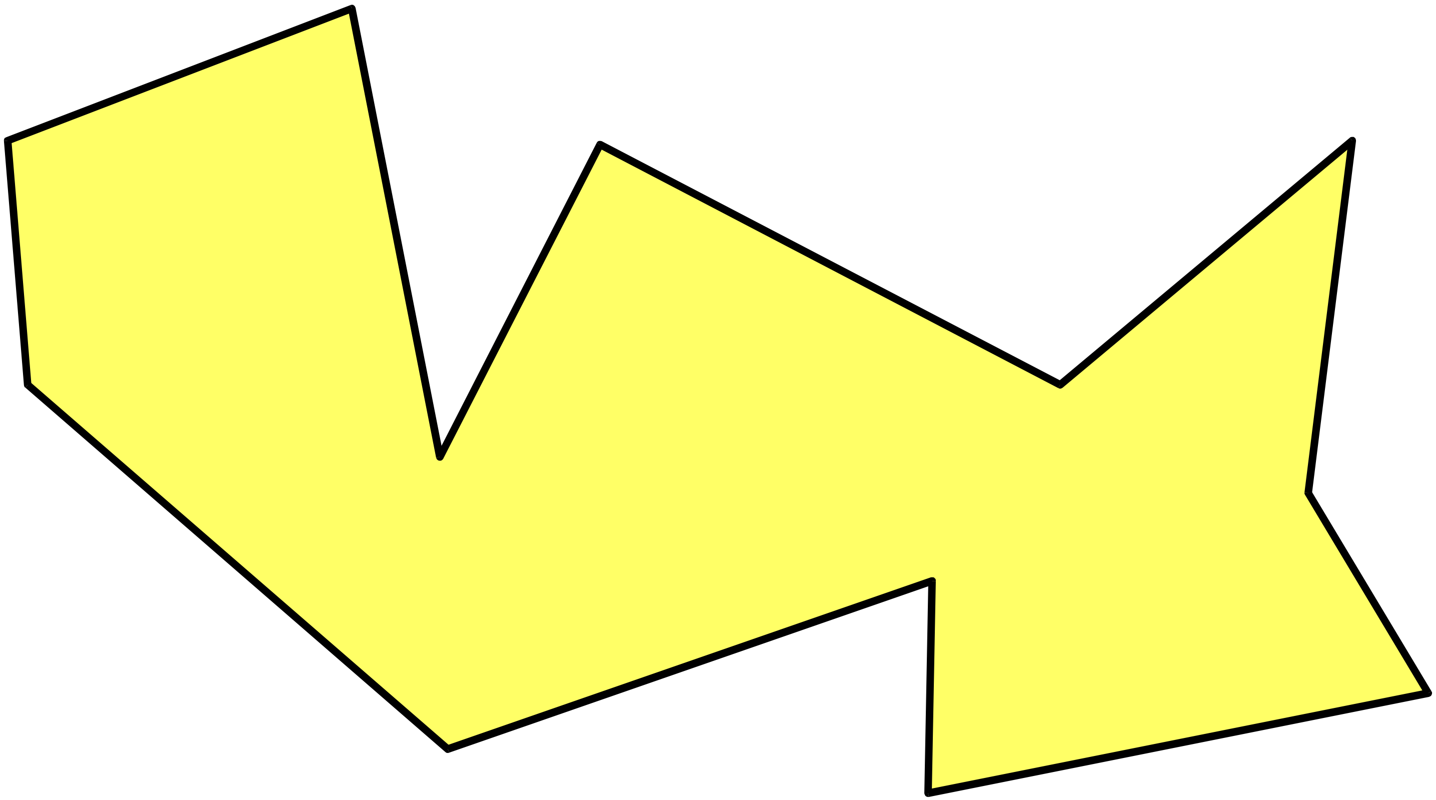
# 7. Triangulation von einfachen Polygonen

# Ziel

Bessere Laufzeit als $O(n \log n)$ durch schnelleres Berechnen der Trapezzerlegung des Polygons.

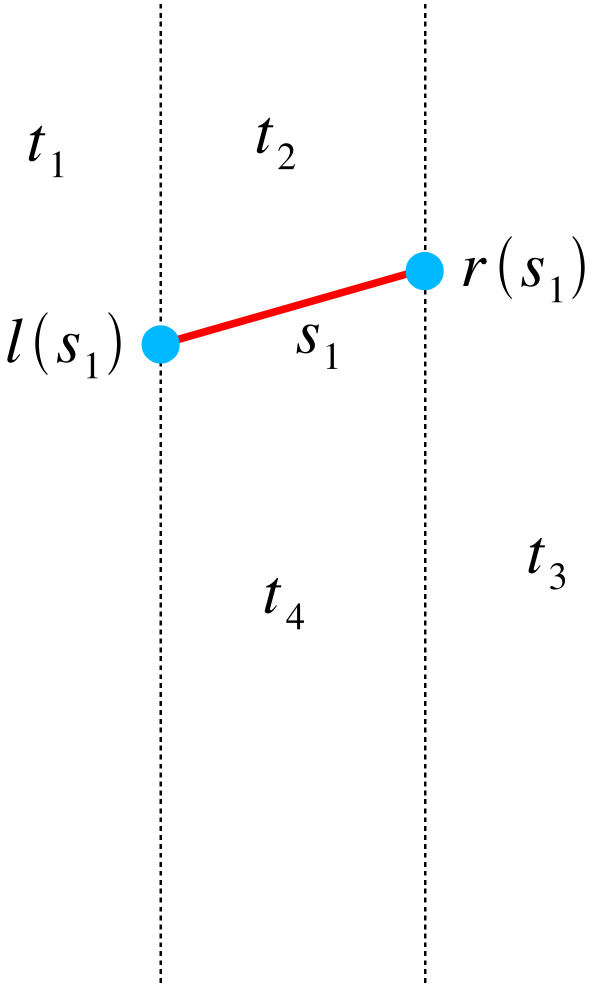


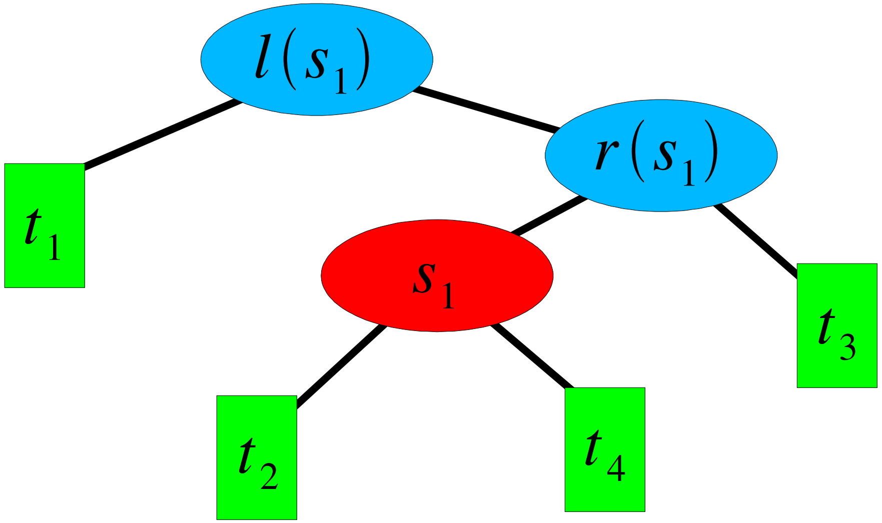
## Idee

Finde Methode, den Anfangspunkt einer Strecke in der Trapezzerlegung schnell zu lokalisieren.

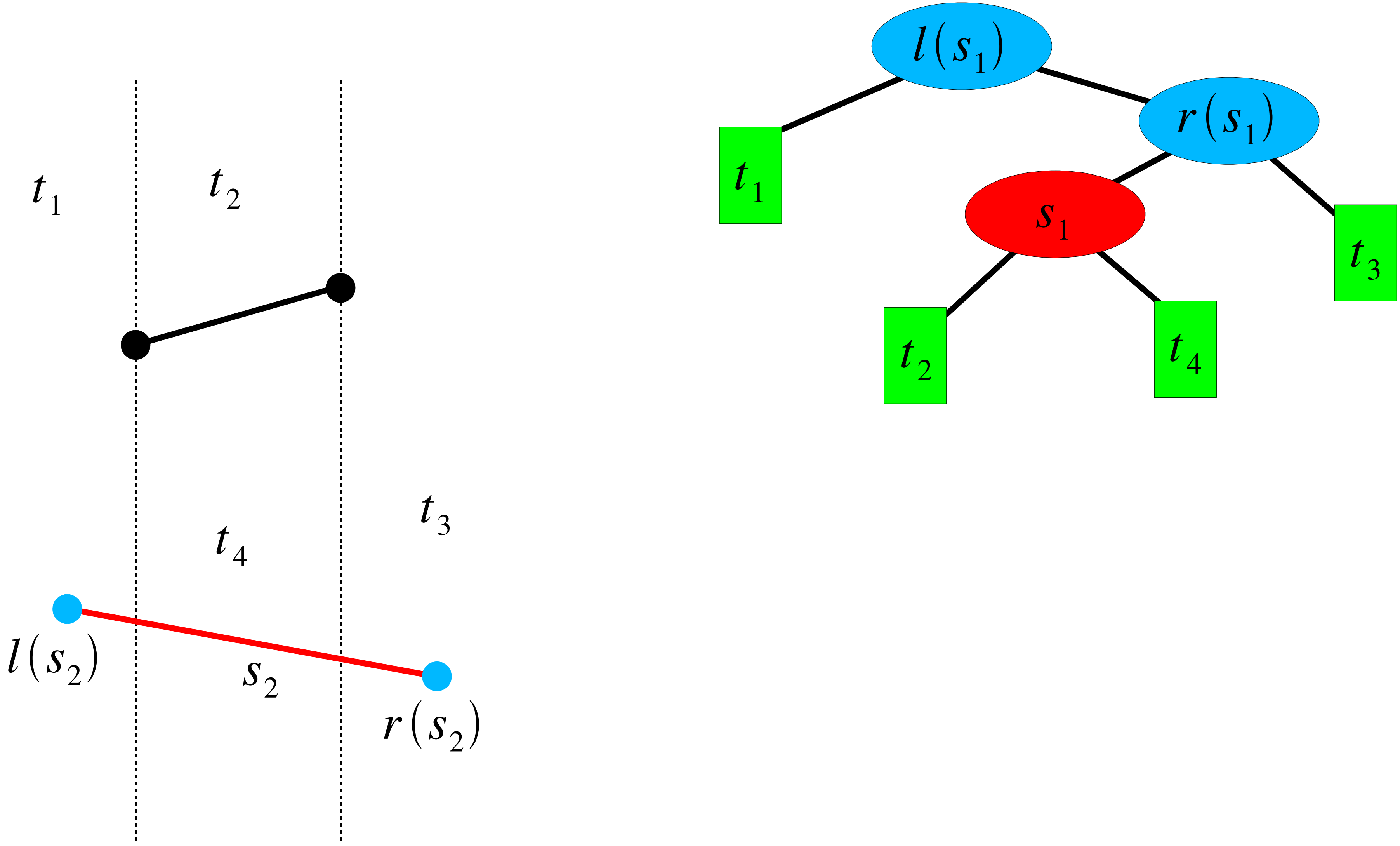
Brauchen also Ersatz für *Konfliktlisten*.

# Datenstruktur zur Punktlokalisierung

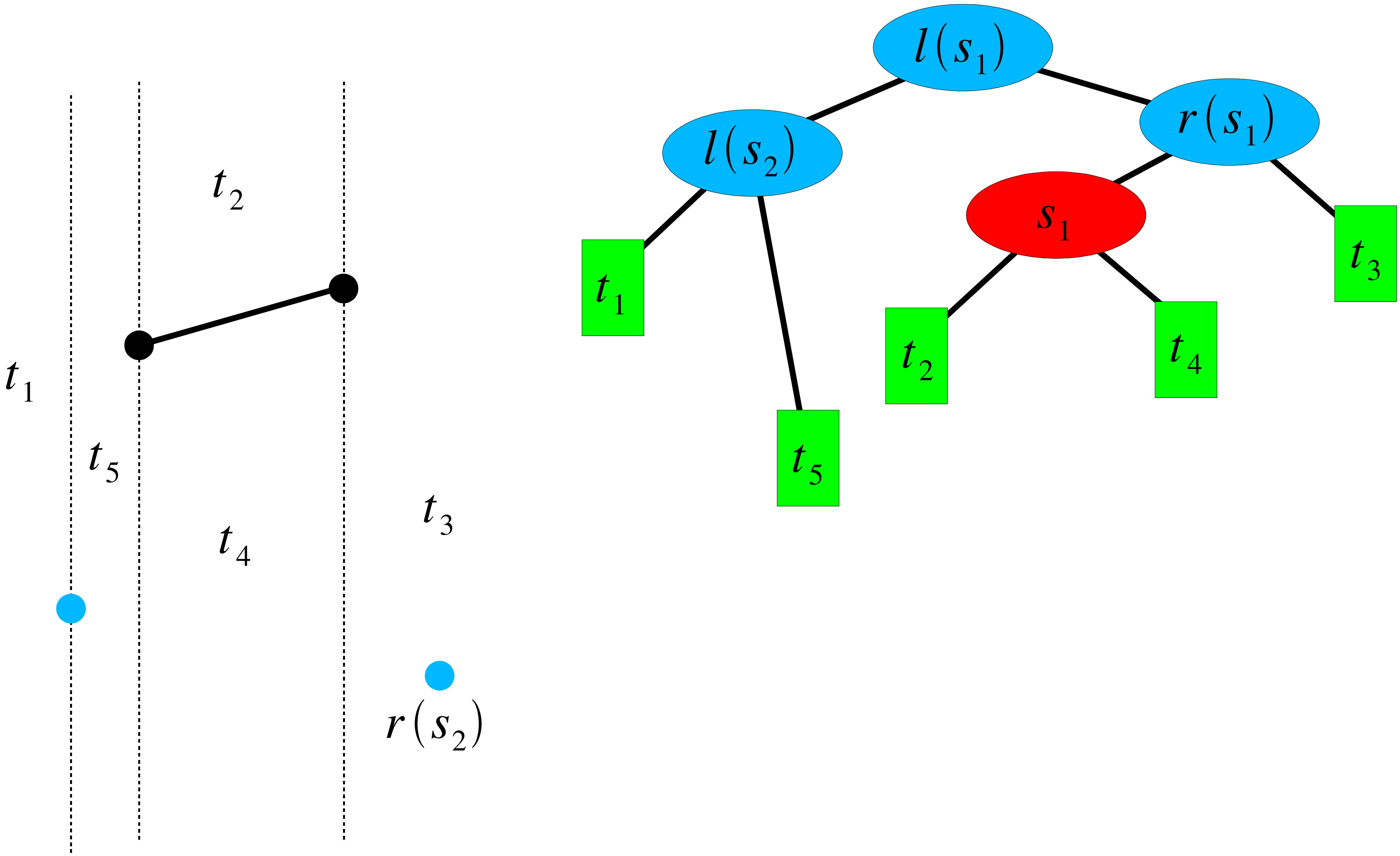


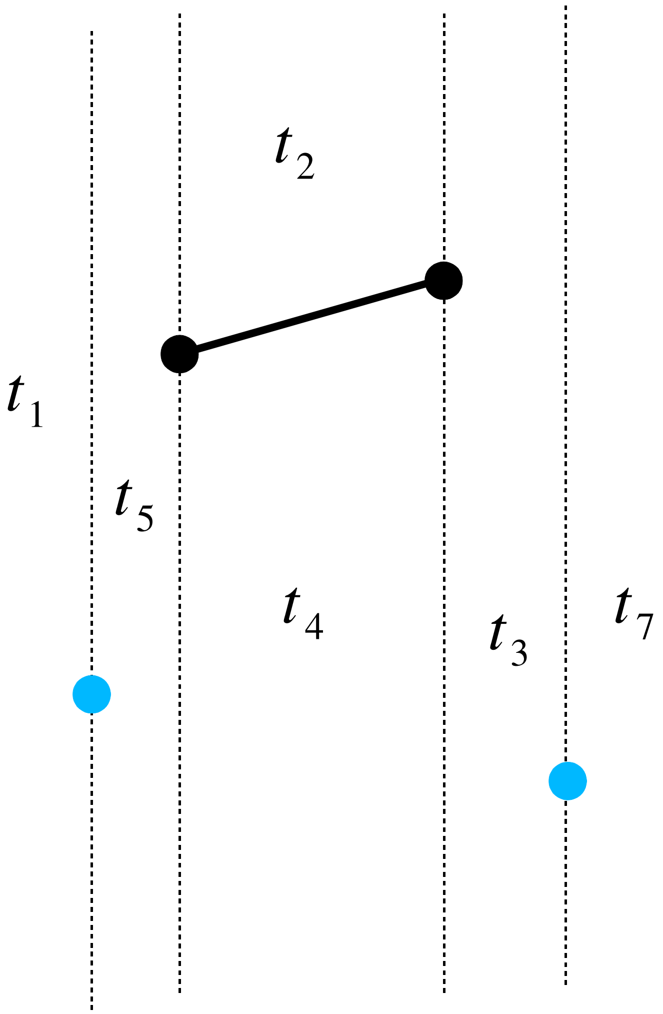
# Datenstruktur zur Punktlokalisierung

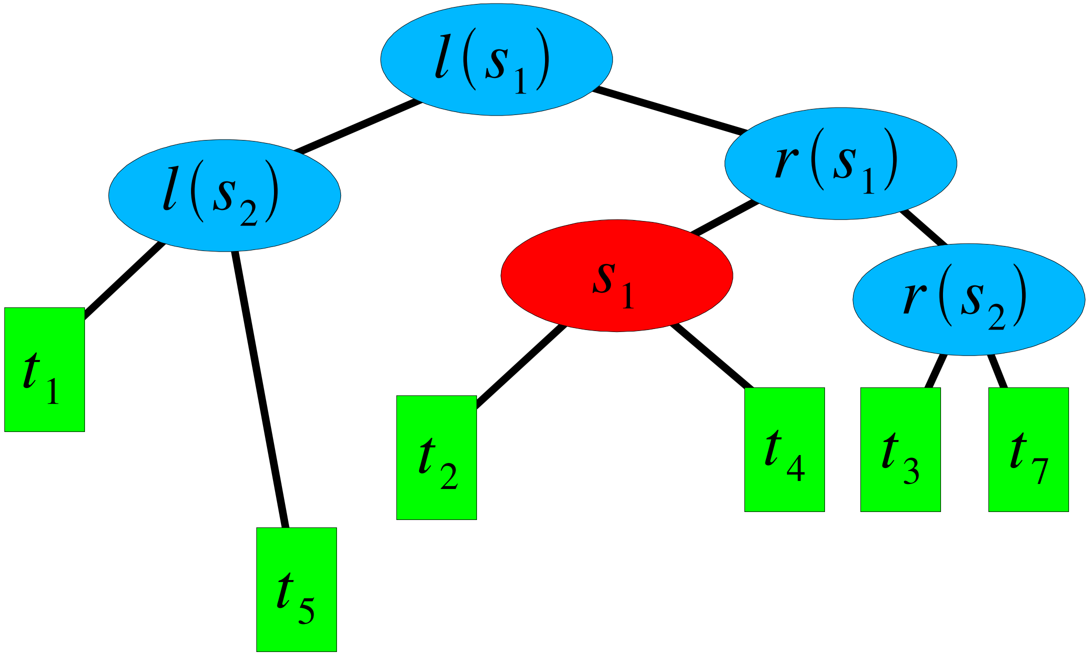


# Datenstruktur zur Punktlokalisierung

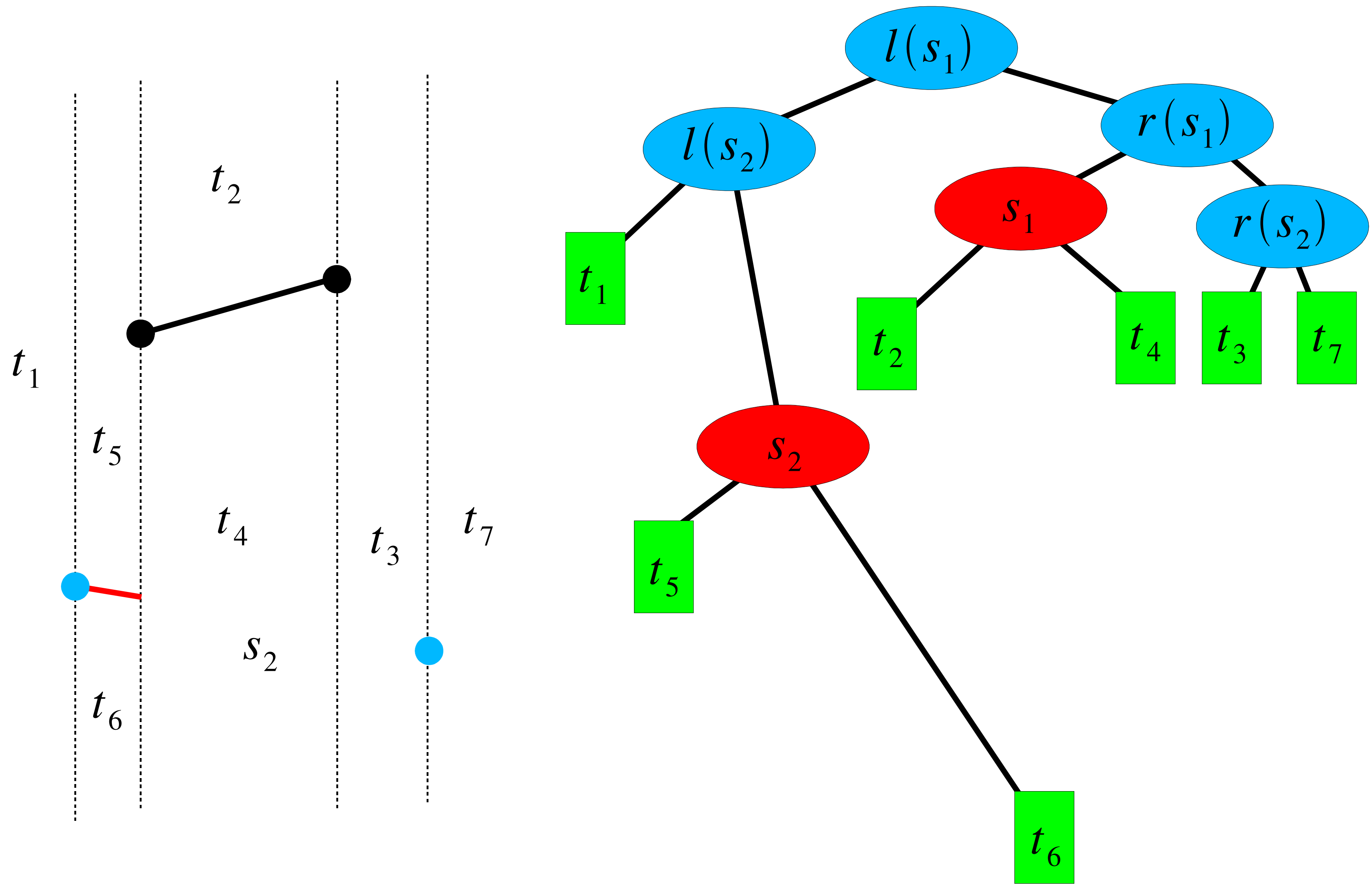


# Datenstruktur zur Punktlokalisierung

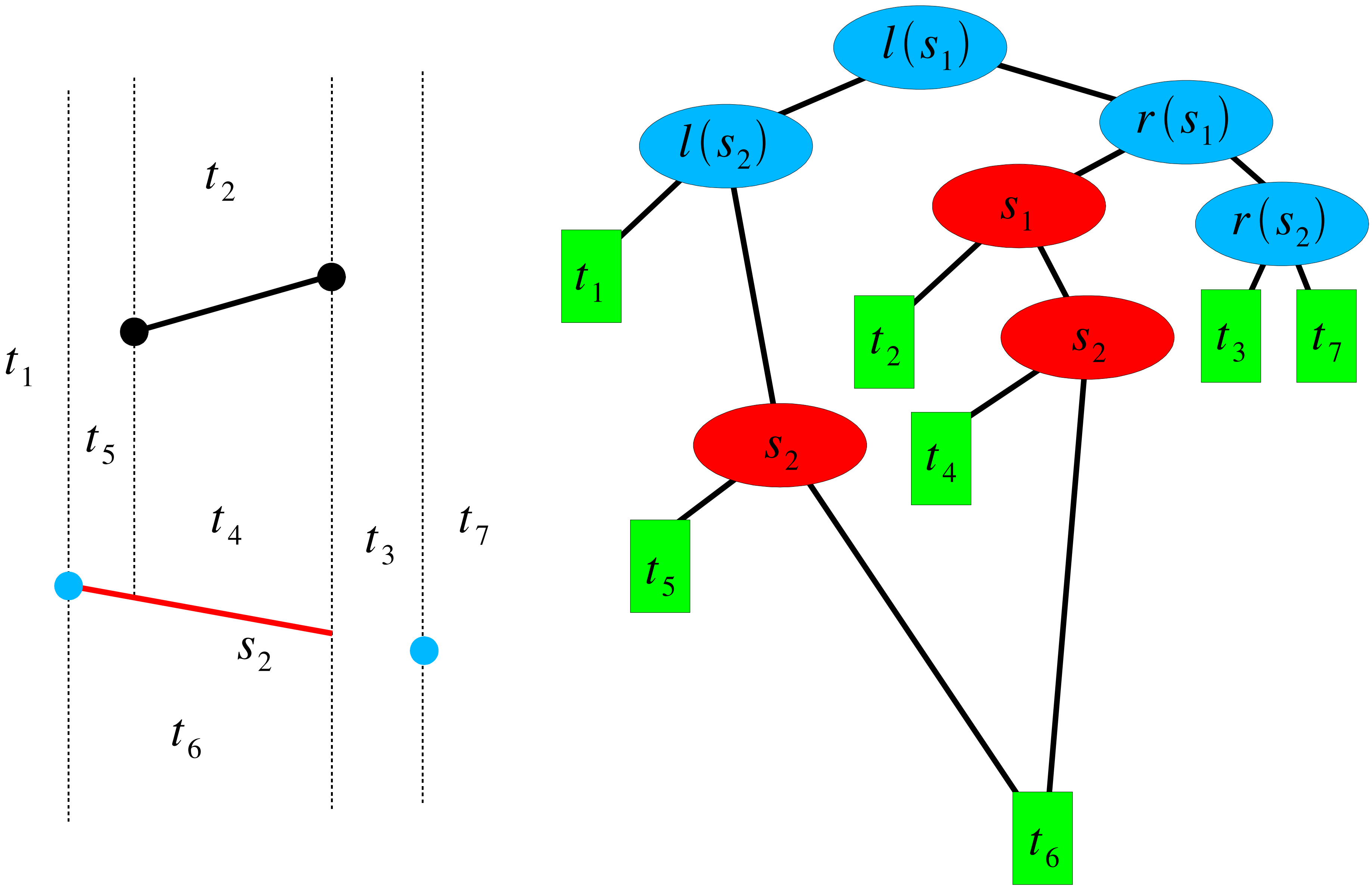


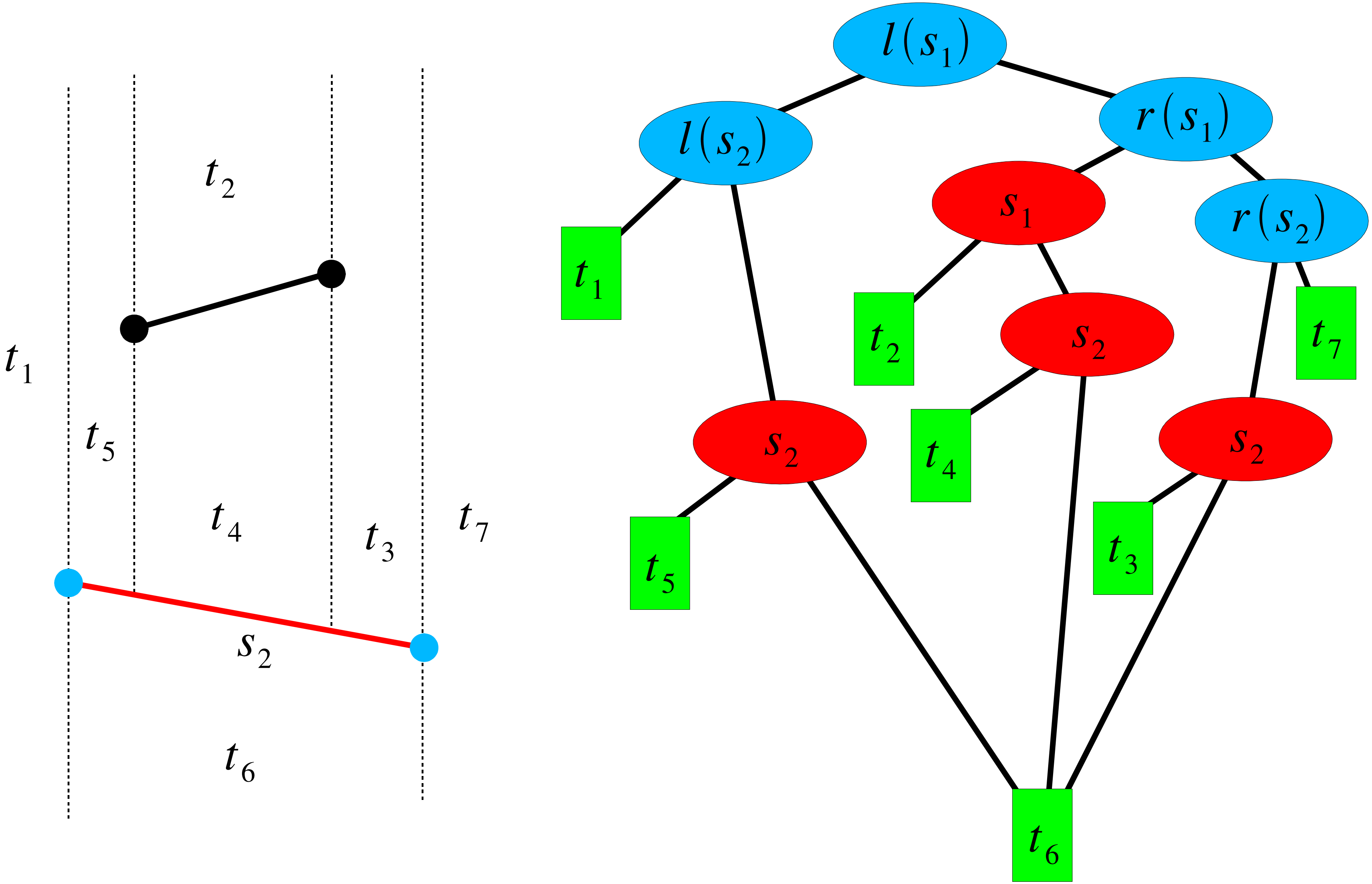
# Datenstruktur zur Punktlokalisierung



# Datenstruktur zur Punktlokalisierung



# Datenstruktur zur Punktlokalisierung



# Vorgehen allgemein

1. Linken Endpunkt lokalisieren und Trapez vertikal zerlegen.
2. Rechten Endpunkt lokalisieren und Trapez vertikal zerlegen.
3. Strecke durch die Trapezzerlegung verfolgen und Trapeze "horizontal" zerlegen.

# Erwarteter Aufwand

Verfolgen der Strecke: $O(1)$ pro Trapez

Lokalisieren der Endpunkte: ???

# Bezeichnungen

| | |
|---|---|
| $P$ | Menge der $n$ Kanten des Polygons |
| $s_1, \ldots, s_n$ | zufällige Permutation der Kanten in $P$ |
| $P_i = \{ s_1, \ldots, s_i \}$ | |
| $T(P_i)$ | Trapezzerlegung der ersten $i$ Strecken / Kanten |
| $D(P_i)$ | DS zur Punktlokalisierung für die ersten $i$ Strecken |

# Kosten der Punktlokalisierung

Lemma 1

Der erwartete Aufwand, um einen festen Punkt $q$ mit der Datenstruktur $D(P_i)$ in der Trapezzerlegung $T(P_i)$ zu lokalisieren, beträgt $O(\log i)$.

Beweis:

Sei $t_j$ das Trapez in $T(P_j)$, das $q$ enthält.

Angenommen wir kennen $t_{i-1}$.

# Kosten der Punktlokalisierung

Was ist die erwartete Anzahl von Vergleichen, um $t_i$ zu finden?

1. Fall: $t_i = t_{i-1}$

Dann sind keine weiteren Vergleiche notwendig.

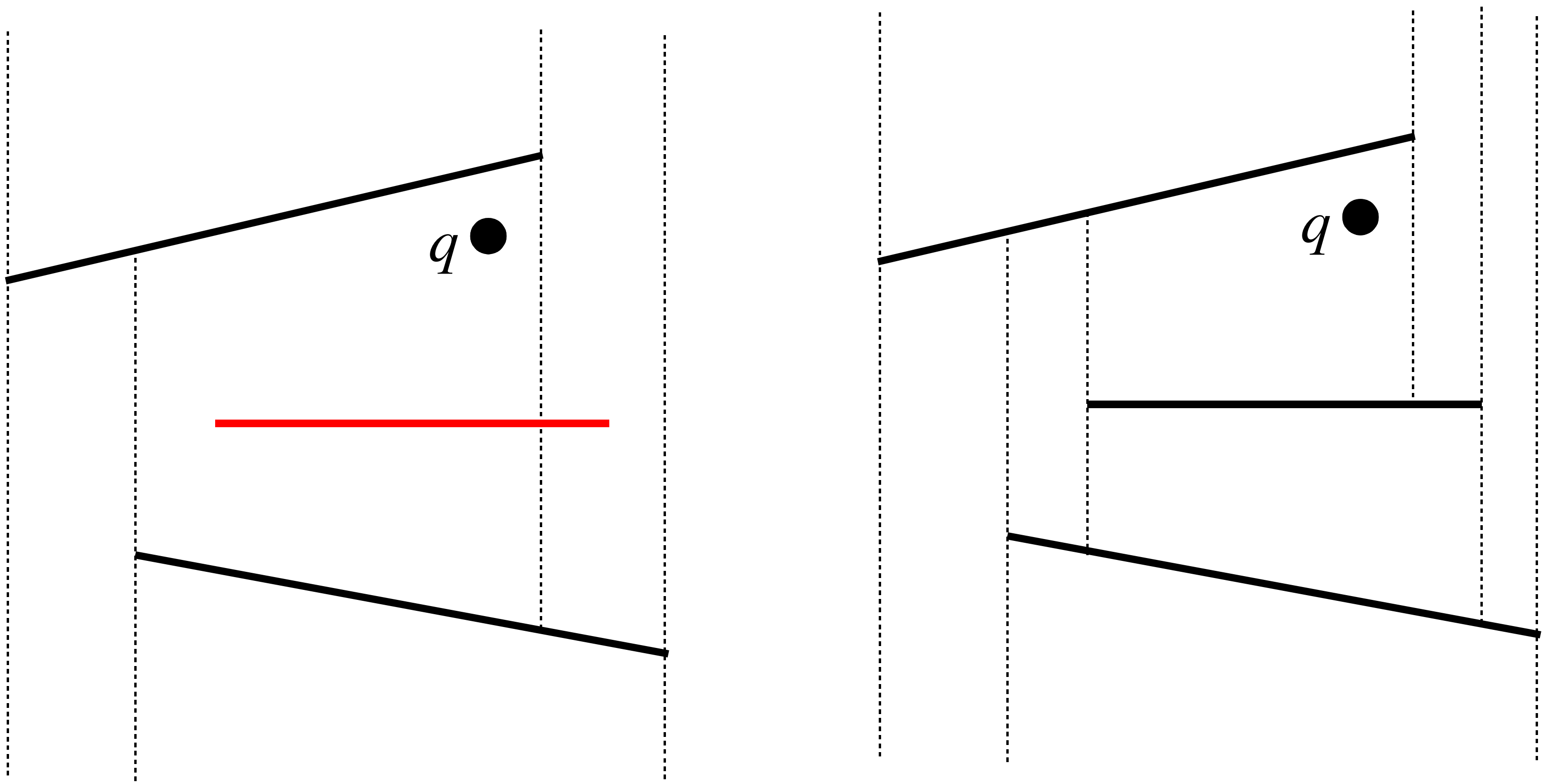
2. Fall: $t_i \neq t_{i-1}$

Dann ist mindestens eine der Seiten von $t_i$ durch das Hinzunehmen von $s_i$ entstanden.

Die Wahrscheinlichkeit dafür ist $O\left(\frac{1}{i}\right)$.

# Kosten der Punktlokalisierung

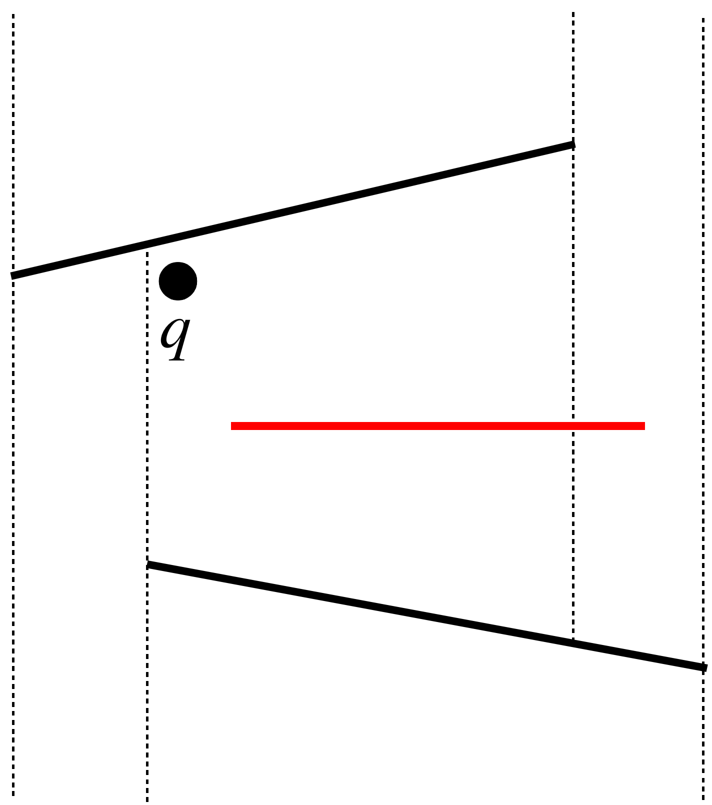
Beispiele zum 2. Fall:



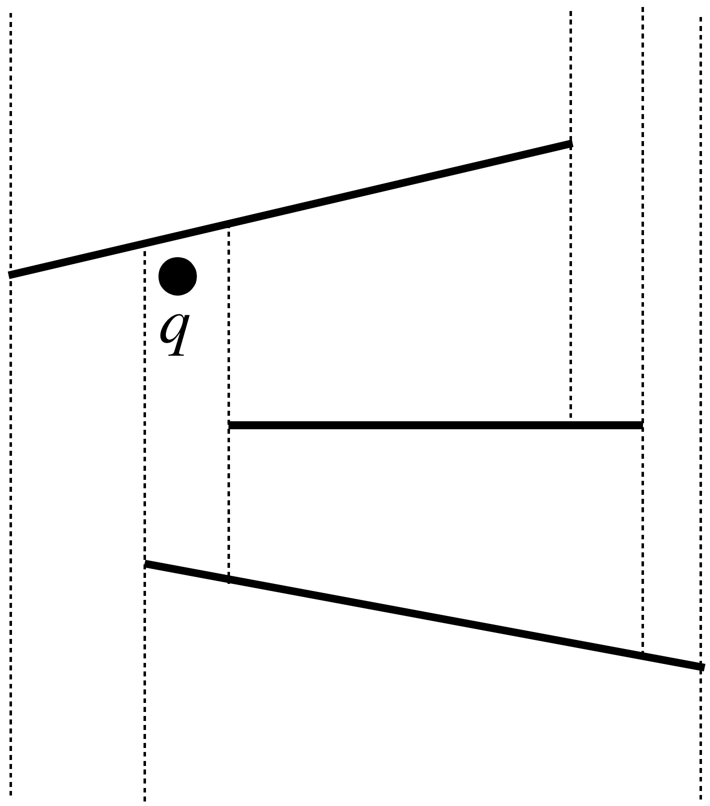

$T(P_{i-1})$ $T(P_i)$

# Kosten der Punktlokalisierung

Beispiele zum 2. Fall:



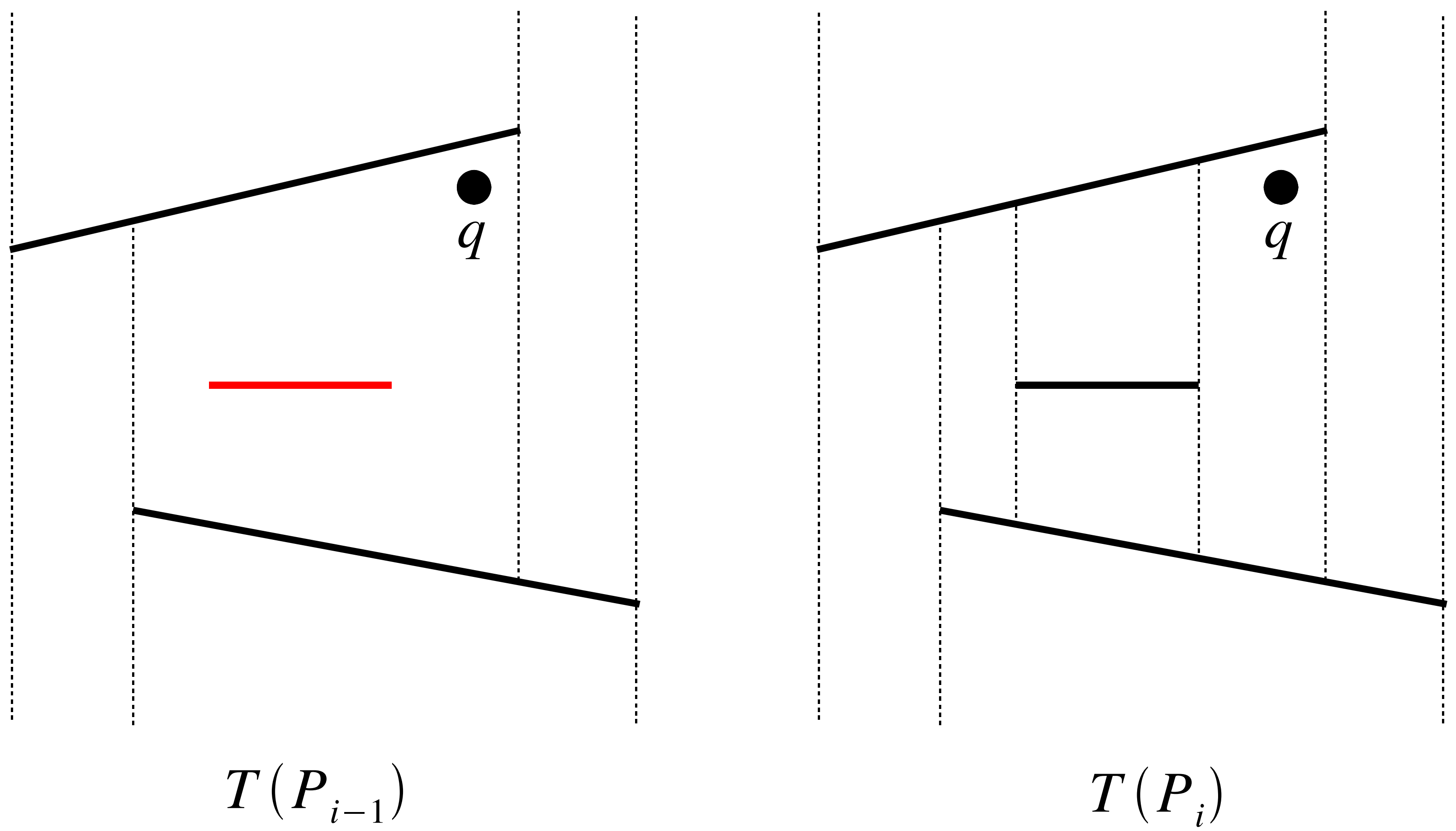

$T(P_{i-1})$




$T(P_i)$

# Kosten der Punktlokalisierung

Beispiele zum 2. Fall:



## Kosten der Punktlokalisierung

Wir erwarten also $O\left(\frac{1}{i}\right)$ weitere Vergleiche.

Aufsummieren: $\sum_{l=1}^{i} \frac{1}{l} = H_i \in O(\log i)$ ■

Gesamtlaufzeit: $\sum_{i=1}^{n} (1 + \log i) \in O(n + n \log n) = O(n \log n)$

## Beobachtung

Wenn wir für einen festen Punkt $q$ das Trapez $t_j$ kennen, dann erwarten wir $O\left(\log\frac{k}{j}\right)$ Vergleiche, um $t_k$ für $k \geq j$ zu finden.

Denn:

$$\sum_{l=j+1}^{k} \frac{1}{l} = H_k - H_j \in O\left(\log\frac{k}{j}\right)$$

# Idee

- Wir lassen den Algorithmus in Phasen ablaufen.
- Am Ende einer Phase lokalisieren wir jede Polygon-Ecke in der aktuellen Trapezzerlegung.
- Das beschleunigt die Punktlokalisierung in der nächsten Phase!

Punkte sind Ecken eines Polygons.
=> Lokalisierung mittels Wandern durch Trapezzerlegung.

# Bezeichnungen

$$\log^{(0)} n = n$$

$$\log^{(h)} n = \log(\log^{(h-1)} n) \text{ für } h > 0$$

$$N(h) = \lceil \frac{n}{\log^{(h)} n} \rceil$$

$$\log^* n = \max\{h \in \mathbb{N} \mid \log^{(h)} n \geq 1\}$$

# Algorithmus von Seidel (1991)

**1.** $s_1, \ldots, s_n$ zufällige Permutation der Strecken in $P$

**2.** $T(\{s_1\})$ berechnen

**3.** Für $h = 1$ bis $\log^* n$: { Phase $h$ }

**3.1.** Für $i = N(h-1)+1$ bis $N(h)$:
Füge $s_i$ in $T(P_{i-1})$ ein.

**3.2.** Verfolge Polygonzug durch $T(P_{N(h)})$
und lokalisiere alle Ecken.

**4.** Für $i = N(\log^* n)+1$ bis $n$:
Füge $s_i$ in $T(P_{i-1})$ ein.

# Analyse

**1.** $O(n)$

**2.** $O(1)$

**3.1.** $$\sum_{i=N(h-1)+1}^{N(h)} \log\left(\frac{i}{N(h-1)}\right) \le \sum_{i=N(h-1)+1}^{N(h)} \log\left(\frac{n}{\left\lceil \frac{n}{\log^{(h-1)} n} \right\rceil}\right)$$

$$\le \sum_{i=N(h-1)+1}^{N(h)} \log\left(\frac{n}{\frac{n}{\log^{(h-1)} n}}\right)$$

$$\le N(h) \log^{(h)} n \in O(n)$$

# Analyse

$$\text{Es gilt: } 1 \leq \log^{\log^* n} n < 2$$

$$\text{Damit: } N(\log^* n) = \left\lceil \frac{n}{\log^{\log^* n} n} \right\rceil \geq \left\lceil \frac{n}{2} \right\rceil \geq \frac{n}{2}$$

**4.** $$\sum_{i=N(\log^* n)+1}^{n} \log\left(\frac{i}{N(\log^* n)}\right) \leq \sum_{i=1}^{n} \log\left(\frac{2i}{n}\right)$$

$$\leq n \log 2 \in O(n)$$

Übrig bleibt Schritt **3.2.**

## Analyse

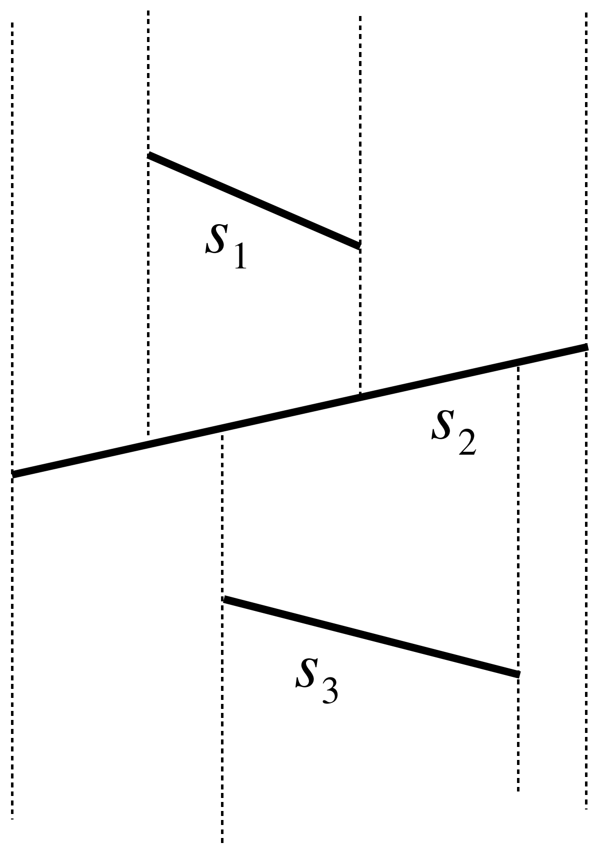
Lemma 2

Sei $R$ eine zufällig gewählte $r$-Teilmenge von $P$. Dann ist der Erwartungswert der Anzahl $X$ von Schnittpunkten zwischen Strahlen in der Trapezzerlegung $T(R)$ von $R$ und Strecken in $P \setminus R$ in $O(n)$.

Beweis:

Für eine Teilmenge $M$ von $P$ und eine Strecke $s$ aus $M$ sei $deg(s,T(M))$ die Zahl der auf $s$ treffenden Strahlen in $T(M)$.

# Analyse




$$M = \{s_1, s_2, s_3\}$$

$$deg(s_2, T(M)) = 4$$

$$deg(s_1, T(M)) = 0$$

## Analyse

Für jede Teilmenge $M$ von $P$ gilt: $\sum_{s \in M} deg(s, T(M)) \leq 4|M|$

$$E[X] = \sum_{\substack{R \subseteq P \\ |R| = r}} \frac{1}{\binom{n}{r}} \sum_{s \in P \setminus R} deg(s, T(R \cup \{s\}))$$

$$= \frac{1}{\binom{n}{r}} \sum_{\substack{R' \subseteq P \\ |R'| = r+1}} \sum_{s \in R'} deg(s, T(R'))$$

$$\leq \frac{1}{\binom{n}{r}} \sum_{\substack{R' \subseteq P \\ |R'| = r+1}} 4(r+1) \quad = 4(r+1) \frac{\binom{n}{r+1}}{\binom{n}{r}} = 4(n-r) \in O(n)$$

# Analyse

Zusammenfassung:

Pro Phase: $O(n)$

Anzahl Phasen: $O(\log^* n)$

Laufzeit: $O(n \log^* n)$

## Bemerkungen

$O(n \log^* n)$ erreicht man auch mit einem Teile-und-Herrsche Algorithmus nach Clarkson, Cole und Tarjan ('91).

$O(n)$ erreichte zuerst ein deterministischer Algorithmus von Chazelle ('90). Dieser ist aber sehr komplex und somit eher von theoretischem Interesse.

$O(n)$ erreicht auch ein weniger komplexer randomisierter Algorithmus nach Goodrich, Amato und Ramos ('00).