

Algorithmische Geometrie

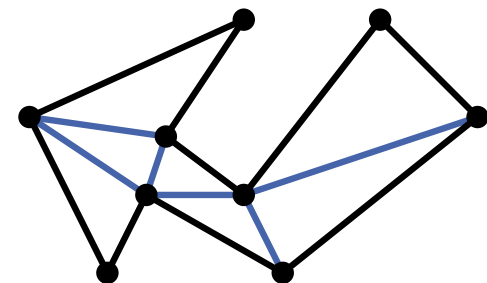
Triangulierung eines Polygons



Triangulierung von Polygonen

Definition

Eine **Triangulierung** eines Polygons P ist eine planare Unterteilung von P , sodass jede Facette ein Dreieck ist.



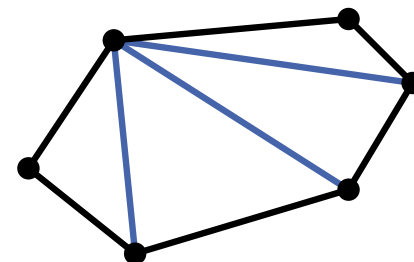
Problem

Gegeben P , finde Diagonalen, die P triangulieren.

Gibt es das immer?

Geht es etwas einfacher?

- konvexe Polygone sind leicht zu triangulieren
- Idee: zerlege P in konvexe Polygone
- trianguliere dann die konvexen Teilpolygone
- Problem: eine solche Zerlegung zu finden ist leider nicht viel leichter



Plan im Folgenden

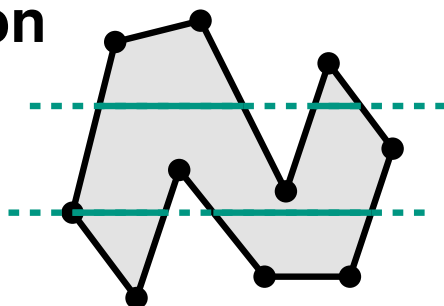
- finde eine schwächer Eigenschaft als Konvexität
- Triangulieren der Teilpolygone mit dieser Eigenschaft wird schwerer
- Zerlegung von P in Teilpolygone mit der Eigenschaft wird leichter

y-monotone Polygone

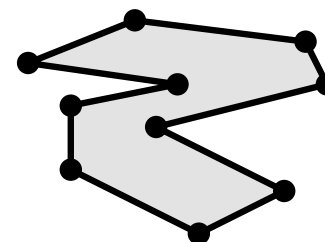
Definition

Ein Polygon ist **y-monoton**, wenn der Schnitt mit jeder horizontalen Geraden zusammenhängend ist.

nicht y-monoton



y-monoton



Anmerkung

- konvexe Polygone sind monoton in jede Richtung

x- und y-monoton $\Rightarrow$ konvex?

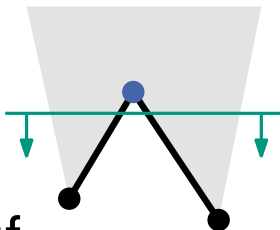
Unser Plan

- zerlege beliebiges Polygon in $O(n \log n)$ Zeit in y-monotone Teilpolygone
→ jetzt gleich
- trianguliere ein y-monotones Polygon in $O(n)$ Zeit
→ Übungsblatt

Was macht ein Polygon nicht y -monoton?

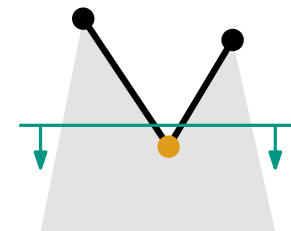
Split-Knoten

- Kanten liegen unten
- Polygon liegt oben
- Polygon spaltet sich auf (von oben kommend)



Merge-Knoten

- Kanten liegen oben
- Polygon liegt unten
- Polygone fügen sich zusammen



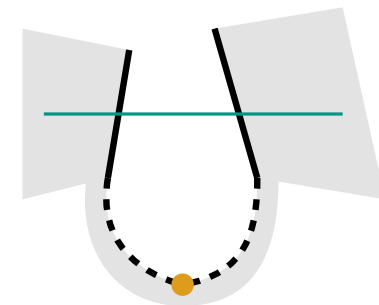
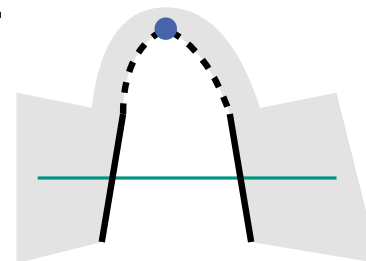
Beobachtung

- es gibt Merge- oder Split-Knoten $\Rightarrow$ das Polygon ist nicht y -monoton
- die Umkehrung gilt auch; Beweis durch Bild:

Lemma

(y -Monotonie)

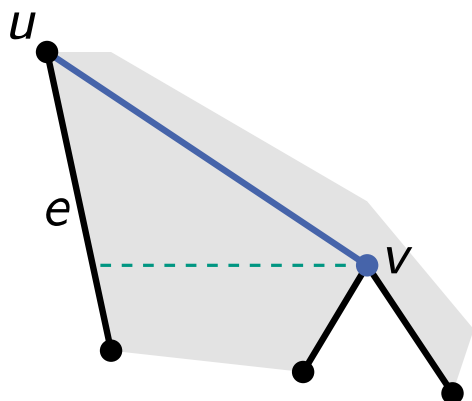
Ein Polygon ist genau dann y -monoton, wenn es keine Split- oder Merge-Knoten gibt.



Ziel

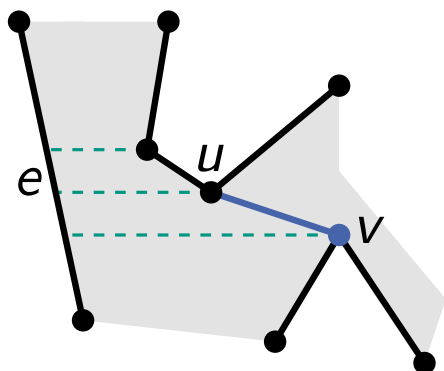
- eliminiere alle Split- und Merge-Knoten
- füge dazu Diagonalen ein
- von Split-Knoten nach oben, von Merge-Knoten nach unten

Eliminierung von Split-Knoten



Idee für Split-Knoten v

- finde Knoten u oberhalb von v
- wenn u nah bei v liegt, dann schneidet die Diagonale uv das Polygon hoffentlich nicht
- finde nächste Kante e links von v
- wähle oberen Endpunkt von e als u



Problem

- uv könnte andere Kanten des Polygons schneiden
- wähle für u stattdessen den tiefsten Knoten über v , der e links neben sich hat
- diesen Knoten nennen wir den **Helfer** von e
- **Beachte:** Helfer hängt von v ab

Lemma

(der Helfer hilft)

Sei v ein Split-Knoten, e die Kante links neben v und u der Helfer von e (bzgl. v). Dann schneidet uv keine Kante des Polygons (außer in u und v).

Beweis: ■ das Viereck zwischen uv und e enthält keine Knoten
 ■ keine Polygonkante kann uv schneiden

Warum?

Warum?

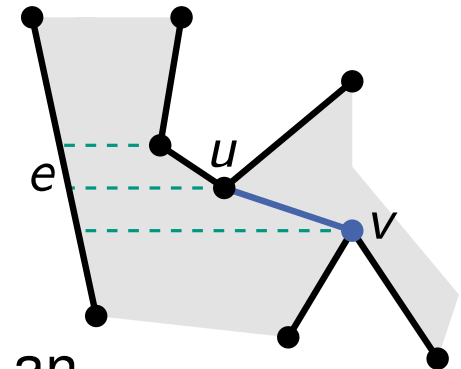
Eliminierung von Split-Knoten

Was müssen wir für einen Split-Knoten v tun?

- finde die Kante e links von v
- finde den Helfer von e (bezüglich v)

Beobachtung

- e liegt (teilweise) über v
 - der Helfer von e liegt über v
- } $\Rightarrow$ Sweep-Line bietet sich an
(horizontale Sweep-Line ℓ von oben nach unten)



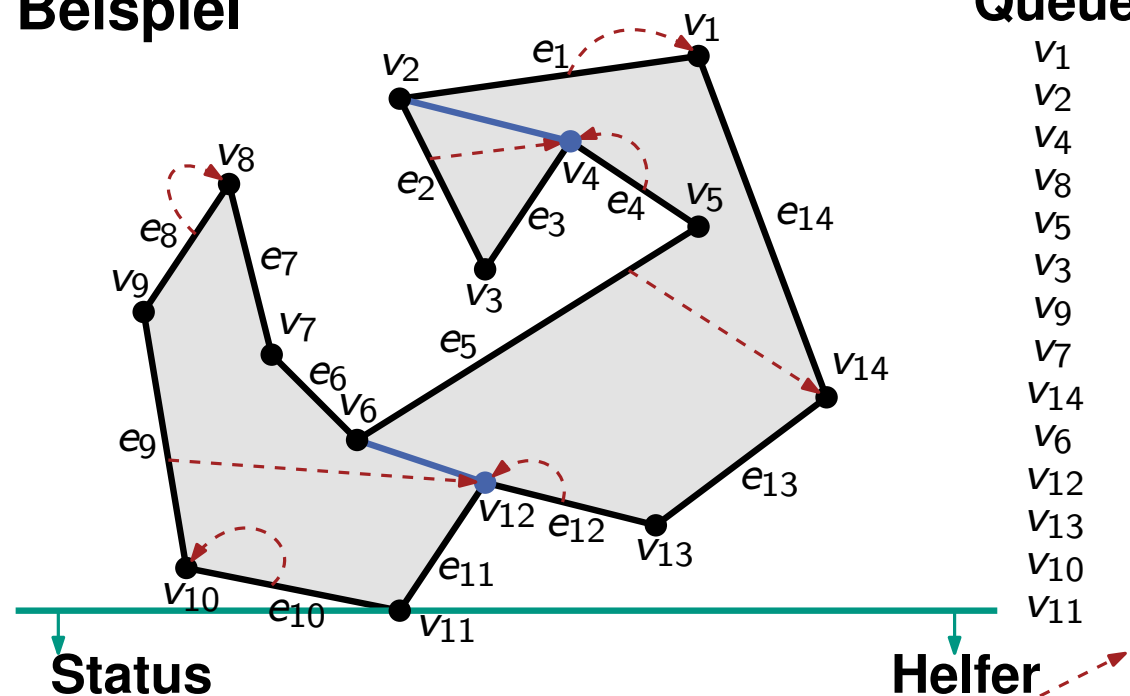
Event-Queue

- Knoten des Polygons
- sortiert nach y -Koordinate
(bzw. lexikographisch nach yx)

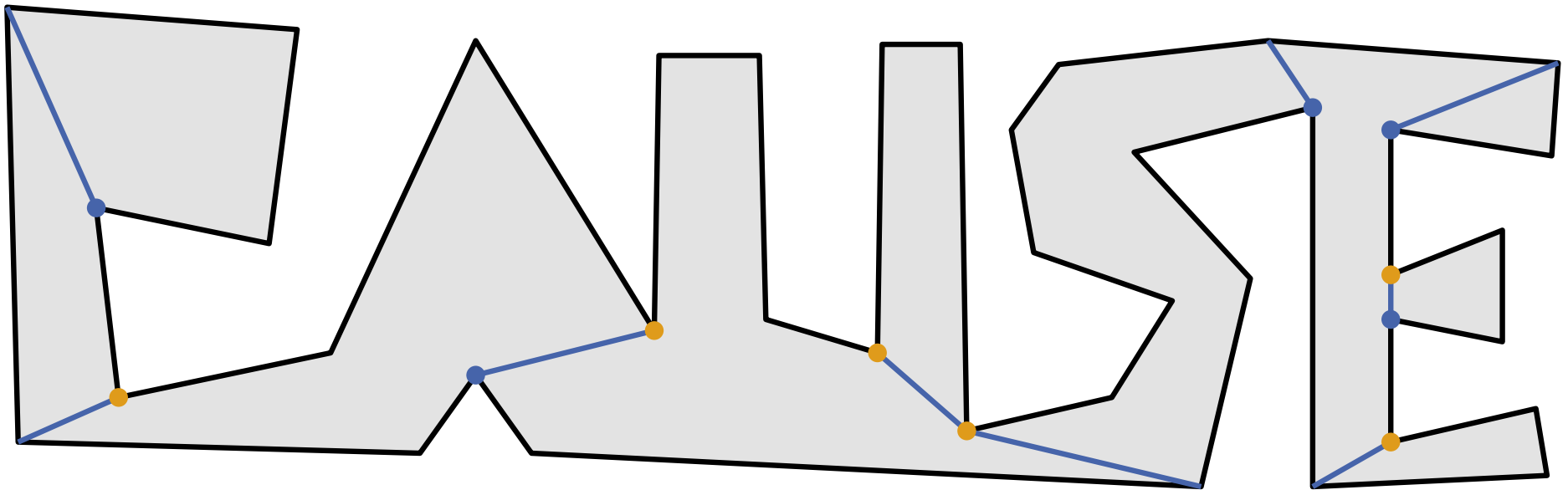
Sweep-Line Status

- Kanten die ℓ schneiden sortiert nach x -Koordinate
- Kanten mit Polygon rechts von sich genügen
- aktueller Helfer für jede Kante

Beispiel



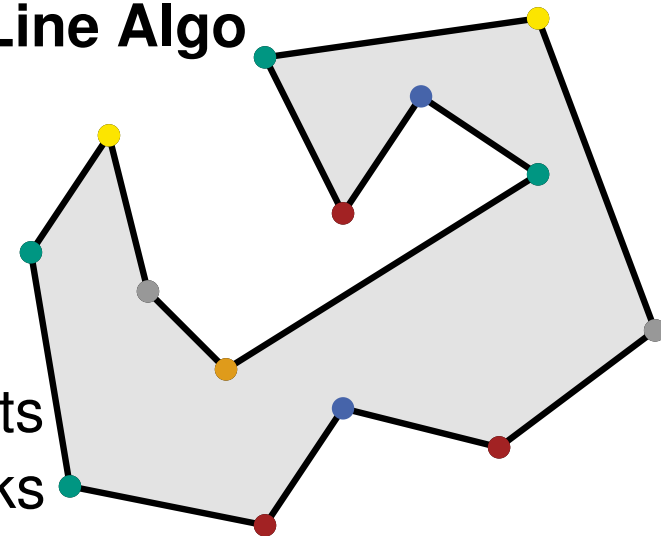
**Wie viele Diagonalen muss man mindestens einfügen,
um y -monotone Polygone zu erhalten?**



Sweep-Line mit Verschiedene Knotentypen

Unterschiedlich behandelte Knoten im Sweep-Line Algo

- **Split-Knoten:** Kanten unten, Polygon oben
- **Merge-Knoten:** Kanten oben, Polygon unten
- **Start-Knoten:** Kanten unten, Polygon unten
- **End-Knoten:** Kanten oben, Polygon oben
- **linker Knoten:** y -monotone Kanten, Polygon rechts
- **rechter Knoten:** y -monotone Kanten, Polygon links



Sweep-Line mit Verschiedene Knotentypen

Algorithmus MACHEMONOTON(P)

Eingabe. Polygon P (gegen den Uhrzeigersinn)

Ausgabe. Diagonalen, die P y -monoton machen

Q = Knoten von P sortiert nach y -Koordinate

T = binärer Suchbaum

solange $Q \neq \emptyset$

$v = \min\{Q\}$ und $Q = Q - v$

BEHANDLEKNOTEN(v)

BEHANDLESTARTKNOTEN(v)

e_v^+ = Kante nach v in P

füge e_v^+ in T ein

helfer(e_v^+) = v

BEHANDLEENDKNOTEN(v)

e_v^- = Kante vor v in P

entferne e_v^- aus T

BEHANDLERECHTENKNOTEN(v)

e = Kante links von v in T

helfer(e) = v

BEHANDLELINKENKNOTEN(v)

e_v^- = Kante vor v in P

entferne e_v^- aus T

e_v^+ = Kante nach v in P

füge e_v^+ in T ein

helfer(e_v^+) = v

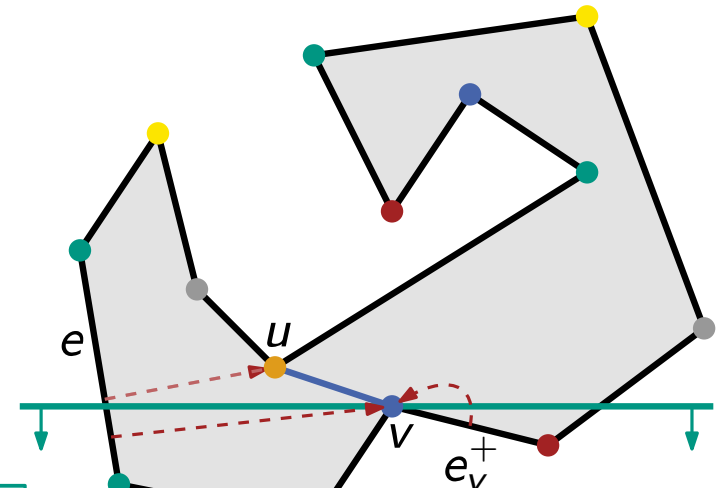
BEHANDLEMERGEKNOTEN(v)

e = Kante links von v in T

helfer(e) = v

e_v^- = Kante vor v in P

entferne e_v^- aus T



BEHANDLESPLITKNOTEN(v)

e = Kante links von v in T
 u = helfer(e)

gib Diagonale uv **aus**

helfer(e) = v

e_v^+ = Kante nach v in P

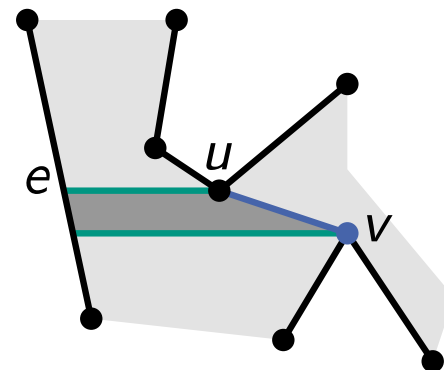
füge e_v^+ in T ein

helfer(e_v^+) = v

Schnittfreiheit

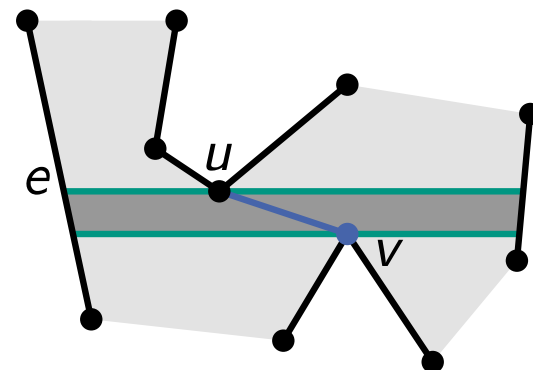
Erinnerung

- die eingefügten Diagonalen schneiden das Polygon nicht
- Argument: das Viereck zwischen uv und e enthält keine Knoten

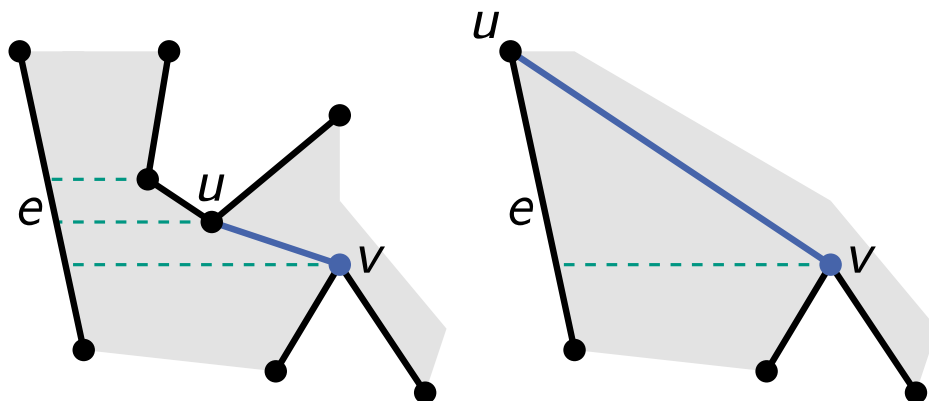


Kann eine zuvor eingefügte Diagonale geschnitten werden?

- Verlängerung des Vierecks nach rechts enthält auch keinen Knoten (gleiches Argument)
 - beide Endpunkte einer zuvor eingefügten Diagonale liegen über v
- $\Rightarrow uv$ schneidet keine vorherige Diagonale



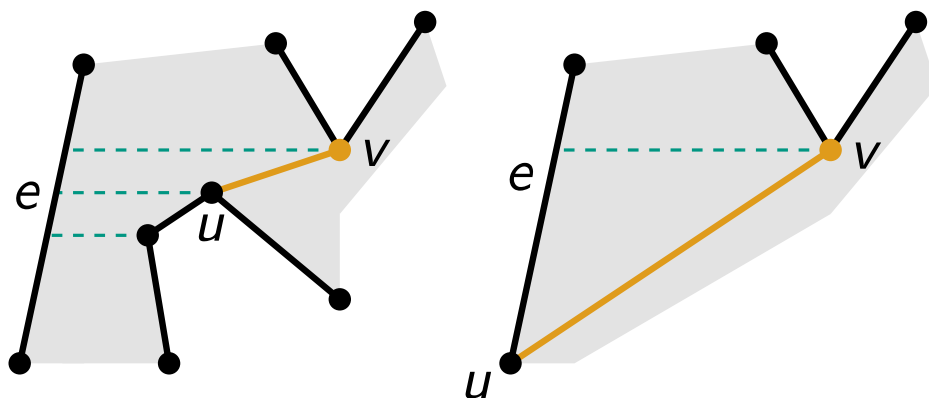
Und was ist mit Merge-Knoten?



Erinnerung: Split-Knoten v

- e : Kante links neben v
- wähle für u tiefsten Knoten über v , der e links von sich hat (Helfer von e)
- falls dieser nicht existiert: wähle für u oberen Knoten von e
- verbinde v mit u

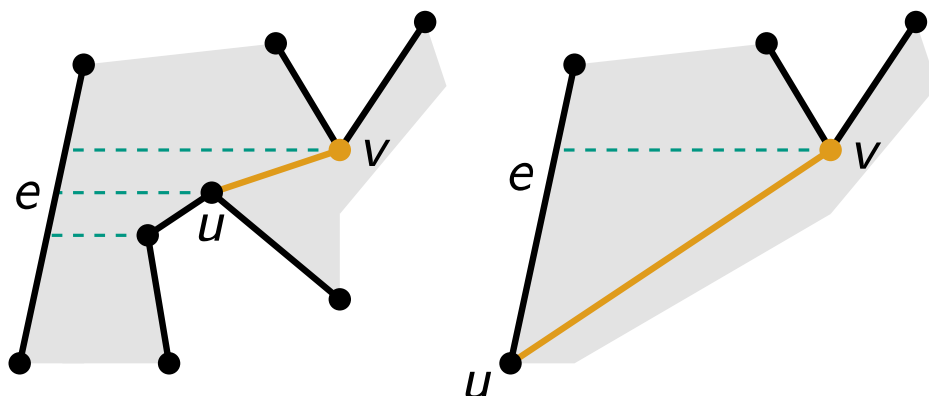
----- Spiegeln macht aus v einen Merge-Knoten



Behandlung von Merge-Knoten v

- e : Kante links neben v
- wähle für u den höchsten Knoten unter v , der e links von sich hat
- falls dieser nicht existiert: wähle für u unteren Knoten von e
- verbinde v mit u

Und was ist mit Merge-Knoten?



Behandlung von Merge-Knoten v

- e : Kante links neben v
- wähle für u höchsten Knoten unter v , der e links von sich hat
- falls dieser nicht existiert: wähle für u unteren Knoten von e
- verbinde v mit u

Problem

- die relevanten Infos liegen unterhalb von v
- Sweep-Line Algo kann Diagonale nicht bei Abarbeitung von v einfügen

Beobachtung

- v wird zunächst der Helfer von e
- u ist der nächste Helfer von e oder der Endpunkt von e
- Plan: eliminiere Merge-Knoten v erst bei Abarbeitung von u

Ersetze

e = Kante links von u in T
 $\text{helfer}(e) = u$

e_u^- = Kante vor u in P
entferne e_u^- aus T

Durch

e = Kante links von u in T
 $v = \text{helfer}(e)$
wenn v ist Merge-Knoten
gib Diagonale uv **aus**
 $\text{helfer}(e) = u$

e_u^- = Kante vor u in P
 $v = \text{helfer}(e)$
wenn v ist Merge-Knoten
gib Diagonale uv **aus**
entferne e_u^- aus T

Zusammenfassung

Theorem

(Zerlegung in y -monotone Teile)

Ein gegebenes Polygon mit n Knoten kann in $O(n \log n)$ Zeit in y -monotone Teilpolygone zerlegt werden.

Außerdem gesehen

- weitere Anwendung der Sweep-Line Technik
- Konzept der Monotonie
- Aufteilung eines komplizierten Problems in zwei leichtere Teilprobleme

Was gibt es sonst noch?

- untere Schranke von $\Omega(n \log n)$ wenn das Polygon Löcher haben darf
- $O(n \log \log n)$, $O(n \log^* n)$, und sogar $O(n)$ möglich, wenn das Polygon keine Löcher hat
- entsprechendes 3-dimensionales Problem ist NP-schwer

Literaturhinweise

$O(n \log \log n)$

- **An $O(n \log \log n)$ -Time Algorithm for Triangulating a Simple Polygon** (1988)
Robert E. Tarjan, Christopher J. Van Wyk <https://doi.org/10.1137/0217010>
- **Polygon triangulation in $O(n \log \log n)$ time with simple data structures** (1992)
David G. Kirkpatrick, Maria M. Klawe, Robert E. Tarjan <https://doi.org/10.1007/BF02187846>

$O(n \log^* n)$

- **A fast Las Vegas algorithm for triangulating a simple polygon** (1989)
Kenneth L. Clarkson, Robert E. Tarjan, Christopher J. Van Wyk <https://doi.org/10.1007/BF02187741>
- **Randomization Yields Simple $O(n \log^* n)$ Algorithms for Difficult $\Omega(n)$ Problems**
Olivier Devillers <https://doi.org/10.1142/S021819599200007X> (1992)
- **A simple and fast incremental randomized algorithm for computing trapezoidal decompositions and for triangulating polygons** (1991)
Raimund Seidel [https://doi.org/10.1016/0925-7721\(91\)90012-4](https://doi.org/10.1016/0925-7721(91)90012-4)

$O(n)$

- **Triangulating a simple polygon in linear time** (1991)
Bernard Chazelle <https://doi.org/10.1007/BF02574703>
- **A Randomized Algorithm for Triangulating a Simple Polygon in Linear Time** (2001)
N. M. Amato, M. T. Goodrich, E. A. Ramos <https://doi.org/10.1007/s00454-001-0027-x>