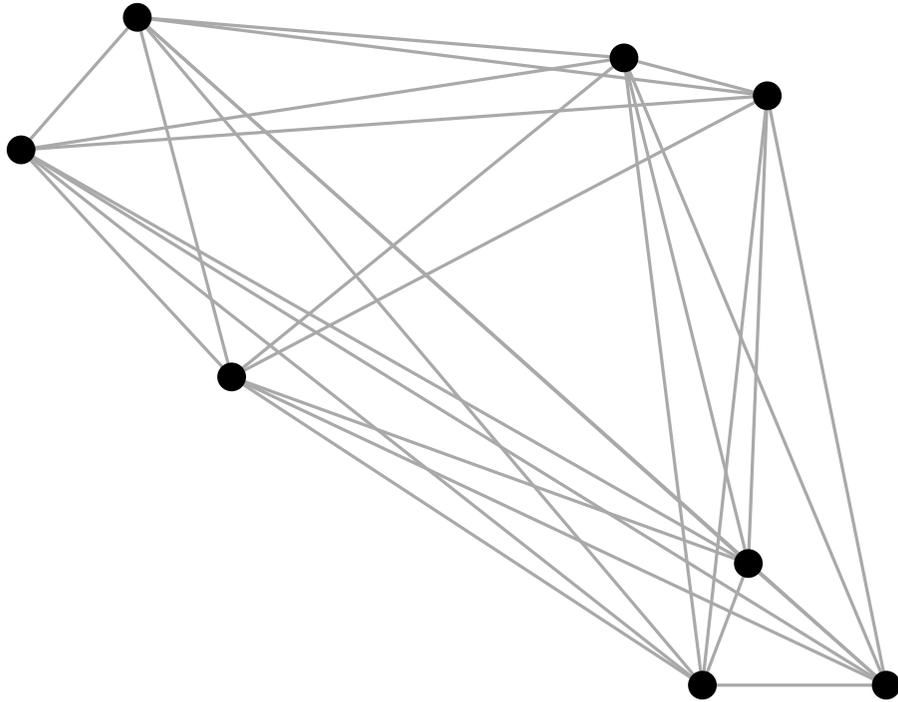
Für wohlsepariertes Paar  $\{A, B\}$  gilt, dass der Abstand für alle Punktpaare in  $A \otimes B = \{\{a, b\} \mid a \in A, b \in B, a \neq b\}$  ähnlich ist.

**Ziel:**  $o(n^2)$ -Datenstruktur, die den Abstand aller  $\binom{n}{2}$  Paare von Punkten einer Menge  $P = \{p_1, \dots, p_n\}$  approximiert.

**Def.:** Für eine Punktmenge  $P$  und ein  $s > 0$  ist eine  **$s$ -well separated pair decomposition** ( $s$ -WSPD) eine Menge von Paaren  $\{\{A_1, B_1\}, \dots, \{A_m, B_m\}\}$  mit

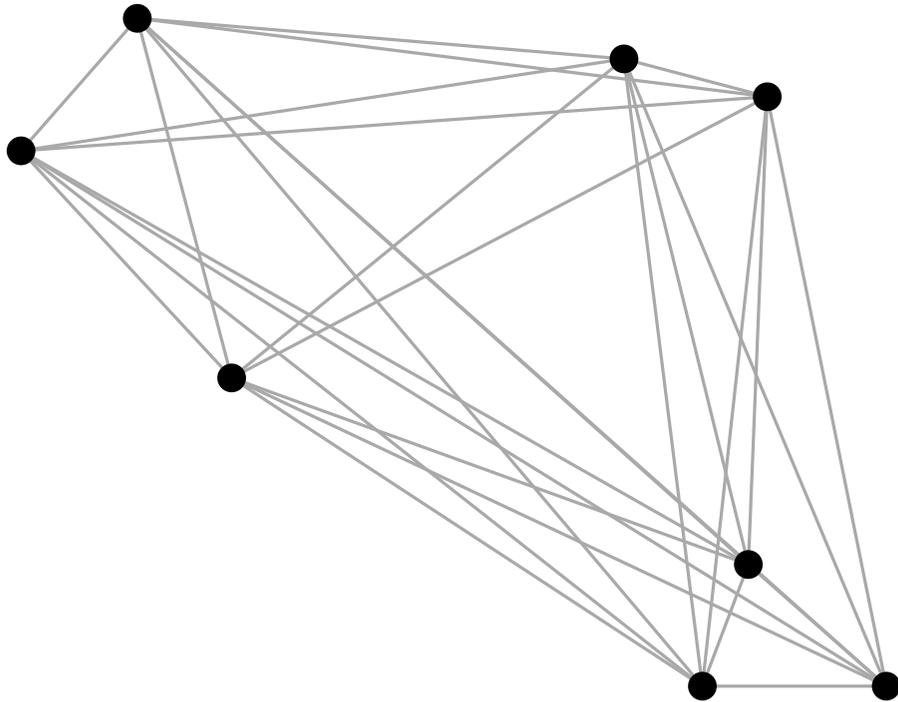
- $A_i, B_i \subset P$  für alle  $i$
- $A_i \cap B_i = \emptyset$  für alle  $i$
- $\bigcup_{i=1}^m A_i \otimes B_i = P \otimes P$
- $\{A_i, B_i\}$   $s$ -well separated für alle  $i$

# Beispiel

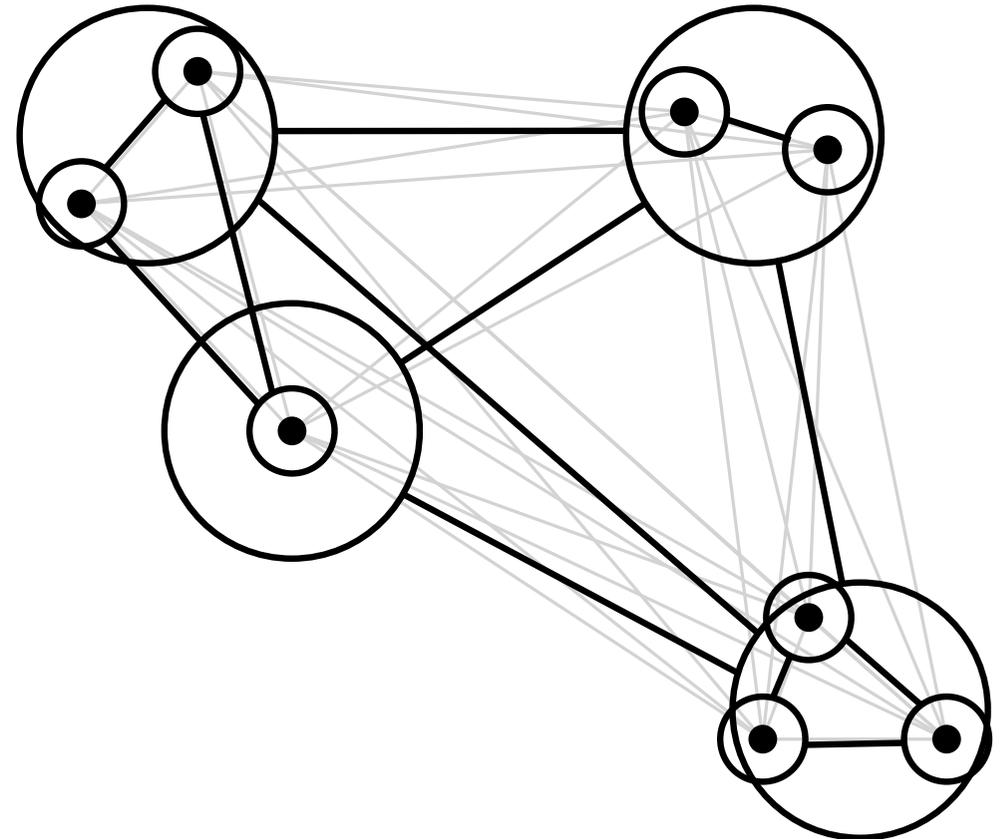


28 Punktpaare

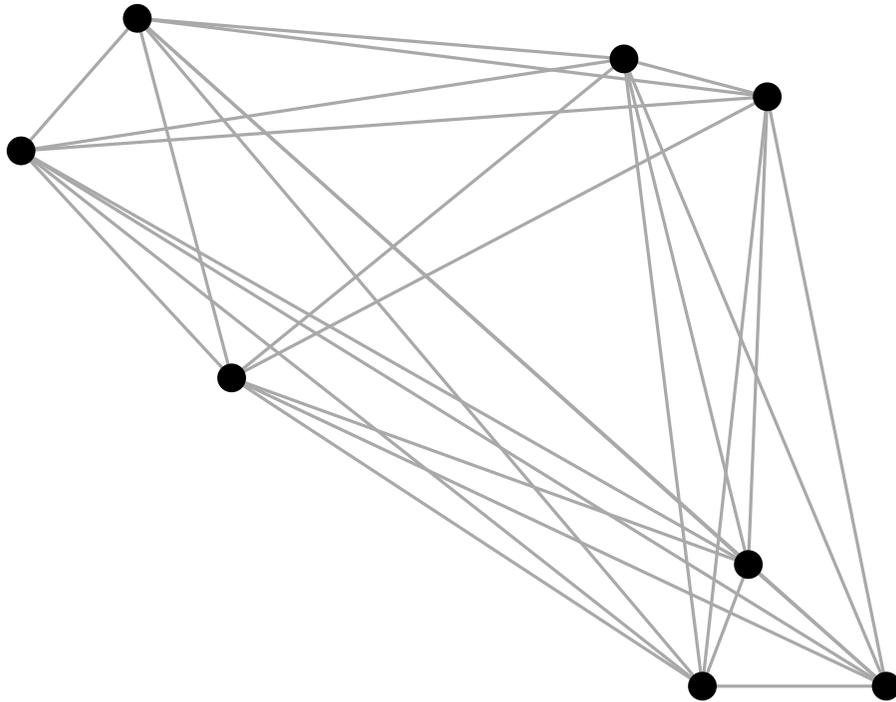
# Beispiel



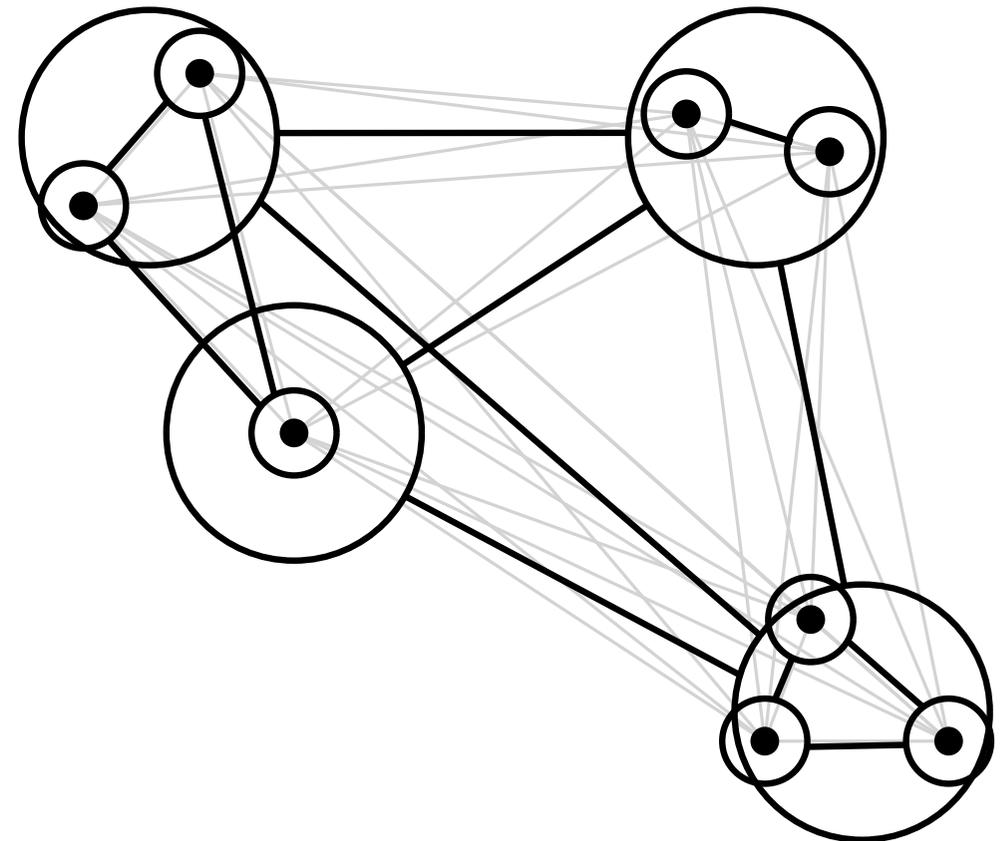
28 Punktpaare



12  $s$ -well separated pairs



28 Punktpaare

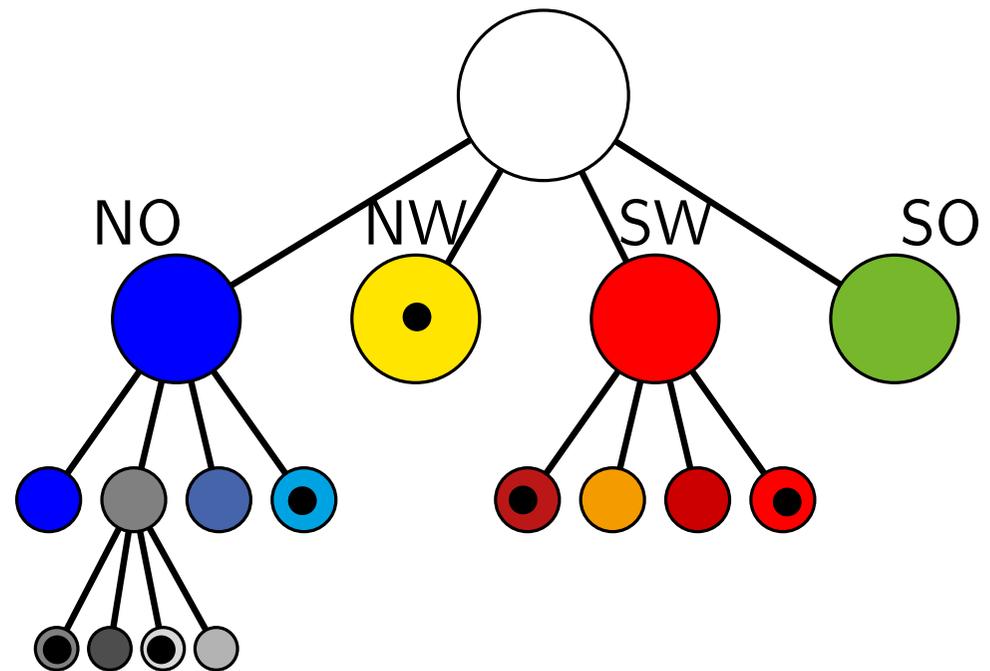
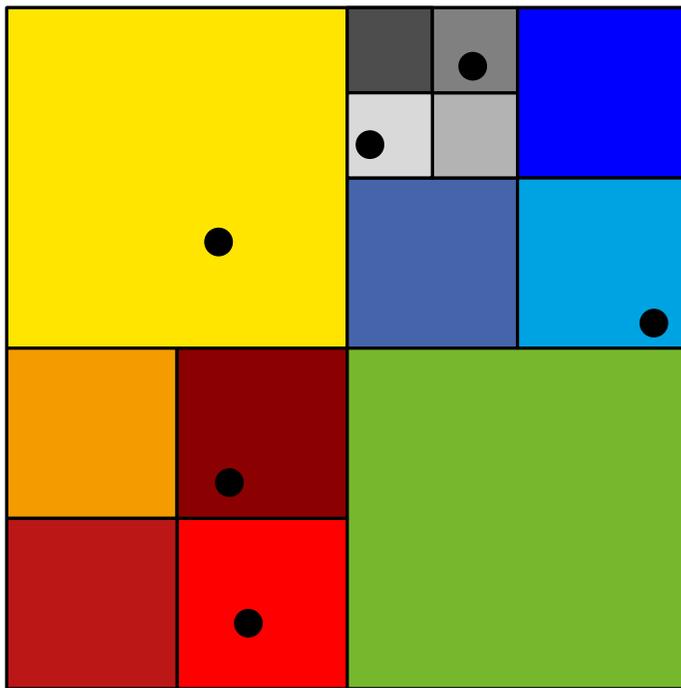


12  $s$ -well separated pairs

WSPD der Größe  $O(n^2)$  ist trivial. Geht es auch in  $O(n)$ ?

# Wiederholung: Quadrees

**Def.:** Ein **Quadtree**  $\mathcal{T}(P)$  für eine Punktmenge  $P$  ist ein Wurzelbaum, in dem jeder innere Knoten vier Kinder hat. Jeder Knoten entspricht einem Quadrat und die Quadrate der Blätter bilden eine Unterteilung des Wurzelquadrats.



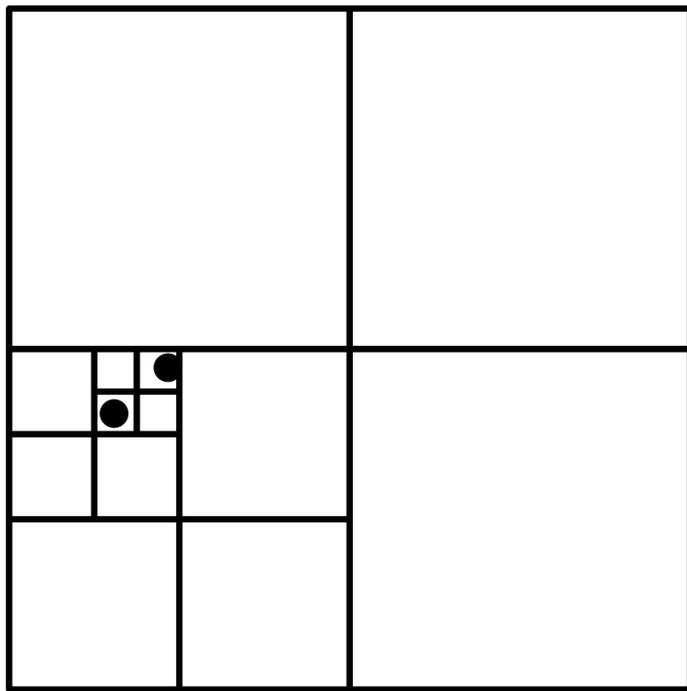
**Def.:** Ein **Quadtree**  $\mathcal{T}(P)$  für eine Punktmenge  $P$  ist ein Wurzelbaum, in dem jeder innere Knoten vier Kinder hat. Jeder Knoten entspricht einem Quadrat und die Quadrate der Blätter bilden eine Unterteilung des Wurzelquadrats.

**Lemma 1:** Die Tiefe  $d$  von  $\mathcal{T}(P)$  ist höchstens  $\log(s/c) + 3/2$ , wobei  $c$  der kleinste Abstand in  $P$  ist und  $s$  die Seitenlänge des Wurzelquadrats  $Q$ .

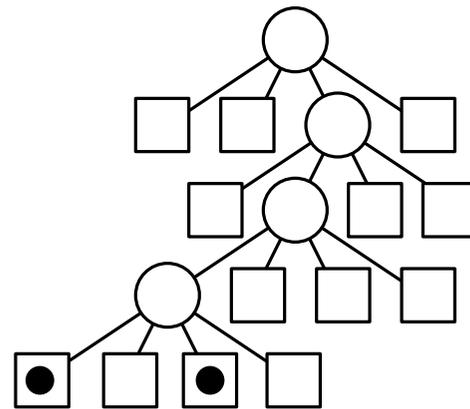
**Satz 1:** Ein Quadtree  $\mathcal{T}(P)$  für  $n$  Punkte und mit Tiefe  $d$  hat  $O((d+1)n)$  Knoten und kann in  $O((d+1)n)$  Zeit konstruiert werden.

# Komprimierte Quadrees

**Def.:** Ein **komprimierter** Quadtree ist ein Quadtree, bei dem Pfade nicht-separierender innerer Knoten zu einer Kante komprimiert sind. Die Kante trägt ein Label zur Rekonstruktion der Pfadstruktur.

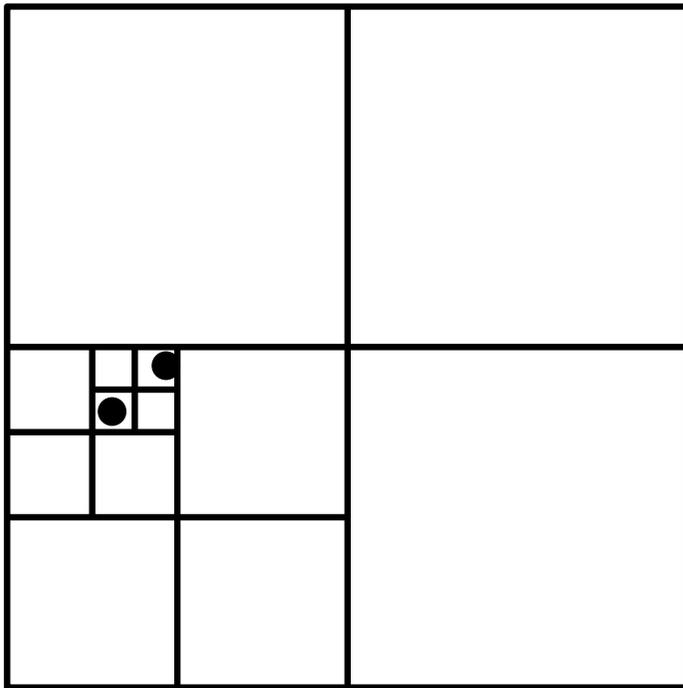


Quadtree

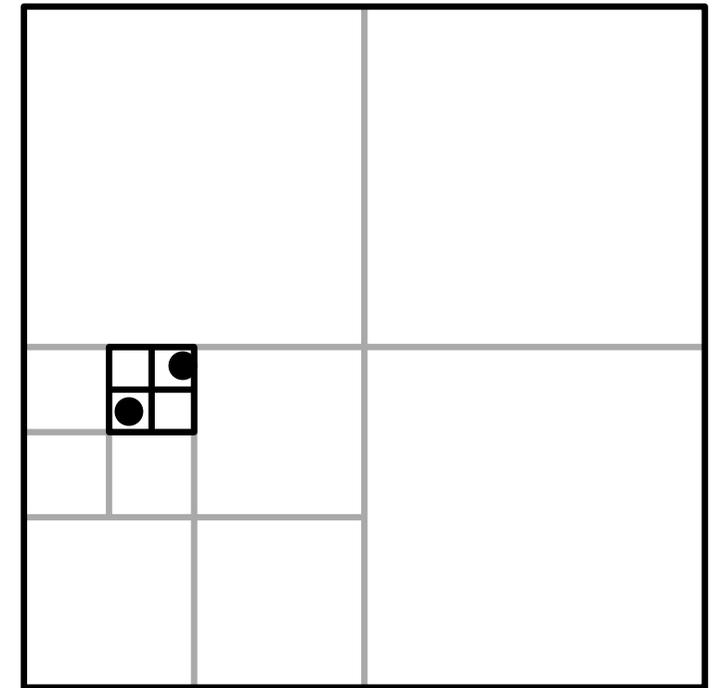
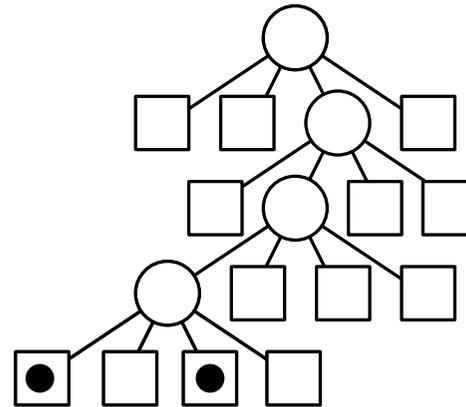


# Komprimierte Quadrees

**Def.:** Ein **komprimierter** Quadtree ist ein Quadtree, bei dem Pfade nicht-separierender innerer Knoten zu einer Kante komprimiert sind. Die Kante trägt ein Label zur Rekonstruktion der Pfadstruktur.



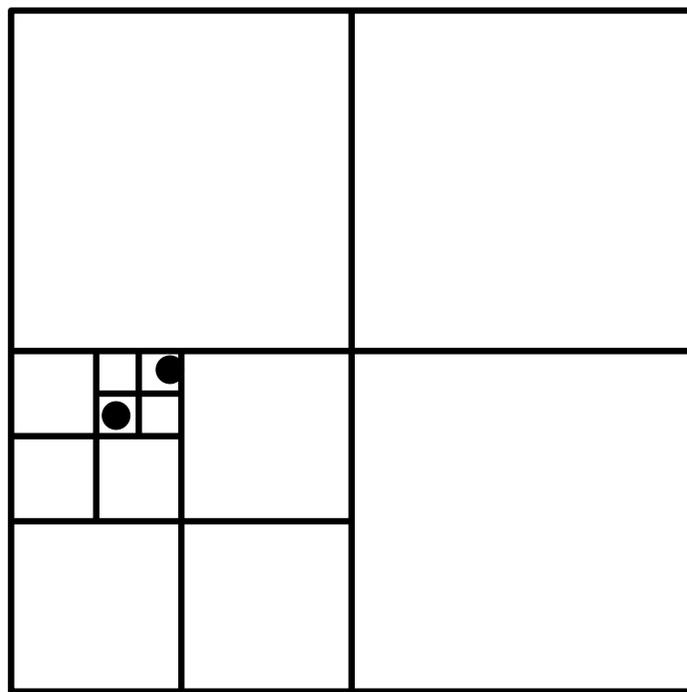
Quadtree



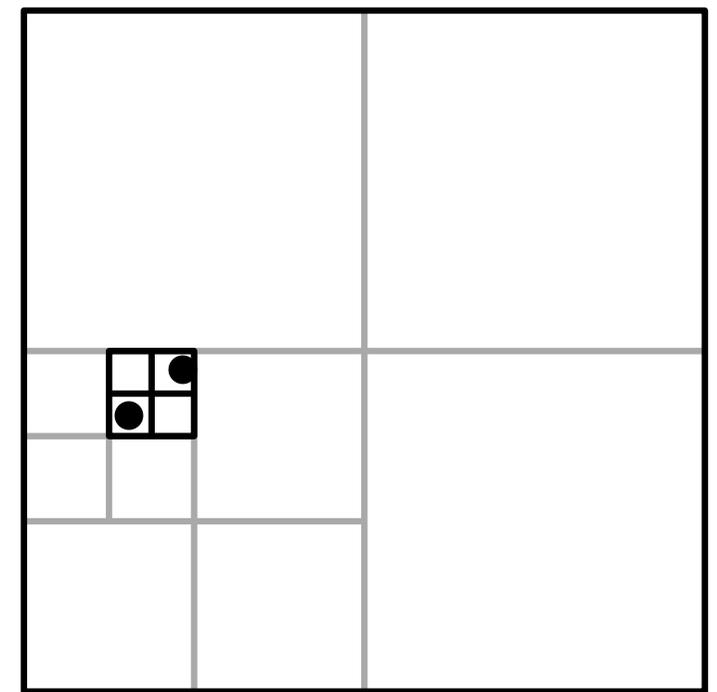
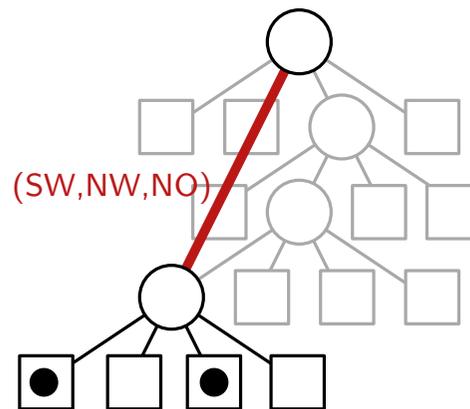
komprimierter Quadtree

# Komprimierte Quadrees

**Def.:** Ein **komprimierter** Quadtree ist ein Quadtree, bei dem Pfade nicht-separierender innerer Knoten zu einer Kante komprimiert sind. Die Kante trägt ein Label zur Rekonstruktion der Pfadstruktur.



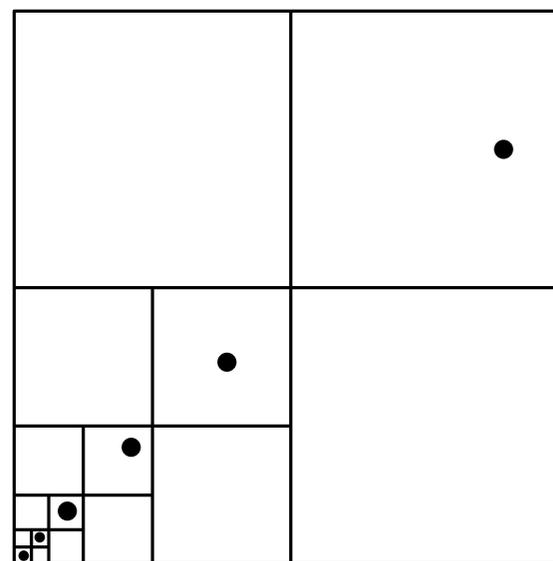
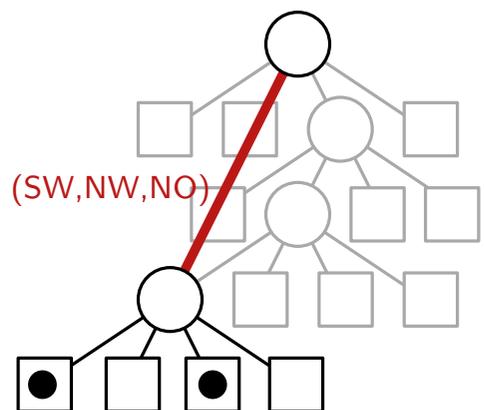
Quadtree



komprimierter Quadtree

# Eigenschaften komprimierter Quadrtrees

- Beob.:**
- innere Knoten teilen ihre Punktmenge in  $\geq 2$  nichtleere Teile  $\Rightarrow$  max.  $n - 1$  innere Knoten
  - Tiefe kann  $d = n$  sein, so dass der Algorithmus aus VL10  $O(n^2)$  Zeit zur Konstruktion braucht

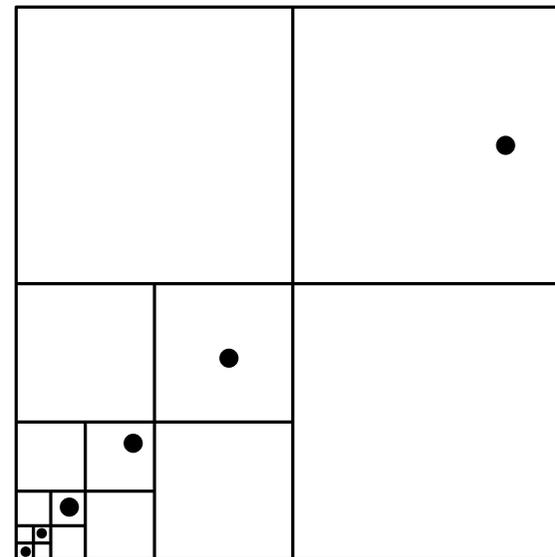
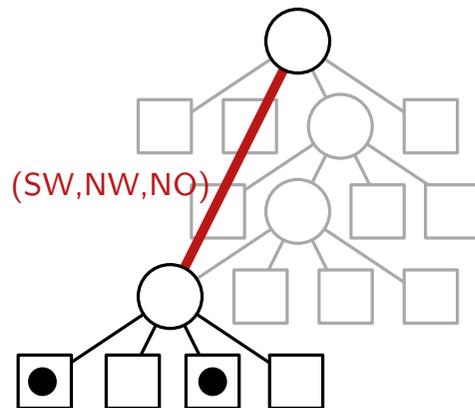


# Eigenschaften komprimierter Quadrees

- Beob.:**
- innere Knoten teilen ihre Punktmenge in  $\geq 2$  nichtleere Teile  $\Rightarrow$  max.  $n - 1$  innere Knoten
  - Tiefe kann  $d = n$  sein, so dass der Algorithmus aus VL10  $O(n^2)$  Zeit zur Konstruktion braucht

**Satz 2:** Ein komprimierter Quadtree für  $n$  Punkte im  $\mathbb{R}^d$  für festes  $d$  kann in  $O(n \log n)$  Zeit konstruiert werden.

(hier ohne Beweis)



**Lemma 2:** Gegeben sei eine Kugel  $K$  mit Radius  $r$  im  $\mathbb{R}^d$  und eine Menge  $X$  paarweise disjunkter Quadtree-Zellen mit Seitenlänge  $\geq x$ , die  $K$  schneiden. Dann gilt

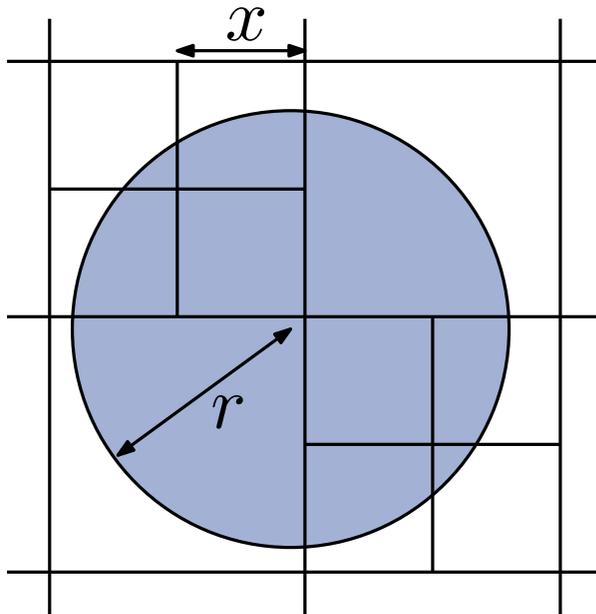
$$|X| \leq (1 + \lceil 2r/x \rceil)^d.$$

# Packungslemma

**Lemma 2:** Gegeben sei eine Kugel  $K$  mit Radius  $r$  im  $\mathbb{R}^d$  und eine Menge  $X$  paarweise disjunkter Quadtree-Zellen mit Seitenlänge  $\geq x$ , die  $K$  schneiden. Dann gilt

$$|X| \leq (1 + \lceil 2r/x \rceil)^d.$$

**Beweis:**

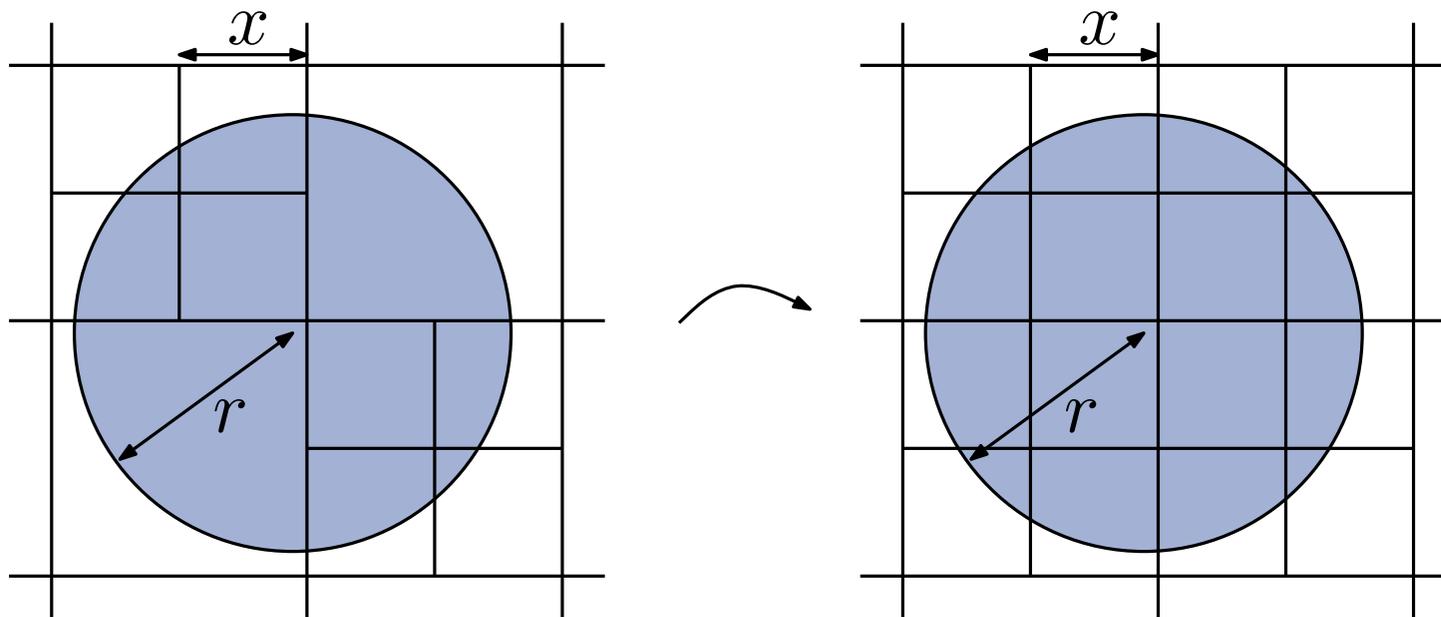


# Packungslemma

**Lemma 2:** Gegeben sei eine Kugel  $K$  mit Radius  $r$  im  $\mathbb{R}^d$  und eine Menge  $X$  paarweise disjunkter Quadtree-Zellen mit Seitenlänge  $\geq x$ , die  $K$  schneiden. Dann gilt

$$|X| \leq (1 + \lceil 2r/x \rceil)^d.$$

**Beweis:**

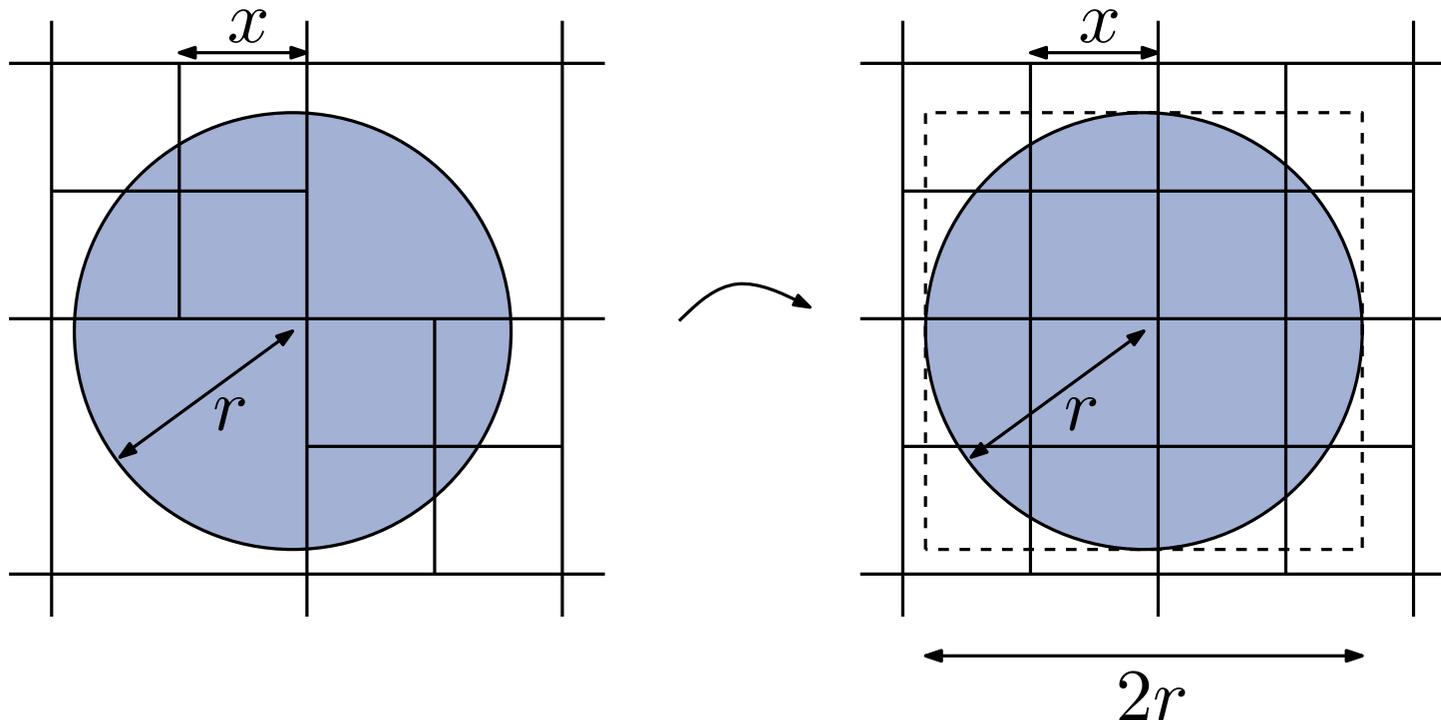


# Packungslemma

**Lemma 2:** Gegeben sei eine Kugel  $K$  mit Radius  $r$  im  $\mathbb{R}^d$  und eine Menge  $X$  paarweise disjunkter Quadtree-Zellen mit Seitenlänge  $\geq x$ , die  $K$  schneiden. Dann gilt

$$|X| \leq (1 + \lceil 2r/x \rceil)^d.$$

**Beweis:**



**Def.:** Für jeden Knoten  $u$  eines Quadrees  $\mathcal{T}(P)$  der Punktmenge  $P$  sei  $P_u = Q_u \cap P$  die Menge der Punkte im zugehörigen Quadrat  $Q_u$ . In jedem Blatt  $u$  definiere den Repräsentanten

$$\text{rep}(u) = \begin{cases} p & \text{falls } P_u = \{p\} \text{ (} u \text{ ist Blatt)} \\ \emptyset & \text{sonst.} \end{cases}$$

Für einen inneren Knoten  $v$  setze  $\text{rep}(v) = \text{rep}(u)$  für ein nichtleeres Kind  $u$  von  $v$ .

**Def.:** Für jeden Knoten  $u$  eines Quadtree  $\mathcal{T}(P)$  der Punktmenge  $P$  sei  $P_u = Q_u \cap P$  die Menge der Punkte im zugehörigen Quadrat  $Q_u$ . In jedem Blatt  $u$  definiere den Repräsentanten

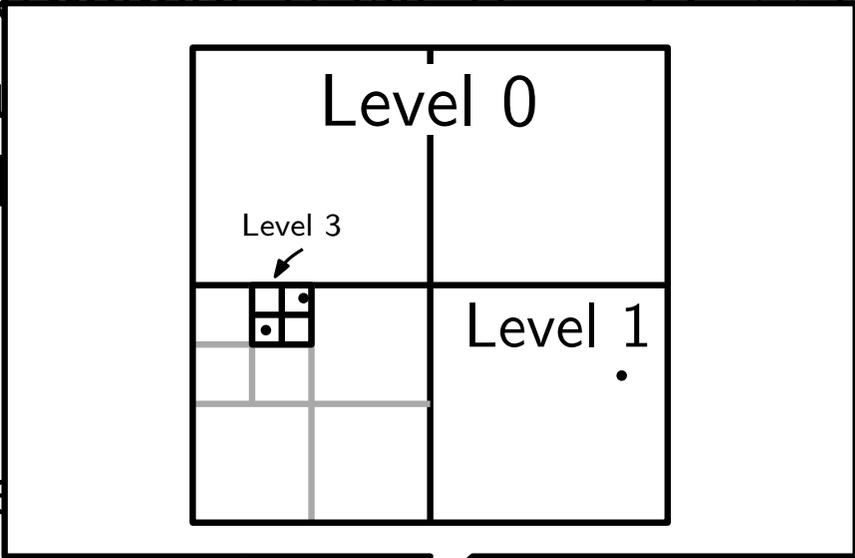
$$\text{rep}(u) = \begin{cases} p & \text{falls } P_u = \{p\} \text{ (} u \text{ ist Blatt)} \\ \emptyset & \text{sonst.} \end{cases}$$

Für einen inneren Knoten  $v$  setze  $\text{rep}(v) = \text{rep}(u)$  für ein nichtleeres Kind  $u$  von  $v$ .

**Def.:** Für jeden Knoten  $u$  eines Quadtree  $\mathcal{T}(P)$  sei  $\text{level}(u)$  das Level von  $u$  im zugeh. *unkomprimierten* Quadtree. Es gilt  $\text{level}(u) \leq \text{level}(v)$  gdw.  $\text{Fläche}(Q_u) \geq \text{Fläche}(Q_v)$ .

# Repräsentanten und Level

**Def.:** Für jeden Knoten  $u$  eines Quadtree  $\mathcal{T}(P)$  der Punktmenge  $P$  sei  $Q_u \subseteq P$  die Menge der Punkte im zugehörigen Blatt  $u$  definiere den Repräsentanten  $rep(u)$  als



$\{p\}$  ( $u$  ist Blatt)

Für ein nichtleeres Kind  $v$  von  $u$  sei  $rep(v) = rep(u)$  für ein

**Def.:** Für jeden Knoten  $u$  eines Quadtree  $\mathcal{T}(P)$  sei  $level(u)$  das Level von  $u$  im zugehörigen *unkomprimierten* Quadtree. Es gilt  $level(u) \leq level(v)$  gdw.  $Fläche(Q_u) \geq Fläche(Q_v)$ .

# Konstruktion einer WSPD

$\text{wsPairs}(u, v, \mathcal{T}, s)$

**Input:** Quadtree-Knoten  $u, v$ , Quadtree  $\mathcal{T}$ ,  $s > 0$

**Output:** WSPD für  $P_u \otimes P_v$

**if**  $\text{rep}(u) = \emptyset$  oder  $\text{rep}(v) = \emptyset$  oder Blätter  $u = v$  **then return**  $\emptyset$

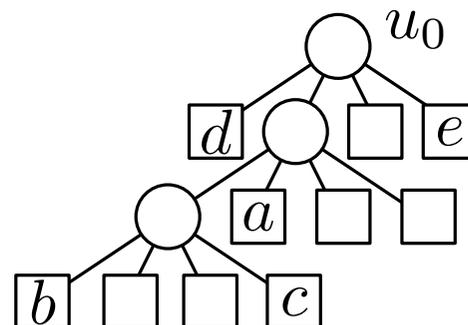
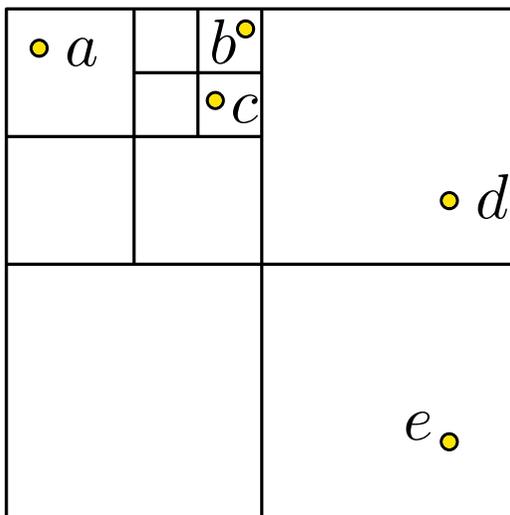
**else if**  $P_u$  und  $P_v$   $s$ -well separated **then return**  $\{\{u, v\}\}$

**else**

**if**  $\text{level}(u) > \text{level}(v)$  **then** tausche  $u$  und  $v$

$(u_1, \dots, u_m) \leftarrow$  Kinder von  $u$  in  $\mathcal{T}$

**return**  $\bigcup_{i=1}^m \text{wsPairs}(u_i, v, \mathcal{T}, s)$



# Konstruktion einer WSPD

$\text{wsPairs}(u, v, \mathcal{T}, s)$

**Input:** Quadtree-Knoten  $u, v$ , Quadtree  $\mathcal{T}$ ,  $s > 0$

**Output:** WSPD für  $P_u \otimes P_v$

**if**  $\text{rep}(u) = \emptyset$  oder  $\text{rep}(v) = \emptyset$  oder Blätter  $u = v$  **then return**  $\emptyset$

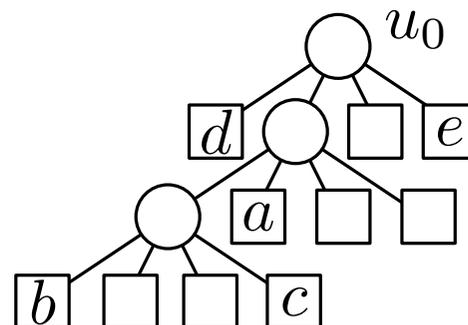
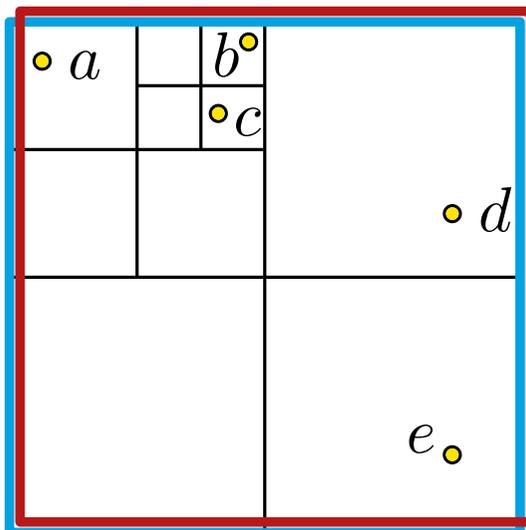
**else if**  $P_u$  und  $P_v$   $s$ -well separated **then return**  $\{\{u, v\}\}$

**else**

**if**  $\text{level}(u) > \text{level}(v)$  **then** tausche  $u$  und  $v$

$(u_1, \dots, u_m) \leftarrow$  Kinder von  $u$  in  $\mathcal{T}$

**return**  $\bigcup_{i=1}^m \text{wsPairs}(u_i, v, \mathcal{T}, s)$



# Konstruktion einer WSPD

$\text{wsPairs}(u, v, \mathcal{T}, s)$

**Input:** Quadtree-Knoten  $u, v$ , Quadtree  $\mathcal{T}$ ,  $s > 0$

**Output:** WSPD für  $P_u \otimes P_v$

**if**  $\text{rep}(u) = \emptyset$  oder  $\text{rep}(v) = \emptyset$  oder Blätter  $u = v$  **then return**  $\emptyset$

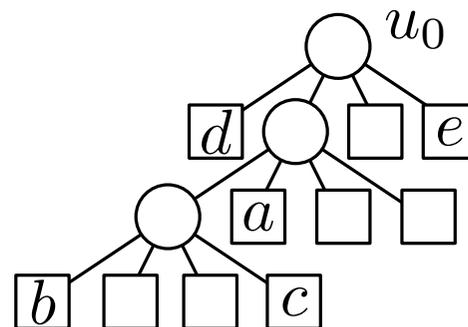
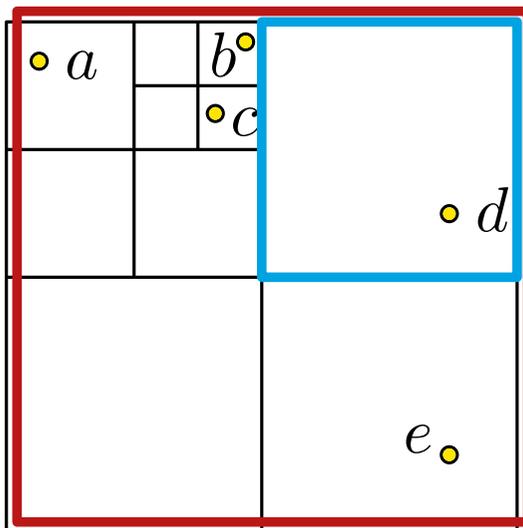
**else if**  $P_u$  und  $P_v$   $s$ -well separated **then return**  $\{\{u, v\}\}$

**else**

**if**  $\text{level}(u) > \text{level}(v)$  **then** tausche  $u$  und  $v$

$(u_1, \dots, u_m) \leftarrow$  Kinder von  $u$  in  $\mathcal{T}$

**return**  $\bigcup_{i=1}^m \text{wsPairs}(u_i, v, \mathcal{T}, s)$



# Konstruktion einer WSPD

$\text{wsPairs}(u, v, \mathcal{T}, s)$

**Input:** Quadtree-Knoten  $u, v$ , Quadtree  $\mathcal{T}$ ,  $s > 0$

**Output:** WSPD für  $P_u \otimes P_v$

**if**  $\text{rep}(u) = \emptyset$  oder  $\text{rep}(v) = \emptyset$  oder Blätter  $u = v$  **then return**  $\emptyset$

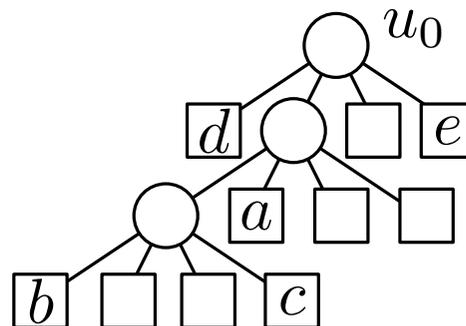
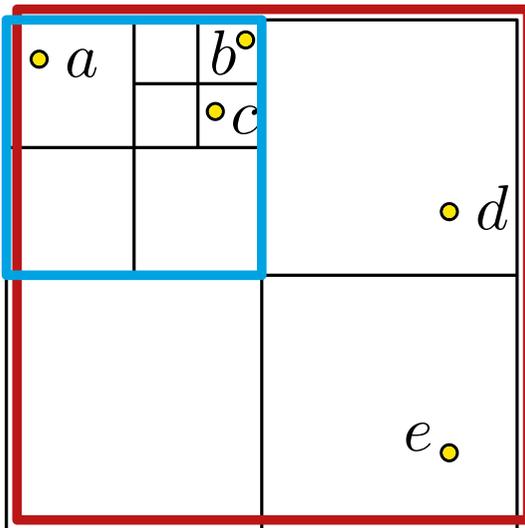
**else if**  $P_u$  und  $P_v$   $s$ -well separated **then return**  $\{\{u, v\}\}$

**else**

**if**  $\text{level}(u) > \text{level}(v)$  **then** tausche  $u$  und  $v$

$(u_1, \dots, u_m) \leftarrow$  Kinder von  $u$  in  $\mathcal{T}$

**return**  $\bigcup_{i=1}^m \text{wsPairs}(u_i, v, \mathcal{T}, s)$



# Konstruktion einer WSPD

$\text{wsPairs}(u, v, \mathcal{T}, s)$

**Input:** Quadtree-Knoten  $u, v$ , Quadtree  $\mathcal{T}$ ,  $s > 0$

**Output:** WSPD für  $P_u \otimes P_v$

**if**  $\text{rep}(u) = \emptyset$  oder  $\text{rep}(v) = \emptyset$  oder Blätter  $u = v$  **then return**  $\emptyset$

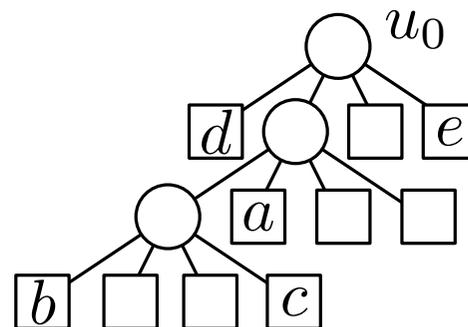
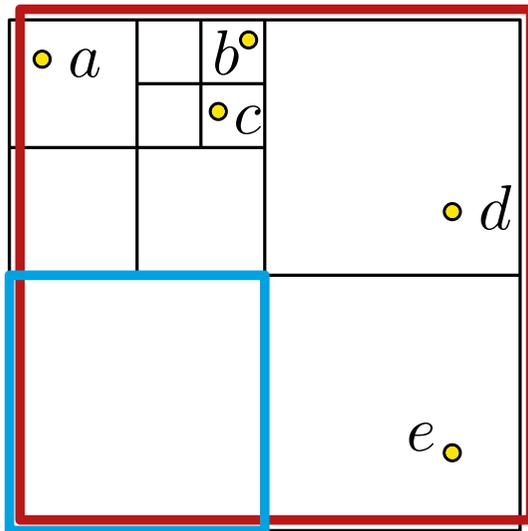
**else if**  $P_u$  und  $P_v$   $s$ -well separated **then return**  $\{\{u, v\}\}$

**else**

**if**  $\text{level}(u) > \text{level}(v)$  **then** tausche  $u$  und  $v$

$(u_1, \dots, u_m) \leftarrow$  Kinder von  $u$  in  $\mathcal{T}$

**return**  $\bigcup_{i=1}^m \text{wsPairs}(u_i, v, \mathcal{T}, s)$



# Konstruktion einer WSPD

$\text{wsPairs}(u, v, \mathcal{T}, s)$

**Input:** Quadtree-Knoten  $u, v$ , Quadtree  $\mathcal{T}$ ,  $s > 0$

**Output:** WSPD für  $P_u \otimes P_v$

**if**  $\text{rep}(u) = \emptyset$  oder  $\text{rep}(v) = \emptyset$  oder Blätter  $u = v$  **then return**  $\emptyset$

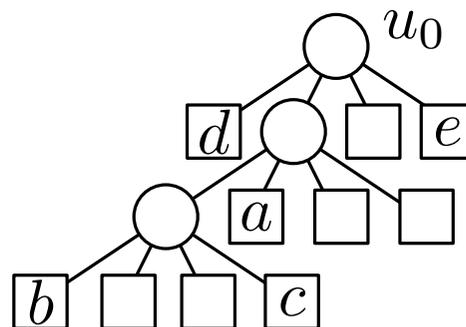
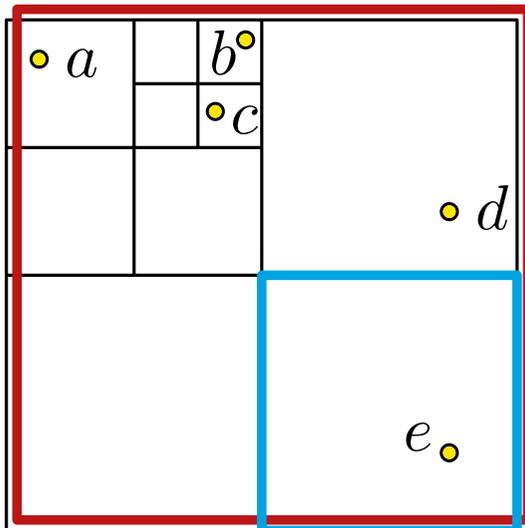
**else if**  $P_u$  und  $P_v$   $s$ -well separated **then return**  $\{\{u, v\}\}$

**else**

**if**  $\text{level}(u) > \text{level}(v)$  **then** tausche  $u$  und  $v$

$(u_1, \dots, u_m) \leftarrow$  Kinder von  $u$  in  $\mathcal{T}$

**return**  $\bigcup_{i=1}^m \text{wsPairs}(u_i, v, \mathcal{T}, s)$



# Konstruktion einer WSPD

$\text{wsPairs}(u, v, \mathcal{T}, s)$

**Input:** Quadtree-Knoten  $u, v$ , Quadtree  $\mathcal{T}$ ,  $s > 0$

**Output:** WSPD für  $P_u \otimes P_v$

**if**  $\text{rep}(u) = \emptyset$  oder  $\text{rep}(v) = \emptyset$  oder Blätter  $u = v$  **then return**  $\emptyset$

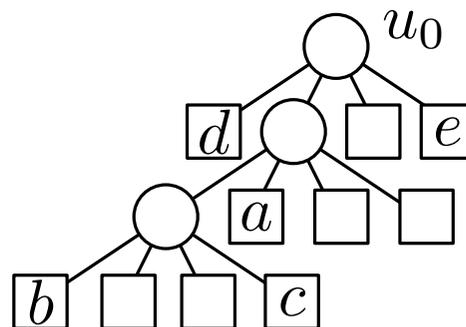
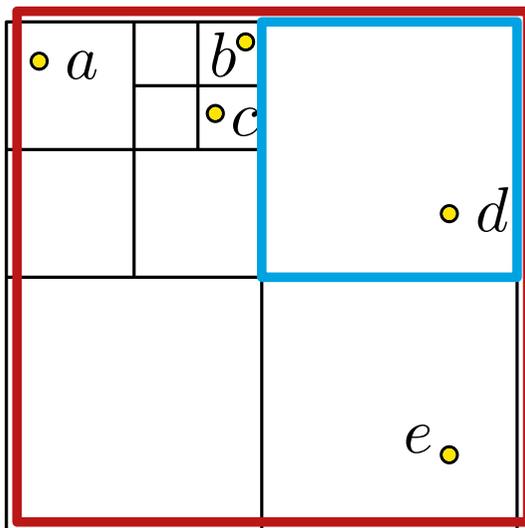
**else if**  $P_u$  und  $P_v$   $s$ -well separated **then return**  $\{\{u, v\}\}$

**else**

**if**  $\text{level}(u) > \text{level}(v)$  **then** tausche  $u$  und  $v$

$(u_1, \dots, u_m) \leftarrow$  Kinder von  $u$  in  $\mathcal{T}$

**return**  $\bigcup_{i=1}^m \text{wsPairs}(u_i, v, \mathcal{T}, s)$



# Konstruktion einer WSPD

$\text{wsPairs}(u, v, \mathcal{T}, s)$

**Input:** Quadtree-Knoten  $u, v$ , Quadtree  $\mathcal{T}$ ,  $s > 0$

**Output:** WSPD für  $P_u \otimes P_v$

**if**  $\text{rep}(u) = \emptyset$  oder  $\text{rep}(v) = \emptyset$  oder Blätter  $u = v$  **then return**  $\emptyset$

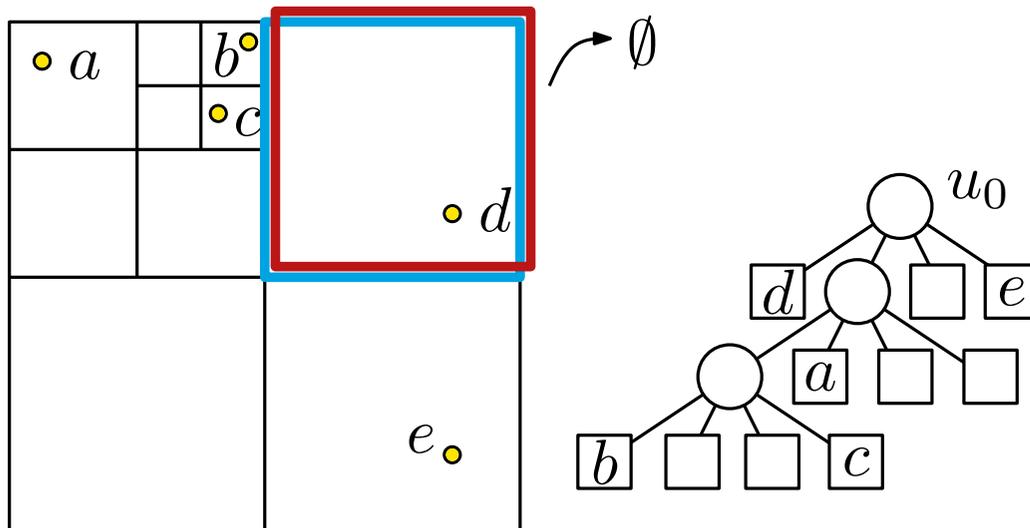
**else if**  $P_u$  und  $P_v$   $s$ -well separated **then return**  $\{\{u, v\}\}$

**else**

**if**  $\text{level}(u) > \text{level}(v)$  **then** tausche  $u$  und  $v$

$(u_1, \dots, u_m) \leftarrow$  Kinder von  $u$  in  $\mathcal{T}$

**return**  $\bigcup_{i=1}^m \text{wsPairs}(u_i, v, \mathcal{T}, s)$



# Konstruktion einer WSPD

$\text{wsPairs}(u, v, \mathcal{T}, s)$

**Input:** Quadrate  $Q_u$  und  $Q_v$  (bzw. Radius 0 für Punkt in Blatt)

**Output:** WS-Quadrate  $Q_u$  und  $Q_v$  (bzw. Radius 0 für Punkt in Blatt) vergrößere kleineren Kreis und prüfe Abstand  $\geq sr$

**if**  $\text{rep}(u) = \emptyset$  oder  $\text{rep}(v) = \emptyset$  oder Blätter  $u = v$  **then return**  $\emptyset$

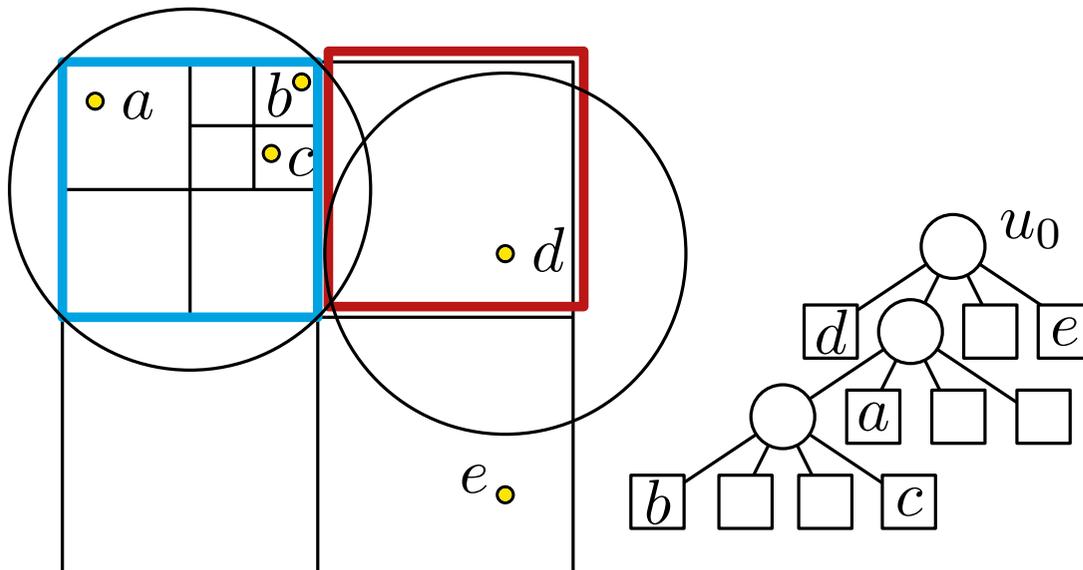
**else if**  $P_u$  und  $P_v$   $s$ -well separated **then return**  $\{\{u, v\}\}$

**else**

**if**  $\text{level}(u) > \text{level}(v)$  **then** tausche  $u$  und  $v$

$(u_1, \dots, u_m) \leftarrow$  Kinder von  $u$  in  $\mathcal{T}$

**return**  $\bigcup_{i=1}^m \text{wsPairs}(u_i, v, \mathcal{T}, s)$



# Konstruktion einer WSPD

$wsPairs(u, v, \mathcal{T}, s)$

**Input:** Quadrate  $Q_u$  und  $Q_v$  (bzw. Radius 0 für Punkt in Blatt)

**Output:** WS  $\{u, v\}$  vergrößere kleineren Kreis und prüfe Abstand  $\geq sr$

**if**  $rep(u) = \emptyset$  oder  $rep(v) = \emptyset$  oder Blätter  $u = v$  **then return**  $\emptyset$

**else if**  $P_u$  und  $P_v$   $s$ -well separated **then return**  $\{\{u, v\}\}$

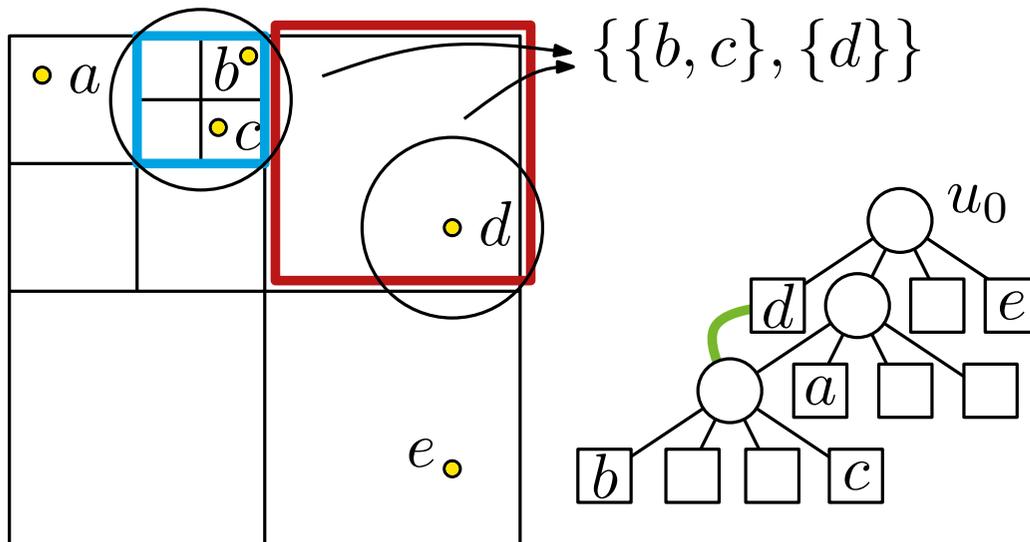
**else**

**if**  $level(u) > level(v)$  **then** tausche  $u$  und  $v$

$(u_1, \dots, u_m) \leftarrow$  Kinder von  $u$  in  $\mathcal{T}$

**return**  $\bigcup_{i=1}^m wsPairs(u_i, v, \mathcal{T}, s)$

$\{\{b, c\}, \{d\}\}$



# Konstruktion einer WSPD

$\text{wsPairs}(u, v, \mathcal{T}, s)$

**Input:** Quadtree-Knoten  $u, v$ , Quadtree  $\mathcal{T}$ ,  $s > 0$

**Output:** WSPD für  $P_u \otimes P_v$

**if**  $\text{rep}(u) = \emptyset$  oder  $\text{rep}(v) = \emptyset$  oder Blätter  $u = v$  **then return**  $\emptyset$

**else if**  $P_u$  und  $P_v$   $s$ -well separated **then return**  $\{\{u, v\}\}$

**else**

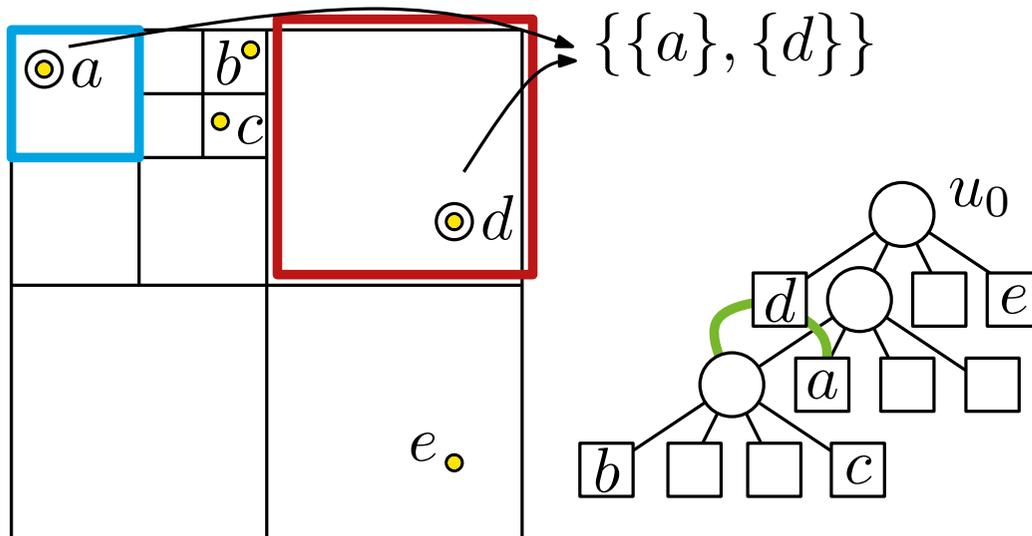
**if**  $\text{level}(u) > \text{level}(v)$  **then** tausche  $u$  und  $v$

$(u_1, \dots, u_m) \leftarrow$  Kinder von  $u$  in  $\mathcal{T}$

**return**  $\bigcup_{i=1}^m \text{wsPairs}(u_i, v, \mathcal{T}, s)$

$\{\{b, c\}, \{d\}\}$

$\{\{a\}, \{d\}\}$



# Konstruktion einer WSPD

$\text{wsPairs}(u, v, \mathcal{T}, s)$

**Input:** Quadtree-Knoten  $u, v$ , Quadtree  $\mathcal{T}$ ,  $s > 0$

**Output:** WSPD für  $P_u \otimes P_v$

**if**  $\text{rep}(u) = \emptyset$  oder  $\text{rep}(v) = \emptyset$  oder Blätter  $u = v$  **then return**  $\emptyset$

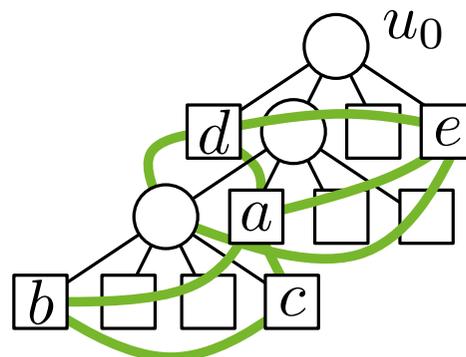
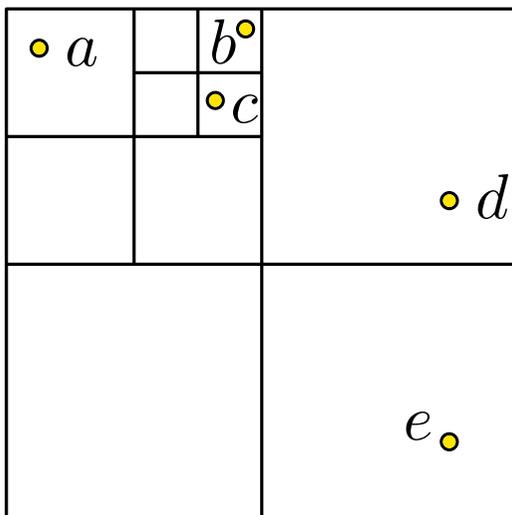
**else if**  $P_u$  und  $P_v$   $s$ -well separated **then return**  $\{\{u, v\}\}$

**else**

**if**  $\text{level}(u) > \text{level}(v)$  **then** tausche  $u$  und  $v$

$(u_1, \dots, u_m) \leftarrow$  Kinder von  $u$  in  $\mathcal{T}$

**return**  $\bigcup_{i=1}^m \text{wsPairs}(u_i, v, \mathcal{T}, s)$



- $\{\{b, c\}, \{d\}\}$
- $\{\{a\}, \{d\}\}$
- $\{\{b, c\}, \{e\}\}$
- $\{\{d\}, \{e\}\}$
- $\{\{a\}, \{b\}\}$
- $\{\{a\}, \{c\}\}$
- $\{\{b\}, \{c\}\}$
- $\{\{a\}, \{e\}\}$

# Konstruktion einer WSPD

$\text{wsPairs}(u, v, \mathcal{T}, s)$

**Input:** Quadtree-Knoten  $u, v$ , Quadtree  $\mathcal{T}$ ,  $s > 0$

**Output:** WSPD für  $P_u \otimes P_v$

**if**  $\text{rep}(u) = \emptyset$  oder  $\text{rep}(v) = \emptyset$  oder Blätter  $u = v$  **then return**  $\emptyset$

**else if**  $P_u$  und  $P_v$   $s$ -well separated **then return**  $\{\{u, v\}\}$

**else**

**if**  $\text{level}(u) > \text{level}(v)$  **then** tausche  $u$  und  $v$

$(u_1, \dots, u_m) \leftarrow$  Kinder von  $u$  in  $\mathcal{T}$

**return**  $\bigcup_{i=1}^m \text{wsPairs}(u_i, v, \mathcal{T}, s)$

- initialer Aufruf  $\text{wsPairs}(u_0, u_0, \mathcal{T}, s)$  Wie?
- Duplikate  $\text{wsPairs}(u_i, u_j, \mathcal{T}, s)$  und  $\text{wsPairs}(u_j, u_i, \mathcal{T}, s)$  vermeiden
- Blattpaare sind immer  $s$ -well separated, Algorithmus terminiert also
- Ausgabe sind Paare von Quadtree-Knoten Speicherplatz?

# Konstruktion einer WSPD

$\text{wsPairs}(u, v, \mathcal{T}, s)$

**Input:** Quadtree-Knoten  $u, v$ , Quadtree  $\mathcal{T}$ ,  $s > 0$

**Output:** WSPD für  $P_u \otimes P_v$

**if**  $\text{rep}(u) = \emptyset$  oder  $\text{rep}(v) = \emptyset$  oder Blätter  $u = v$  **then return**  $\emptyset$

**else if**  $P_u$  und  $P_v$   $s$ -well separated **then return**  $\{\{u, v\}\}$

**else**

**if**  $\text{level}(u) > \text{level}(v)$  **then** tausche  $u$  und  $v$

$(u_1, \dots, u_m) \leftarrow$  Kinder von  $u$  in  $\mathcal{T}$

**return**  $\bigcup_{i=1}^m \text{wsPairs}(u_i, v, \mathcal{T}, s)$

- initialer Aufruf  $\text{wsPairs}(u_0, u_0, \mathcal{T}, s)$  Wie?
- Duplikate  $\text{wsPairs}(u_i, u_j, \mathcal{T}, s)$  und  $\text{wsPairs}(u_j, u_i, \mathcal{T}, s)$  vermeiden
- Blattpaare sind immer  $s$ -well separated, Algorithmus terminiert also
- Ausgabe sind Paare von Quadtree-Knoten Speicherplatz?

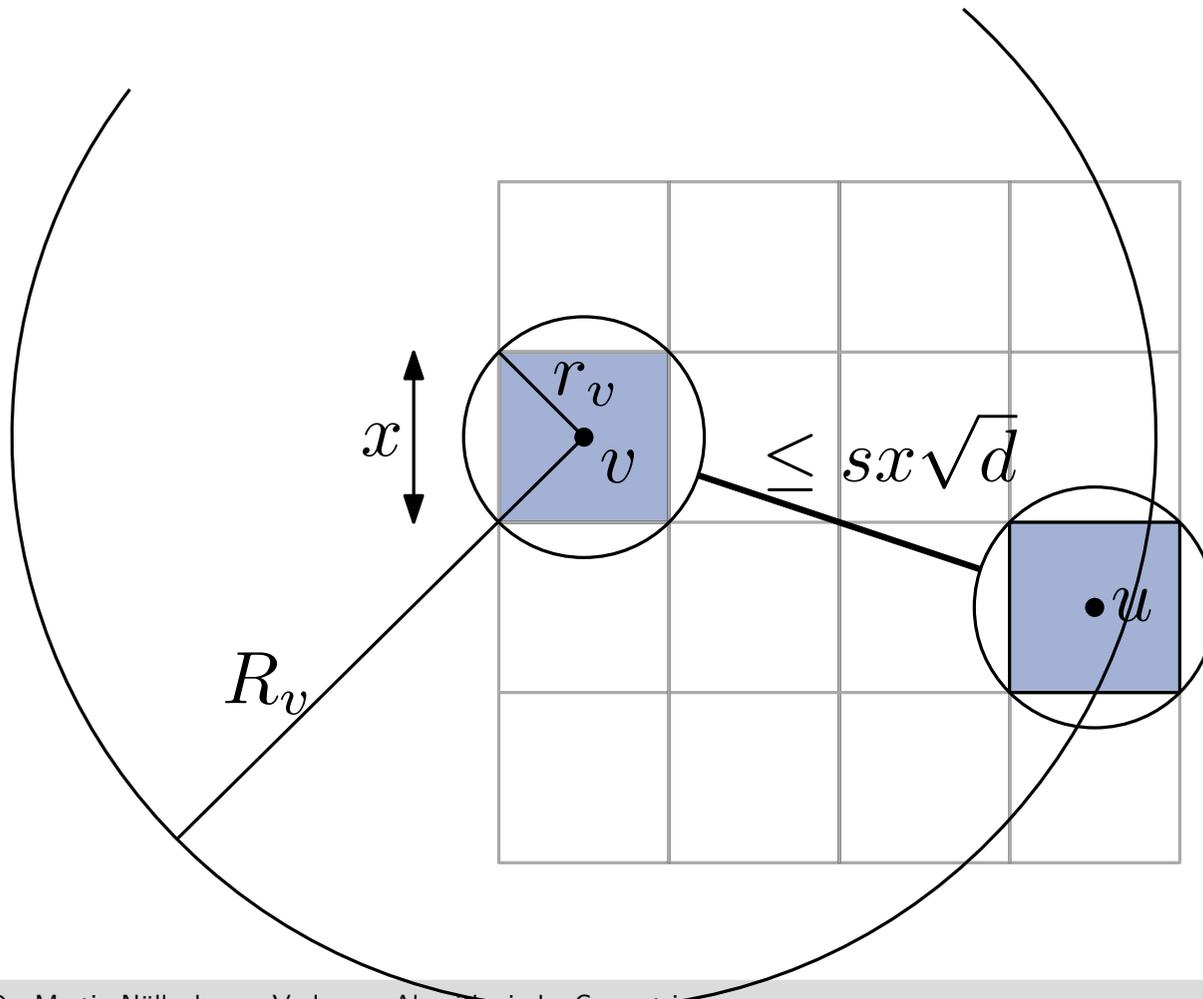
**Frage:** Wie viele Paare erzeugt der Algorithmus?

**Satz 3:** Gegeben eine Punktmenge  $P$  im  $\mathbb{R}^d$  und  $s \geq 1$  so lässt sich eine  $s$ -WSPD mit  $O(s^d n)$  Paaren in Zeit  $O(n \log n + s^d n)$  konstruieren.

# Analyse WSPD-Konstruktion

**Satz 3:** Gegeben eine Punktmenge  $P$  im  $\mathbb{R}^d$  und  $s \geq 1$  so lässt sich eine  $s$ -WSPD mit  $O(s^d n)$  Paaren in Zeit  $O(n \log n + s^d n)$  konstruieren.

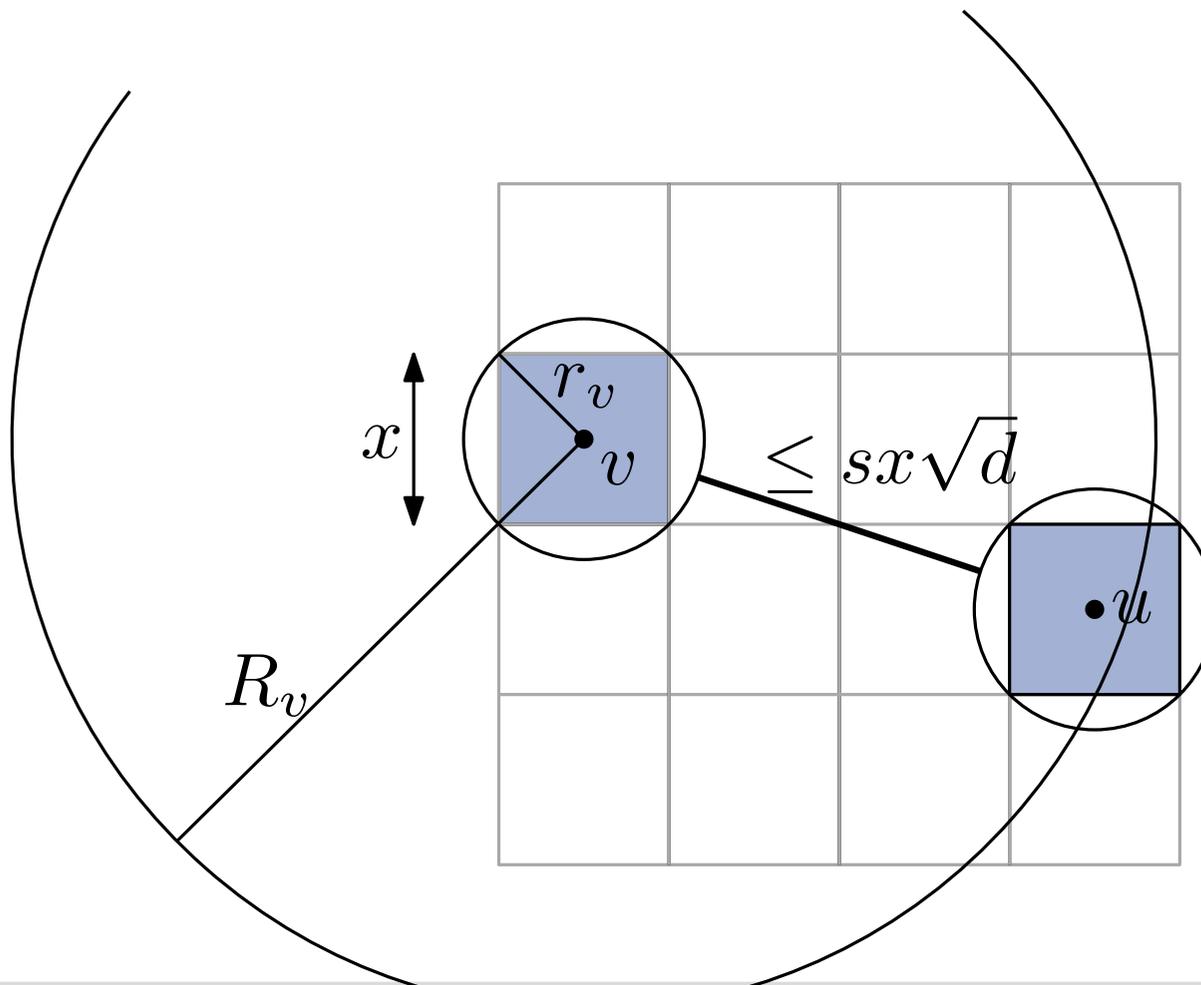
**Beweisskizze:**



# Analyse WSPD-Konstruktion

**Satz 3:** Gegeben eine Punktmenge  $P$  im  $\mathbb{R}^d$  und  $s \geq 1$  so lässt sich eine  $s$ -WSPD mit  $O(s^d n)$  Paaren in Zeit  $O(n \log n + s^d n)$  konstruieren.

**Beweisskizze:**



**Wdh. Lemma 2:**

Geg. Kugel  $K$  mit Radius  $r$  im  $\mathbb{R}^d$  und Menge  $X$  paarw. disjunkter Quadtree-Zellen mit Seitenlänge  $\geq x$ , die  $K$  schneiden. Dann gilt

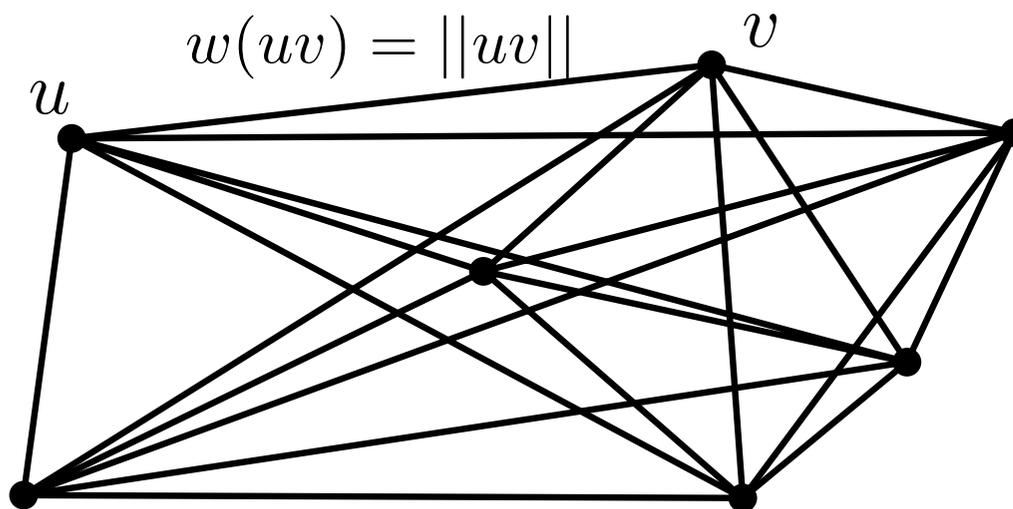
$$|X| \leq (1 + \lceil 2r/x \rceil)^d.$$

# $t$ -Spanner

Für eine Menge  $P$  von  $n$  Punkten in  $\mathbb{R}^d$  ist der **Euklidische Graph**  $\mathcal{EG}(P) = (P, \binom{P}{2})$  der vollständige gewichtete Graph, dessen Kantengewichte dem Euklidischen Abstand der Kantenendpunkte entsprechen.

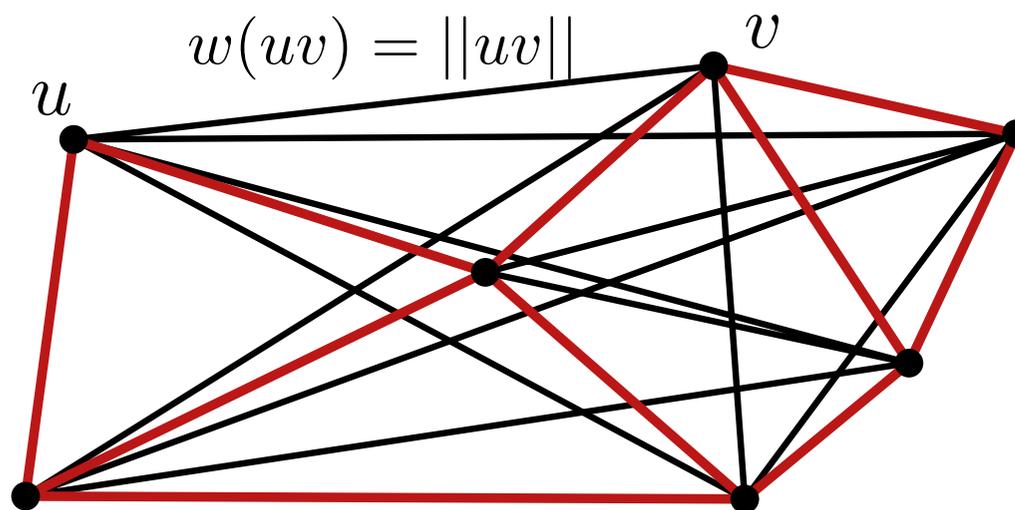
# $t$ -Spanner

Für eine Menge  $P$  von  $n$  Punkten in  $\mathbb{R}^d$  ist der **Euklidische Graph**  $\mathcal{EG}(P) = (P, \binom{P}{2})$  der vollständige gewichtete Graph, dessen Kantengewichte dem Euklidischen Abstand der Kantenendpunkte entsprechen.



Für eine Menge  $P$  von  $n$  Punkten in  $\mathbb{R}^d$  ist der **Euklidische Graph**  $\mathcal{EG}(P) = (P, \binom{P}{2})$  der vollständige gewichtete Graph, dessen Kantengewichte dem Euklidischen Abstand der Kantenendpunkte entsprechen.

Da  $\mathcal{EG}(P) \Theta(n^2)$  Kanten hat, wird oft ein dünner Graph mit  $O(n)$  Kanten gesucht, dessen kürzeste Wege die Kantengewichte in  $\mathcal{EG}(P)$  approximieren.



Für eine Menge  $P$  von  $n$  Punkten in  $\mathbb{R}^d$  ist der **Euklidische Graph**  $\mathcal{EG}(P) = (P, \binom{P}{2})$  der vollständige gewichtete Graph, dessen Kantengewichte dem Euklidischen Abstand der Kantenendpunkte entsprechen.

Da  $\mathcal{EG}(P) \Theta(n^2)$  Kanten hat, wird oft ein dünner Graph mit  $O(n)$  Kanten gesucht, dessen kürzeste Wege die Kantengewichte in  $\mathcal{EG}(P)$  approximieren.

**Def.:** Ein gewichteter Graph  $G$  mit Knotenmenge  $P$  heißt  **$t$ -Spanner** für  $P$  und einen Dehnungsfaktor  $t \geq 1$ , falls für alle Paare  $x, y \in P$  gilt

$$\|xy\| \leq \delta_G(x, y) \leq t \cdot \|xy\|,$$

wobei  $\delta_G(x, y) =$  Länge kürzester  $x$ - $y$ -Weg in  $G$ .

**Def.:** Für  $n$  Punkte  $P$  in  $\mathbb{R}^d$  und eine WSPD  $W$  von  $P$  definiere den Graphen  $G = (P, E)$  mit  
 $E = \{\{x, y\} \mid \{u, v\} \in W \text{ und } \text{rep}(u) = x, \text{rep}(v) = y\}$ .

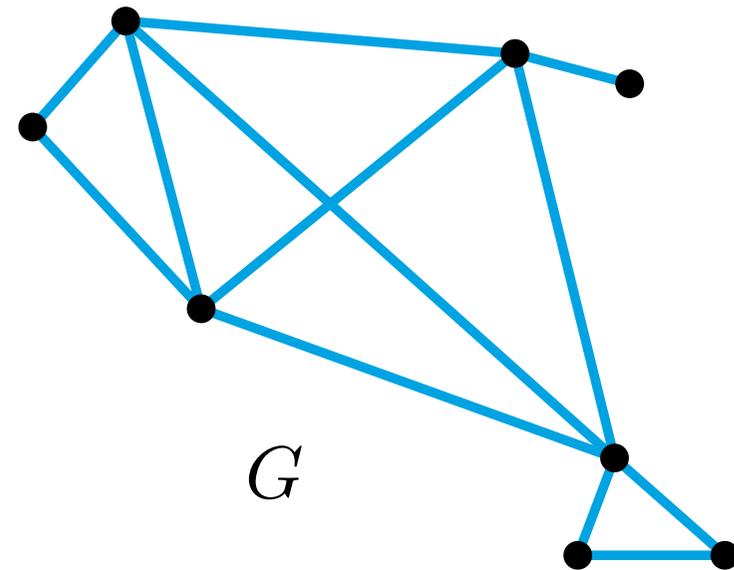
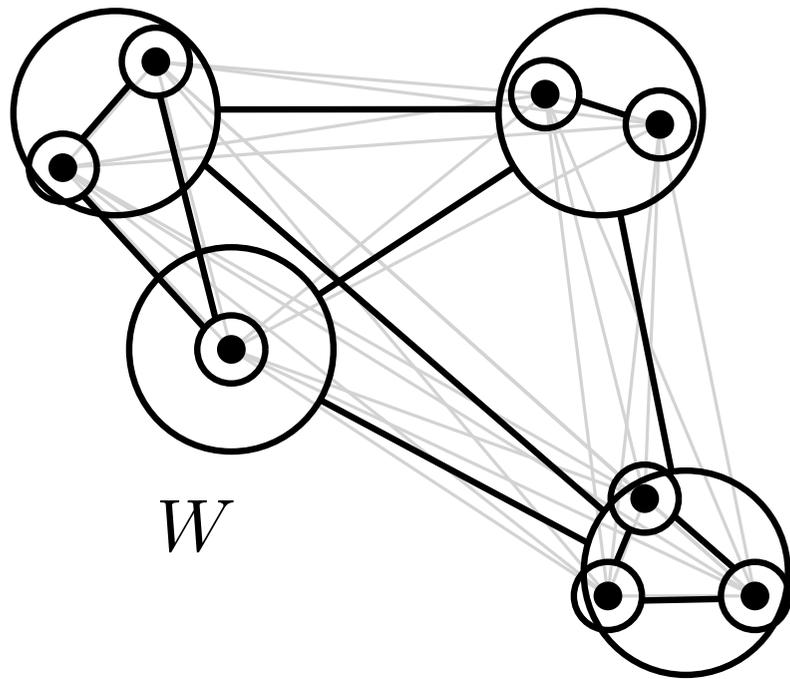
**Wdh.:** Jedes Paar  $\{u, v\} \in W$  entspricht zwei Quadtreeknoten  $u$  und  $v$ . Aus jedem Quadtreeknoten wird wie folgt ein Repräsentant gewählt. Für Blatt  $u$  definiere den Repräsentanten

$$\text{rep}(u) = \begin{cases} p & \text{falls } P_u = \{p\} \text{ (} u \text{ ist Blatt)} \\ \emptyset & \text{sonst.} \end{cases}$$

Für einen inneren Knoten  $v$  setze  $\text{rep}(v) = \text{rep}(u)$  für ein nichtleeres Kind  $u$  von  $v$ .

# WSPD und $t$ -Spanner

**Def.:** Für  $n$  Punkte  $P$  in  $\mathbb{R}^d$  und eine WSPD  $W$  von  $P$  definiere den Graphen  $G = (P, E)$  mit  
 $E = \{\{x, y\} \mid \{u, v\} \in W \text{ und } \text{rep}(u) = x, \text{rep}(v) = y\}$ .



# WSPD und $t$ -Spanner

**Def.:** Für  $n$  Punkte  $P$  in  $\mathbb{R}^d$  und eine WSPD  $W$  von  $P$  definiere den Graphen  $G = (P, E)$  mit

$$E = \{\{x, y\} \mid \{u, v\} \in W \text{ und } \text{rep}(u) = x, \text{rep}(v) = y\}.$$

**Lemma 3:** Ist  $W$  eine  $s$ -WSPD für ein geeignetes  $s = s(t)$ , so ist  $G$  ein  $t$ -Spanner für  $P$  mit  $O(s^d n)$  Kanten.

# WSPD und $t$ -Spanner

**Def.:** Für  $n$  Punkte  $P$  in  $\mathbb{R}^d$  und eine WSPD  $W$  von  $P$  definiere den Graphen  $G = (P, E)$  mit

$$E = \{\{x, y\} \mid \{u, v\} \in W \text{ und } \text{rep}(u) = x, \text{rep}(v) = y\}.$$

**Lemma 3:** Ist  $W$  eine  $s$ -WSPD für ein geeignetes  $s = s(t)$ , so ist  $G$  ein  $t$ -Spanner für  $P$  mit  $O(s^d n)$  Kanten.

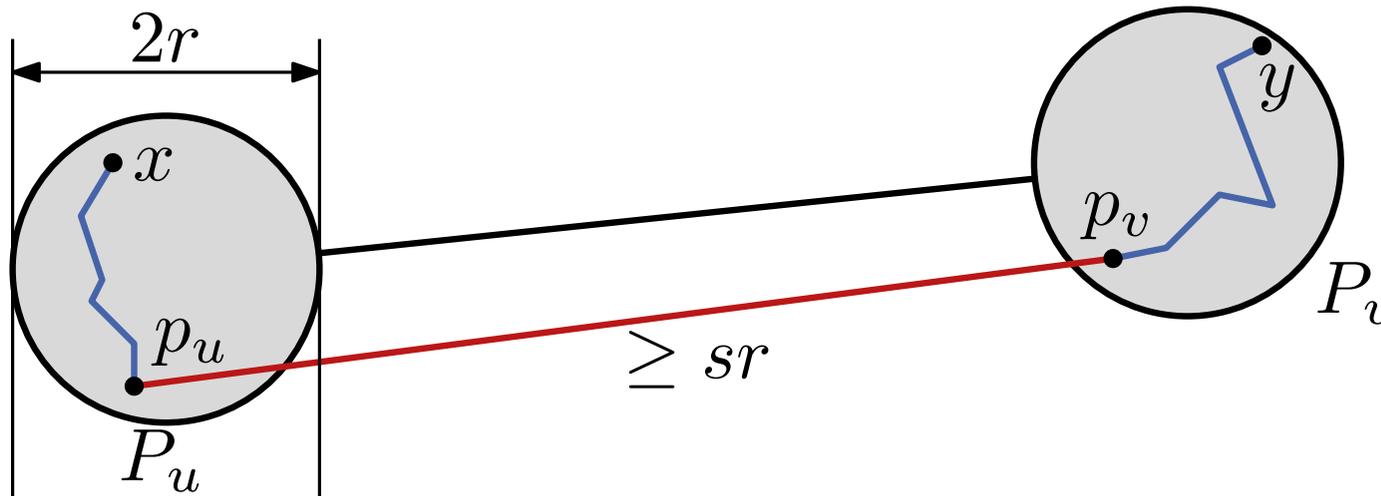
**Beweis:** (Tafel)

# WSPD und $t$ -Spanner

**Def.:** Für  $n$  Punkte  $P$  in  $\mathbb{R}^d$  und eine WSPD  $W$  von  $P$  definiere den Graphen  $G = (P, E)$  mit  
 $E = \{\{x, y\} \mid \{u, v\} \in W \text{ und } \text{rep}(u) = x, \text{rep}(v) = y\}$ .

**Lemma 3:** Ist  $W$  eine  $s$ -WSPD für ein geeignetes  $s = s(t)$ , so ist  $G$  ein  $t$ -Spanner für  $P$  mit  $O(s^d n)$  Kanten.

**Beweis:** (Tafel)



**Satz 4:** Für eine Menge  $P$  von  $n$  Punkten in  $\mathbb{R}^d$  und ein  $\varepsilon \in (0, 1]$  kann ein  $(1 + \varepsilon)$ -Spanner für  $P$  mit  $O(n/\varepsilon^d)$  Kanten in  $O(n \log n + n/\varepsilon^d)$  Zeit berechnet werden.

**Satz 4:** Für eine Menge  $P$  von  $n$  Punkten in  $\mathbb{R}^d$  und ein  $\varepsilon \in (0, 1]$  kann ein  $(1 + \varepsilon)$ -Spanner für  $P$  mit  $O(n/\varepsilon^d)$  Kanten in  $O(n \log n + n/\varepsilon^d)$  Zeit berechnet werden.

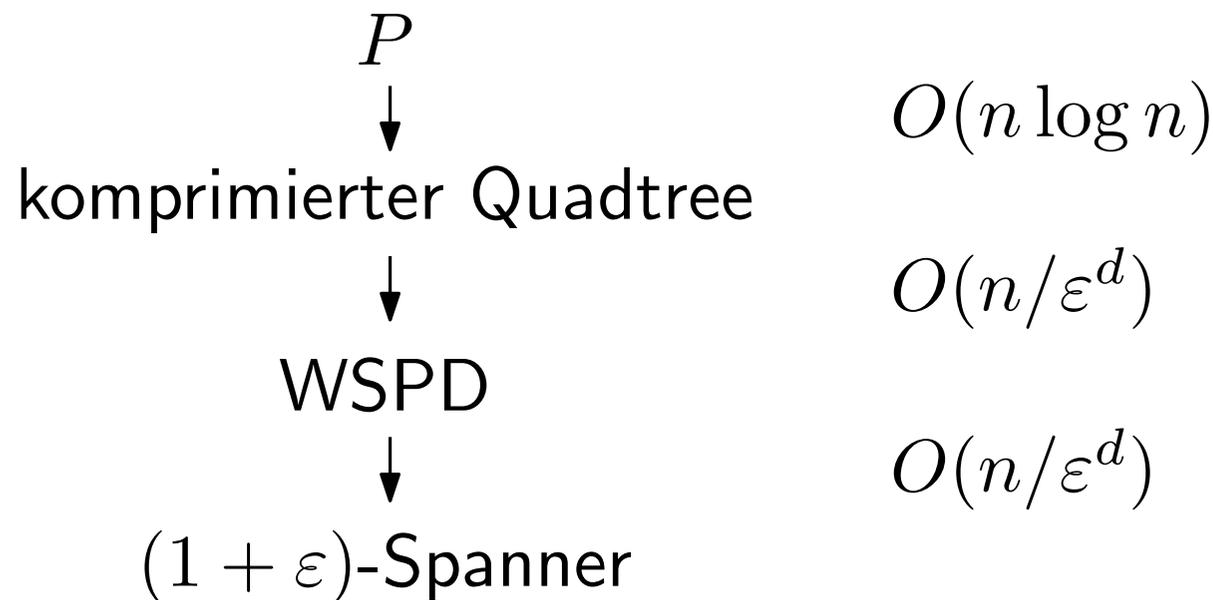
**Beweis:** Für  $t = (1 + \varepsilon)$  gilt mit  $s = 4 \cdot \frac{t+1}{t-1}$

$$O(s^d n) = O\left(\left(4 \cdot \frac{2 + \varepsilon}{\varepsilon}\right)^d n\right) \subseteq O\left(\left(\frac{12}{\varepsilon}\right)^d n\right) = O\left(\frac{n}{\varepsilon^d}\right)$$

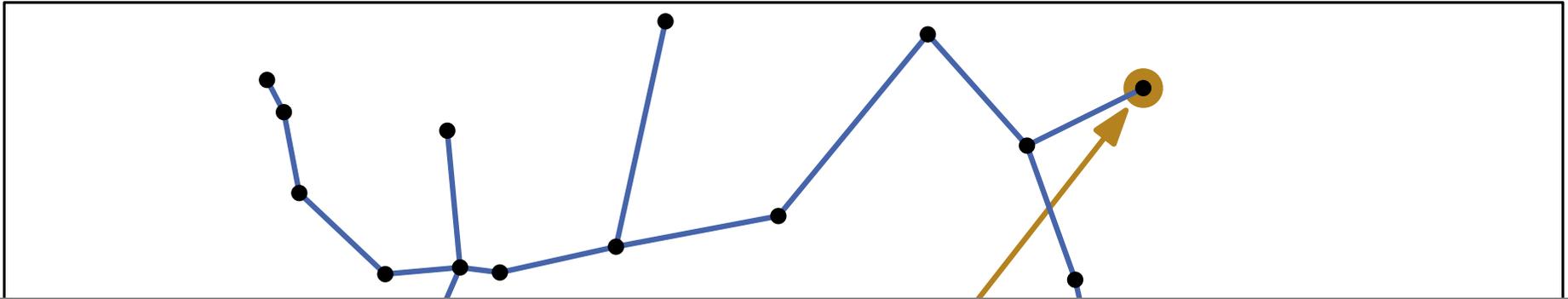
**Satz 4:** Für eine Menge  $P$  von  $n$  Punkten in  $\mathbb{R}^d$  und ein  $\varepsilon \in (0, 1]$  kann ein  $(1 + \varepsilon)$ -Spanner für  $P$  mit  $O(n/\varepsilon^d)$  Kanten in  $O(n \log n + n/\varepsilon^d)$  Zeit berechnet werden.

**Beweis:** Für  $t = (1 + \varepsilon)$  gilt mit  $s = 4 \cdot \frac{t+1}{t-1}$

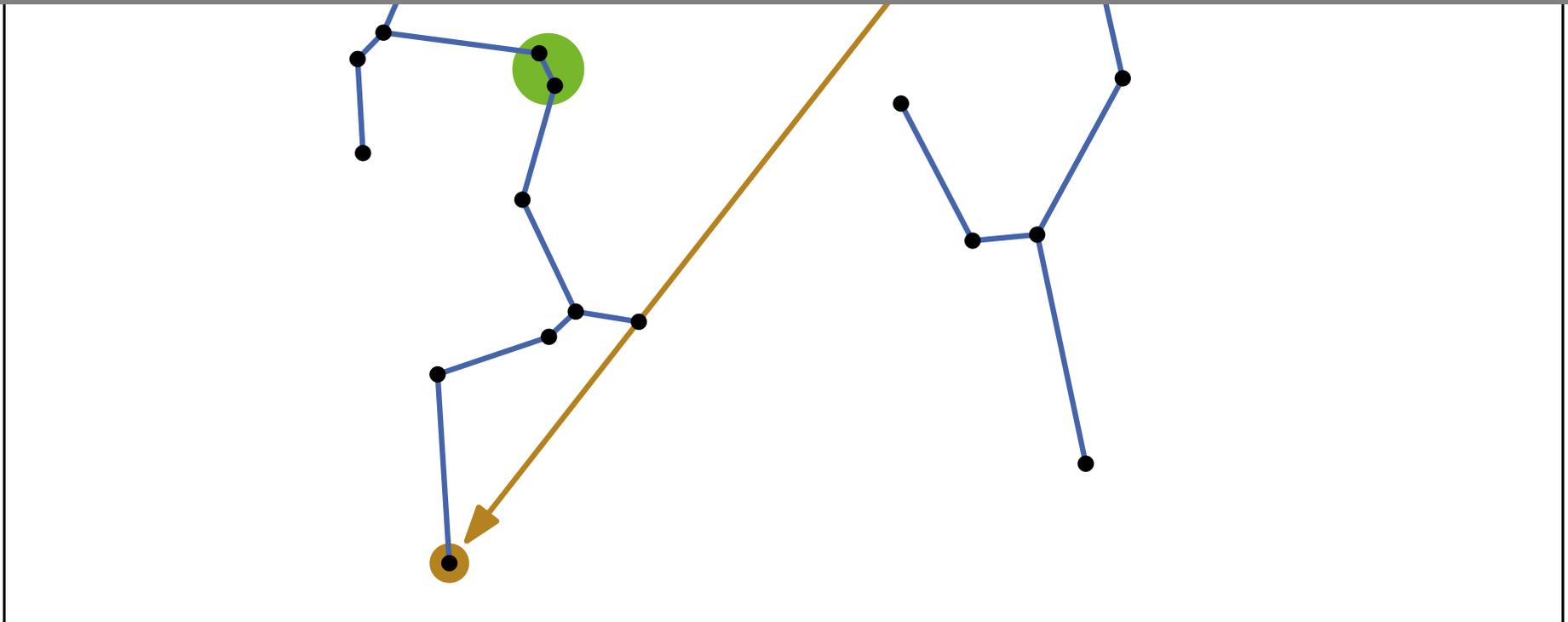
$$O(s^d n) = O\left(\left(4 \cdot \frac{2 + \varepsilon}{\varepsilon}\right)^d n\right) \subseteq O\left(\left(\frac{12}{\varepsilon}\right)^d n\right) = O\left(\frac{n}{\varepsilon^d}\right)$$



□



## Weitere Anwendungen der WSPD



# Euklidischer MST

**Problem:** Finde für eine Punktmenge  $P$  einen minimalen Spannbaum (MST) im Euklidischen Graphen  $\mathcal{EG}(P)$ .

# Euklidischer MST

**Problem:** Finde für eine Punktmenge  $P$  einen minimalen Spannbaum (MST) im Euklidischen Graphen  $\mathcal{EG}(P)$ .

**Prim:** MST in einem Graph  $G = (V, E)$  kann in  $O(|E| + |V| \log |V|)$  Zeit berechnet werden.

**Problem:** Finde für eine Punktmenge  $P$  einen minimalen Spannbaum (MST) im Euklidischen Graphen  $\mathcal{EG}(P)$ .

**Prim:** MST in einem Graph  $G = (V, E)$  kann in  $O(|E| + |V| \log |V|)$  Zeit berechnet werden.

- $\mathcal{EG}(P)$  hat  $\Theta(n^2)$  Kanten  $\Rightarrow$  Laufzeit  $O(n^2)$  :-(
- $(1 + \varepsilon)$ -Spanner für  $P$  hat  $O(n/\varepsilon^d)$  Kanten  $\Rightarrow$  Laufzeit  $O(n \log n + n/\varepsilon^d)$  :-)

**Problem:** Finde für eine Punktmenge  $P$  einen minimalen Spannbaum (MST) im Euklidischen Graphen  $\mathcal{EG}(P)$ .

**Prim:** MST in einem Graph  $G = (V, E)$  kann in  $O(|E| + |V| \log |V|)$  Zeit berechnet werden.

- $\mathcal{EG}(P)$  hat  $\Theta(n^2)$  Kanten  $\Rightarrow$  Laufzeit  $O(n^2)$  :-)
- $(1 + \varepsilon)$ -Spanner für  $P$  hat  $O(n/\varepsilon^d)$  Kanten  $\Rightarrow$  Laufzeit  $O(n \log n + n/\varepsilon^d)$  :-)

Wie gut ist der MST eines  $(1 + \varepsilon)$ -Spanners?

**Problem:** Finde für eine Punktmenge  $P$  einen minimalen Spannbaum (MST) im Euklidischen Graphen  $\mathcal{EG}(P)$ .

**Prim:** MST in einem Graph  $G = (V, E)$  kann in  $O(|E| + |V| \log |V|)$  Zeit berechnet werden.

- $\mathcal{EG}(P)$  hat  $\Theta(n^2)$  Kanten  $\Rightarrow$  Laufzeit  $O(n^2)$  :-)
- $(1 + \varepsilon)$ -Spanner für  $P$  hat  $O(n/\varepsilon^d)$  Kanten  $\Rightarrow$  Laufzeit  $O(n \log n + n/\varepsilon^d)$  :-)

Wie gut ist der MST eines  $(1 + \varepsilon)$ -Spanners?

**Satz 5:** Der aus einem  $(1 + \varepsilon)$ -Spanner von  $P$  gewonnene MST ist eine  $(1 + \varepsilon)$ -Approximation des EMST von  $P$ .

# Durchmesser von $P$

**Problem:** Finde den Durchmesser einer Punktmenge  $P$ , d.h. das Paar  $\{x, y\} \subset P$  mit größtem Abstand.

# Durchmesser von $P$

**Problem:** Finde den Durchmesser einer Punktmenge  $P$ , d.h. das Paar  $\{x, y\} \subset P$  mit größtem Abstand.

- brute-force alle Punktpaare testen  $\Rightarrow$  Laufzeit  $O(n^2)$  :-)
- teste Abstände  $\| \text{rep}(u) - \text{rep}(v) \|$  aller ws-Paare  $\{P_u, P_v\}$   
 $\Rightarrow$  Laufzeit  $O(n \log n + s^d n)$  :-)

# Durchmesser von $P$

**Problem:** Finde den Durchmesser einer Punktmenge  $P$ , d.h. das Paar  $\{x, y\} \subset P$  mit größtem Abstand.

- brute-force alle Punktpaare testen  $\Rightarrow$  Laufzeit  $O(n^2)$  :-)
- teste Abstände  $\| \text{rep}(u) - \text{rep}(v) \|$  aller ws-Paare  $\{P_u, P_v\}$   
 $\Rightarrow$  Laufzeit  $O(n \log n + s^d n)$  :-)

Wie gut ist der berechnete Durchmesser?

# Durchmesser von $P$

**Problem:** Finde den Durchmesser einer Punktmenge  $P$ , d.h. das Paar  $\{x, y\} \subset P$  mit größtem Abstand.

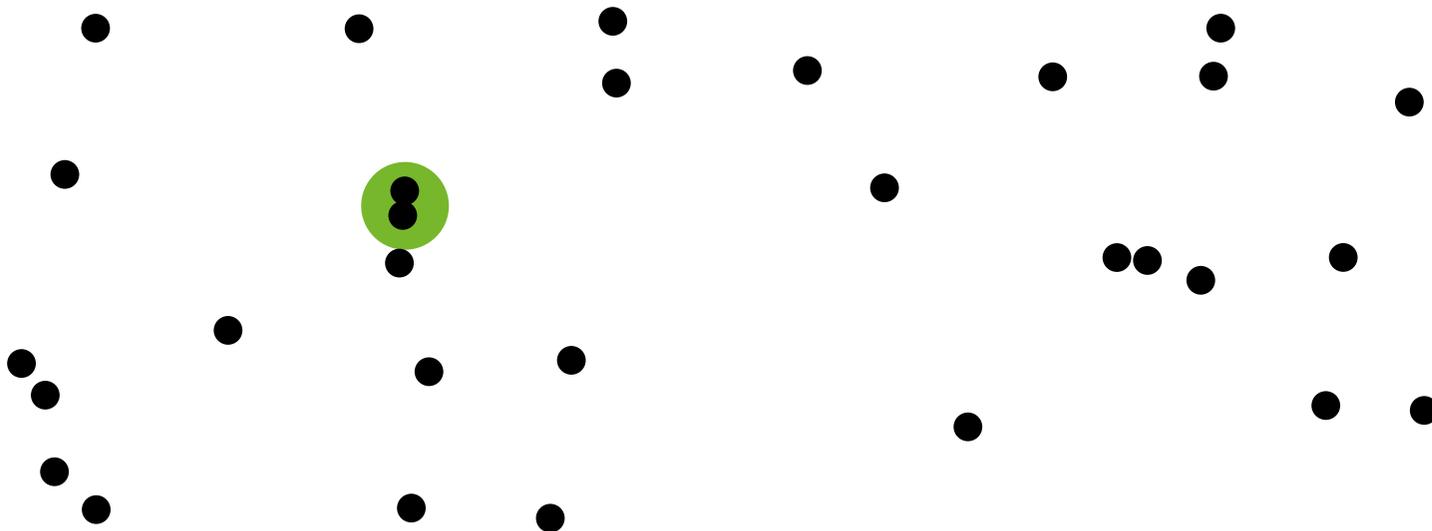
- brute-force alle Punktpaare testen  $\Rightarrow$  Laufzeit  $O(n^2)$  :-)
- teste Abstände  $\| \text{rep}(u) - \text{rep}(v) \|$  aller ws-Paare  $\{P_u, P_v\}$   
 $\Rightarrow$  Laufzeit  $O(n \log n + s^d n)$  :-)

Wie gut ist der berechnete Durchmesser?

**Satz 6:** Der aus einer  $s$ -WSPD von  $P$  berechnete Durchmesser ist eine  $(1 + \varepsilon)$ -Approximation des Durchmessers von  $P$ .

# Nächstes Punktepaar

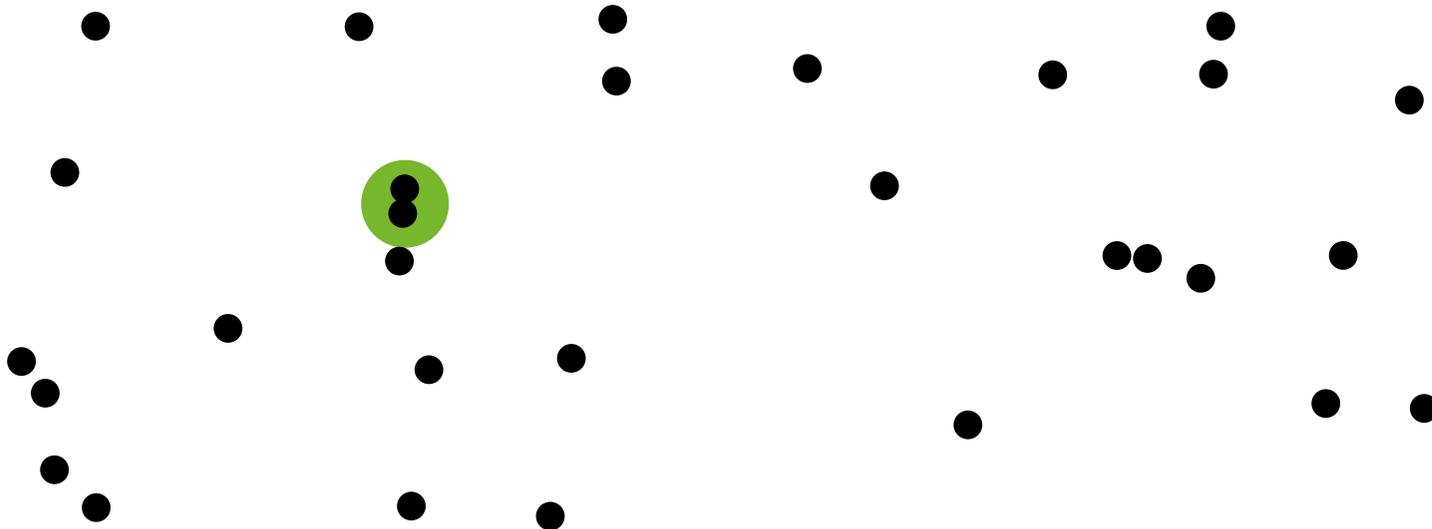
**Problem:** Finde das Paar  $\{x, y\} \subset P$  mit geringstem Abstand.



# Nächstes Punktepaar

**Problem:** Finde das Paar  $\{x, y\} \subset P$  mit geringstem Abstand.

- brute-force alle Punktepaare testen  $\Rightarrow$  Laufzeit  $O(n^2)$  :-)
- teste Abstände  $\|rep(u) - rep(v)\|$  aller ws-Paare  $\{P_u, P_v\}$   
 $\Rightarrow$  Laufzeit  $O(n \log n + s^d n)$  :-)

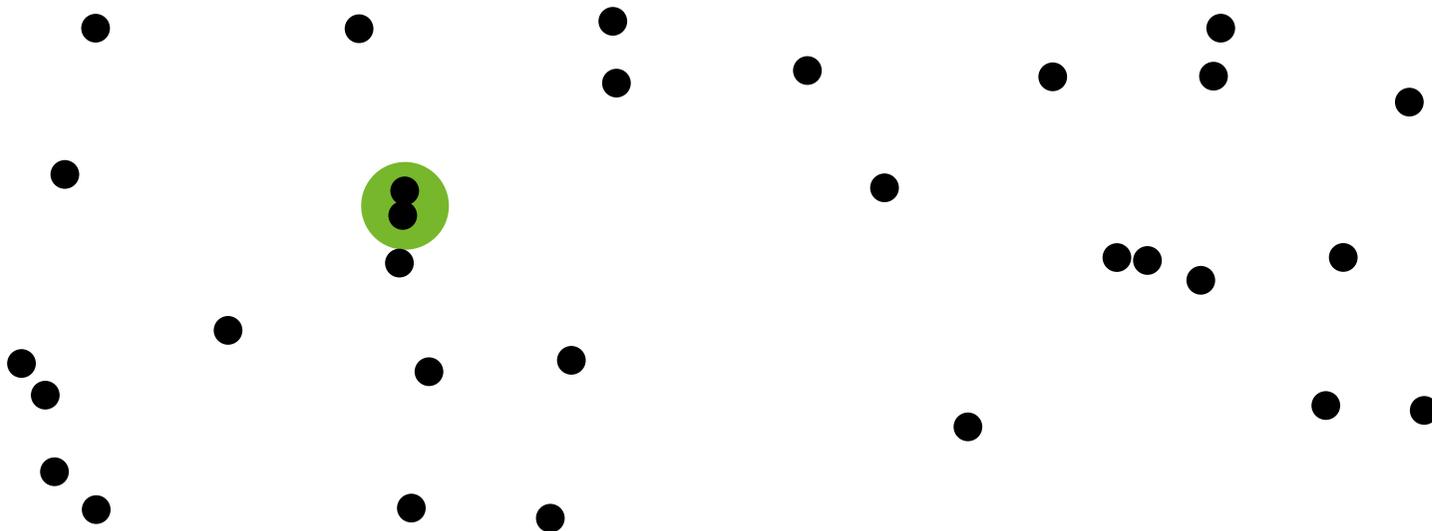


# Nächstes Punktepaar

**Problem:** Finde das Paar  $\{x, y\} \subset P$  mit geringstem Abstand.

- brute-force alle Punktepaare testen  $\Rightarrow$  Laufzeit  $O(n^2)$  :-)
- teste Abstände  $\|\text{rep}(u) - \text{rep}(v)\|$  aller ws-Paare  $\{P_u, P_v\}$   
 $\Rightarrow$  Laufzeit  $O(n \log n + s^d n)$  :-)

**Übungsblatt:** Für  $s > 2$  liefert dieser Ansatz sogar exakt das nächste Paar.



**Welche weiteren Anwendungen hat die WSPD?**

## Welche weiteren Anwendungen hat die WSPD?

WSPD bietet sich immer dann an, wenn man auf die  $\Theta(n^2)$  exakten Distanzen in einer Punktmenge verzichten kann und diese stattdessen approximiert. Ein Beispiel sind kräftebasierte Layoutalgorithmen im Graphenzeichnen, wo paarweise Abstoßungskräfte von  $n$  Objekten berechnet werden müssen.

## Welche weiteren Anwendungen hat die WSPD?

WSPD bietet sich immer dann an, wenn man auf die  $\Theta(n^2)$  exakten Distanzen in einer Punktmenge verzichten kann und diese stattdessen approximiert. Ein Beispiel sind kräftebasierte Layoutalgorithmen im Graphenzeichnen, wo paarweise Abstoßungskräfte von  $n$  Objekten berechnet werden müssen.

## Wozu geometrisch approximieren?

## Welche weiteren Anwendungen hat die WSPD?

WSPD bietet sich immer dann an, wenn man auf die  $\Theta(n^2)$  exakten Distanzen in einer Punktmenge verzichten kann und diese stattdessen approximiert. Ein Beispiel sind kräftebasierte Layoutalgorithmen im Graphenzeichnen, wo paarweise Abstoßungskräfte von  $n$  Objekten berechnet werden müssen.

## Wozu geometrisch approximieren?

Einerseits ersetzt man dadurch langsame Berechnungen durch schnellere (aber ungenauere), andererseits sind oft auch die Eingabedaten schon mit einer gewissen Ungenauigkeit behaftet, so dass approximative Lösungen je nach Anwendung ausreichend sind.

## Welche weiteren Anwendungen hat die WSPD?

WSPD bietet sich immer dann an, wenn man auf die  $\Theta(n^2)$  exakten Distanzen in einer Punktmenge verzichten kann und diese stattdessen approximiert. Ein Beispiel sind kräftebasierte Layoutalgorithmen im Graphenzeichnen, wo paarweise Abstoßungskräfte von  $n$  Objekten berechnet werden müssen.

## Wozu geometrisch approximieren?

Einerseits ersetzt man dadurch langsame Berechnungen durch schnellere (aber ungenauere), andererseits sind oft auch die Eingabedaten schon mit einer gewissen Ungenauigkeit behaftet, so dass approximative Lösungen je nach Anwendung ausreichend sind.

## Geht es nicht auch genauso schnell exakt?

## Welche weiteren Anwendungen hat die WSPD?

WSPD bietet sich immer dann an, wenn man auf die  $\Theta(n^2)$  exakten Distanzen in einer Punktmenge verzichten kann und diese stattdessen approximiert. Ein Beispiel sind kräftebasierte Layoutalgorithmen im Graphenzeichnen, wo paarweise Abstoßungskräfte von  $n$  Objekten berechnet werden müssen.

## Wozu geometrisch approximieren?

Einerseits ersetzt man dadurch langsame Berechnungen durch schnellere (aber ungenauere), andererseits sind oft auch die Eingabedaten schon mit einer gewissen Ungenauigkeit behaftet, so dass approximative Lösungen je nach Anwendung ausreichend sind.

## Geht es nicht auch genauso schnell exakt?

Oft im  $\mathbb{R}^2$  ja, aber nicht mehr im  $\mathbb{R}^d$  für  $d > 2$ . (EMST, Durchmesser)