Stack

0,1
0,1,2
0,1,2,3
3,4
3,5
5,6
5,7
5,7,8
5,7,8,9

5,10
10,11

a → mit allen auf dem Stack verbinden
$S=(v_t, p_i)$

b → solange verbinden & poppen wie v_t höher
push p_i

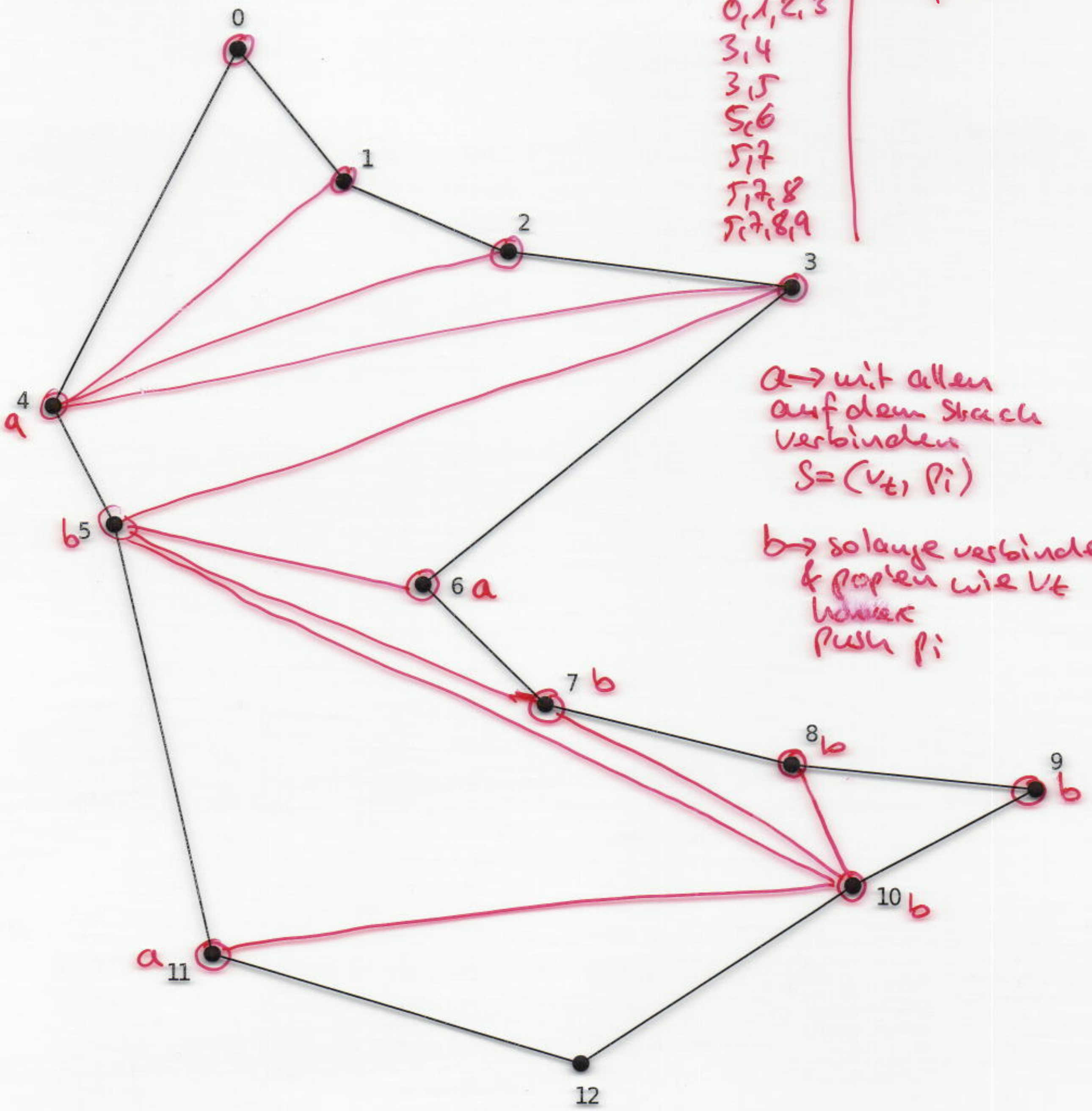


Abbildung 1: Monotones Polygon für Aufgabe 1. Die Zahlen geben die Nummerierung nach der Y-Koordinate an.

2) Ein Polygon heißt rechtwinklig, wenn alle Kanten entweder waagerecht oder senkrecht. Zeige: Für ein beliebiges $n \in \mathbb{N}$ gibt es ein rechtwinkliges Polygon mit n Ecken so dass mindestens $\lfloor \frac{n}{4} \rfloor$ Wächter nötig sind um das Polygon zu bewachen.

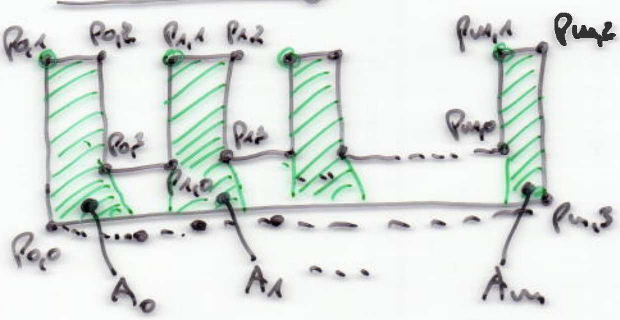
Ideen

a) 

Können 2 Wächter 3 Ecken sehen?
4-Zacken $= n$

b) Zerlege in 4-Ecke, pro 4-Eck 4 Farben analog zum bekannten Satz.

Musterlösung



„Kamm" mit m „Zacken" und $4m$ Ecken.

Für jedes $i \in \{1, \ldots, m\}$ gilt:
$P_{i,1}$ ist nur von einem Wächter im Polygon A_i sichtbar.

Da die Polygone $A_0 \ldots A_m$ paarweise überlappungsfrei sind, werden m Wächter benötigt um $P_{0,1} \ldots P_{m,1}$ abzudecken.

Mit n Ecken lässt sich immer solch ein Kamm mit $m = \lfloor \frac{n}{4} \rfloor$ „Zacken" bilden (zusätzliche Knoten irgendwo hinfügen, wo sie nicht „stören")

3) Gib einen Algorithmus an, der in Zeit $O(n \log n)$ ein Polygon mit Löchern trianguliert

Ideen

- Alg aus der VL
  - Monotone gehen in $O(n)$
  - Zerlegung in $O(n \log n)$
  - Löcher entfernen: $O(n) = O(n)$

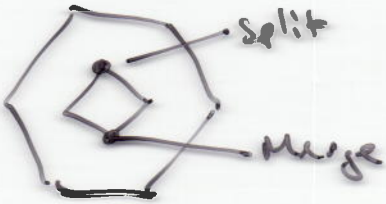
  $\Rightarrow O(n \log n)$

Musterlösung

1. Zerlege das Polygon mit dem Algo aus der VL in monotone Teilpolygone
2. Trianguliere diese Teilpolygone (nach VL)

Korrektheit

1. Der Plane-Sweep-Algo ist korrekt für einfache Polygone.
   Jedes Loch, das vom Algorithmus verarbeitet wird, muss mindestens einen Split- und einen Merge-Punkt haben



→ Jedes Loch wird mit je einer Diagonalen nach oben und nach unten verbunden

→ Das oberste Loch ist oben mit dem Außenrand verbunden, das untere unten auch.

⇒ Die entstehenden Teilpolygone sind einfach

Der VL-Algo garantiert, dass alle Sattelpunkte oberhalb der Sweep-Line durch Diagonalen eliminiert sind.
Das muss auch für Löcher gelten, da der Algo keine Informationen von unterhalb der Sweep-Line berücksichtigt.

Zwischenfrage:

2. (Triangulierung) korrekt nach VL

<u>Laufzeit</u>

1. – Sortieren nach Y-Koordinate
   $O(n \log n)$
   – Abarbeiten der Knoten:
   $O(n)$ Schritte, die jeweils (nach mail) $O(\log n)$ Zeit brauchen
   $\Rightarrow O(n \log n)$

Diese Zerlegung liefert „schlimmstenfalls" eine Triangulation $\Rightarrow$ Insgesamt höchstens $O(n)$ Ecken (siehe Aufgabe 4).

2. Die eigentliche Triangulierung benötigt $O(n)$ Schritte.

$\Rightarrow O(n \log n) + O(n) = O(n \log n)$ □

4) <u>Ideen</u>

worst-case: Triang, $n-2$ Teilpolygone, je 3-Ecken
$3n-6 \in O(n)$

Frage: Sind weniger Flächen evtl. schlimmer?

$\rightarrow$ nein, jede Diagonale erzeugt 2 Ecken

<u>Musterlösung</u>

Sei d die Anzahl der in P eingefügten Diagonalen. Dann gilt: Die Gesamtzahl der Ecken von Ps Teilpolygonen ist $n+2d$.

<u>Beweis</u>: Induktion über d

Start: $d=1$

Ecken an Diag. werden doppelt gezählt, alle anderen einfach.

Schritt: Beh. gelte d, zeige für $d+1$.

Sei P mit $d+1$ Diags zerteilt. Entferne eine, dann gibt es insgesamt laut Vor. $n+2d$ Ecken. Wenn jetzt die Diag wieder eingefügt wird, wird sie in ein Teilpolygon eingefügt.

Also muss es $n+2d+2 = n+2(d+1)$ Ecken geben. □

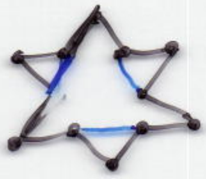
Die größtmögliche Zahl von Diagonalen ist $n+3h-3$ (Triangulierung).

$\Rightarrow$ max. $n+2(n+3h-3)$ Ecken $\in O(n+h) = O(n)$ □

4 Partitionierung in konvexe Teilpolygone

Problem: Geg. ein Polygon P, teile dieses durch Einfügen von Knoten und Strecken auf in möglichst wenige konvexe Teilpolygone.

Wie gut kann das gehen?

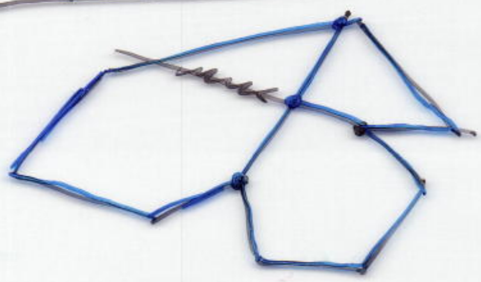


- Alle konkaven Ecken müssen benutzt werden
- Eine Diagonale deckt höchstens 2 konkave Ecken ab.

⇒ Bei r konkaven Ecken muss man mindestens in $\lceil \frac{r}{2} \rceil + 1$ Flächen zerlegen

$OPT \geq \lceil \frac{r}{2} \rceil + 1$

Einfacher Ansatz



- Winkelhalbierende durch konkave Ecken
- Schnittpunkt mit nächster Strecke oder Rand ist Endpunkt.

Laufzeit: $O(n^2)$

Chazelle: $\underset{\text{höchstens}}{O}(n + r^2 \log(\frac{n}{r}))$

Flächen: $r+1 \leq 2OPT$

(r = Anzahl der konkaven Ecken)

Außerdem

- Triangulierung + Diagonalen wegnehmen
  - → $O(n \log n)$ Zeit
  - → $\leq 4OPT$ Flächen
- Optimaler Partitionierungsalgo von Chazelle (1980) (Beschreibung umfasst 97 Seiten)