Union-Find – Analyse

Theorem

In einer anfangs leeren Union-Find–Datenstruktur mit Rangheuristik werden m Operationen in $O(m \log m)$ Zeit ausgeführt.

Beweis.

- Es gibt stets höchstens m Elemente
- Die Höhe aller Bäume ist durch $\log(m) + 1$ beschränkt
- Union und Find benötigt also nur $O(\log m)$ Zeit

$\square$

Rang und Pfadkompression

Mittels amortisierter Analyse (Tarjan 1975): m Operationen in $O(m\alpha(m))$ mit $\alpha(m)$ funktionale Inverse der Ackermannfunktion

Tarjan 1979, Fredman, Saks 1989: Das ist optimal!

Beweis recht kompliziert…

Übersicht

Algorithmische Geometrie

Probleme der Algorithmischen Geometrie haben üblicherweise diese Eigenschaften:

- Die Eingabe besteht aus Punkten, Segmenten, Kreisbögen usw. in der euklidischen Ebene.
- Die Fragestellung ist relativ einfach.
- Sehr große Eingaben müssen bewältigt werden.

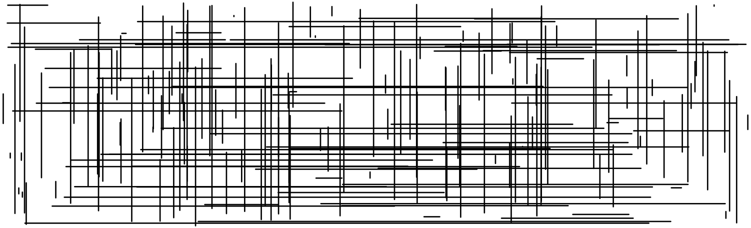
Anwendungen beispielsweise im VLSI-Design.

Übersicht

Schnitte von Segmenten

Eingabe: Horizontale und vertikale Segmente

Ausgabe: Paare von Segmenten, die sich schneiden



Naiver Algorithmus:

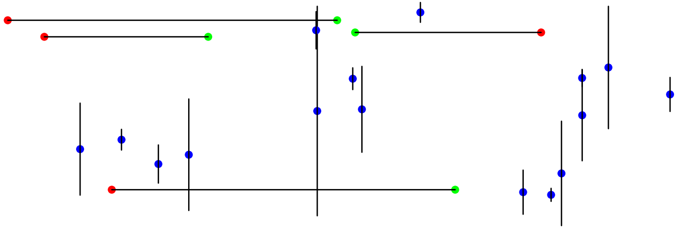
Teste alle Paare, ob sie sich schneiden.

Laufzeit: $\Theta(n^2)$

Variante: Finde heraus, ob es einen Schnitt gibt.

Übersicht

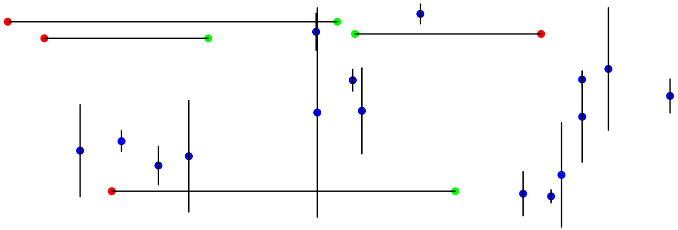
Sweepline-Algorithmen



- Interessante Punkte: Nach x-Koordinate sortieren
- Es gibt ein aktive Menge von Segmenten
- Eine imaginäre vertikale Linie bewegt sich von links nach rechts
- Roter Punkt: Segment in aktive Menge aufnehmen
- Grüner Punkt: Segment aus aktiver Menge entfernen
- Blauer Punkt: Segment mit aktiver Menge vergleichen

Suche nur Schnitte zwischen horizontalem und vertikalem Segment.

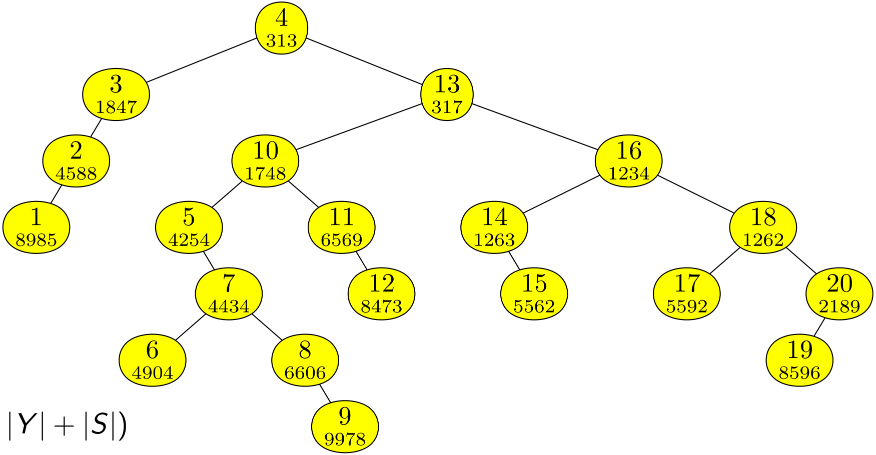
Sweepline-Algorithmen



- Wie finden wir bei einem vertikalen Segment die geschnittenen horizontalen Segmente?
- Welche Datenstruktur für die aktive Menge?

Lösung: Speichere y-Koordinaten Y der aktiven Menge in balanziertem Suchbaum.
Gibt es einen Schnitt? $\rightarrow$ $O(\log|Y|)$ Schritte

Alle Schnitte S ausgeben: $O(\log|Y| + |S|)$ Schritte
In aktive Menge einfügen oder löschen: $O(\log|Y|)$ Schritte



Aufgabe: Finde y-Koordinaten mit $a \leq y \leq b$.

Laufzeit: $O(\log |Y| + |S|)$

1. Finde größtes Element, daß kleiner oder gleich a ist.
2. Gehe von dort aus Element aufsteigend durch
3. Beende, wenn b überschritten.

Range-Search

Operationen:

1. Einfügen einer Zahl x
2. Löschen einer Zahl x
3. Ausgabe aller gespeicherter Zahlen in $[a, b]$

Welche Datenstrukturen sind geeignet?

- Arrays?
- Listen?
- AVL-Bäume?
- Splay-Bäume?
- $(2, 3)$-Bäume?
- Treaps?
- Skiplists?
- Hashtabellen?