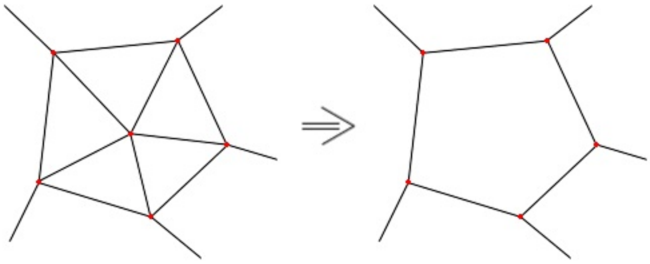
Algorithmische Geometrie - Vorlesung vom 11-05-2005

Stephan Scheerer, Wolfgang Sprenger

13. Mai 2005

Suchen in ebenen Unterteilungen
Beispiel: "Post-Office-Problem'"
Geg.: Menge S = $\{p_1,...,p_2\} \subset \mathbb{R}^2$
Ziel: Datenstruktur für S anlegen, so dass für einen Anfragepunkt q $\in \mathbb{R}^2$ effizient ein q am nächsten liegende $p_i \in$ S bestimmt werden kann.
allgemeiner:
Ebene Unterteilung ist Partition von $\mathbb{R}^2$ in durch Strecken oder Strahlen begrenzte Gebiete, also geometrischer Graph G. Finde Datenstruktur für G_0, so dass für Anfragepunkt q $\in \mathbb{R}^2$ effizient die Facette bestimmt werden kann, die q enthält.
("point location'")
Ressourcen:
1. Vorbereitungszeit zur Konstruktion der Datenstruktur aus G_0
2. Speicherplatzbedarf
3. Anfragezeit
zunächst: triangulierte ebene Unterteilungen
Lemma: Es gibt Konstanten 1 > c > 0, k $\in \mathbb{N}$, so dass für jede ebene Unterteilung G_0 mit n Knoten (einschliesslich ∞ entfernten Punkt und Annahme, dass dieser Punkt auch existiert). Gibt es eine unabhängige Kontenmenge I mit $\geq$ cn Elementen, in der jeder Knoten Grad $\leq$ k hat. I kann in Zeit O(n) gefunden werden.
Beweis: Da die Anzahl der Kanten $\leq$ 3n - 6 ist, ist die Summe der Knotengrade $\leq$ 6n - ~~12~~ (da jede Kante 2mal gezählt wird) : es gibt höchstens $\frac{n}{2}$ Knoten vom Grad $\geq$ 12
Nimm die anderen ($\geq \frac{n}{2}$, Grad $\leq$ 11) und bezeichne das als Menge K.
Nimm den ersten Knoten davon in I auf, entferne alle dazu adjazenten Knoten aus K (höchstens 11 entfernt),
nimm nächsten Knoten nach I auf usw.
für das entstehende I: $|I| \geq \frac{n}{24}$, das Lemma gilt für c = $\frac{1}{24}$, K = 11
Algorithmus zur Konstruktion einer Datenstruktur:
geg.: triangulierte ebene Unterteilung G_0, Suchstruktur: Baum, Blätter $\widehat{=}$ Dreiecke in G_0
① falls G_0 3 Knoten und 3 Kanten hat: Suchstruktur ist ein Baum mit 1 Wurzel F_0 ($\widehat{=}$ ganze Ebene) mit 2 Kindern ($\widehat{=}$ 2 Facetten)
② bestimme Menge I gemä ßdem Lemma
③ Elemente von I mit anhängenden Kanten entfernen



$\rightarrow$ ebene Unterteilung G_1 mit $\leq$ d*n Knoten wobei d = 1 - c

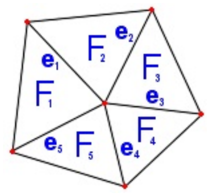
Facetten von G_1: m-Ecken mit m ≤ 11

④ trianguliere alle diese Facetten: triangulierte ebene Unterteilung G_2

⑤ konstruiere rekursiv eine Datenstruktur D_2 zum Suchen in G_2

⑥ daraus: Datenstruktur zum Suchen in G_1 durch Zusammenfassen entsprechender Dreiecke, die von einer Facette von G_1 her rühren.

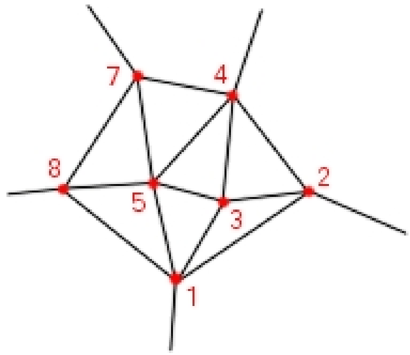
⑦ Daraus Datenstruktur D_0 (Suchstruktur) zum Suchen in G_0: betrachte alle Facetten F von G_1, falls ein Knoten x ∈ I in F liegt (dann genauer einer)



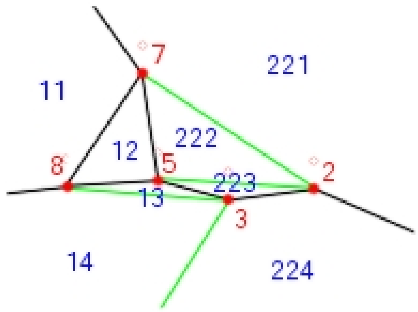
Kanten e_1,...,e_k inzident zu x in G_0 (k ≤ 11) in Suchstruktur neue Knoten F_1,...,F_k: Kinder von F

<u>Beispiel:</u>

I = 1,4 Zeile 2 / Schritt 2: G_0

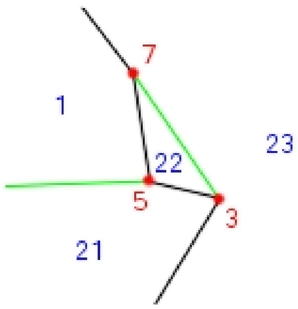


Schritt 3: G_1

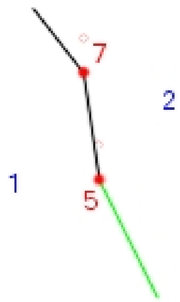


Schritt 4: Triangulierung : G_2

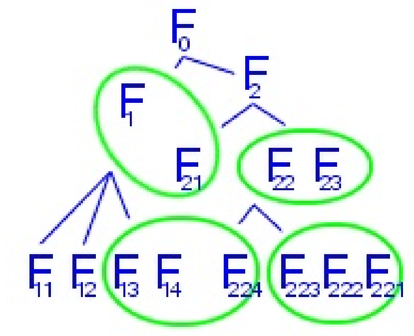
1. rek. Schritt I = 2,8



2. rek. Schritt I = 3



Datenstruktur:



Suche nach einem Punkt $q \in \mathbb{R}^2$, starte mit der Wurzel in einem Knoten v der Baumstruktur, vergleiche q mit allen Kindern von v (höchstens 11) und bestimme das, dessen zugeh. Dreieck q enthält, usw.

Laufzeit: O(Höhe der Baumstruktur)

Anzahl der Blätter der Dreiecke in G_0 = O(n)

in jedem Vergröberungsschritt v unten nach oben im Baum: Anteil $\subset$ der Kosten fallen raus, d = 1 - c bleiben $\frac{23}{24}$

also ist die Höhe des Baumes $\log_{\frac{23}{24}} n = O(\log_2 n)$