Karlsruher Institut für Technologie
Fakultät für Informatik
ITI Wagner

Algorithmische Geometrie
Sommersemester 2014
Martin Nöllenburg/Benjamin Niedermann

Übungsblatt 1 - Konvexe Hüllen (Lösung)

Ausgabe: Mittwoch, 16. April 2014
Abgabe: Dienstag, 22. April 2014

1 Einfacher Algorithmus für konvexe Hülle

In der Vorlesung wurde der Algorithmus FirstConvexHull(P) vorgestellt, der für eine gegebene Punktmenge P die konvexe Hülle berechnet. Hierzu werden im letzten Schritt die berechneten Kanten in eine sortierte Knotenliste L von $CH(P)$ überführt. In der Vorlesung wurde gezeigt, dass dies in $O(n^2)$ Zeit möglich ist. Zeigen Sie, dass dieser Schritt sogar in $O(n \log n)$ Zeit berechnet werden kann.

Lösung: Siehe Folien der Übung vom 30.04.

2 Algorithmus Gift Wrapping

In der Vorlesung wurde der Algorithmus *Gift Wrapping* für die Berechnung der konvexen Hülle vorgestellt. Zeigen Sie folgenden Satz und gehen Sie dabei insbesondere auf die Korrektheit des Algorithmus ein.

Satz 1. *Die konvexe Hülle $CH(P)$ von n Punkten P in $\mathbb{R}^2$ lässt sich mit* Gift Wrapping *(auch* Jarvis' March*) in $O(n \cdot h)$ Zeit berechnen, wobei $h = |CH(P)|$.*

Welche degenerierten Fälle können auftreten? Wie kann man mit diesen Fällen umgehen?

Lösung:

Sei T die Ausgabe des Algorithmus.

Angenommen es existiert ein Punkt $p[\ell]$ in T, der nicht zur konvexen Hülle $CH(P)$ gehört. Da er zu T aufgenommen wurde, gibt es keinen Punkt links von der gerichteten Linie durch $p[\ell-1]$ und $p[\ell]$. Wenn $p[\ell-1]$ Teil der konvexen Hülle $CH(P)$ ist, dann muss auch $p[\ell]$ Teil der konvexen Hülle sein, da sonst dieser Punkt nicht im Inneren der konvexen Hülle wäre. Wir wissen, dass der rechteste Punkt in $CH(P)$ ist und dieser ist $p[1]$. Also kann es keinen Punkt $p[\ell]$ in T geben der nicht Teil von $CH(P)$ ist.

Angenommen es existiert ein Punkt der in $CH(P)$ aber nicht in T enthalten ist. Wir wissen, dass der rechteste Punkt aus P sowohl in $CH(P)$ als auch in T enthalten ist. Wir nehmen an, dass sowohl T als auch $CH(P)$ im Uhrzeigersinn sortiert sind. Sei dann p der erste Punkt welcher in $CH(P)$ aber nicht in T enthalten ist. Wir bezeichnen den letzten gemeinsamen Punkt von T

und $CH(P)$ vor p als $p[k]$. Der Punkt p kann nicht links von der gerichteten Linie durch $p[k]$ und $p[k+1]$ liegen, da sonst der Algorithmus den Punkt zu T hinzugefügt hätte. Also muss der Punkt rechts von dieser Linie liegen. Nun müssen zwei Fälle betrachtet werden:

- Wenn $CH(P)$ den Punkt $p[k+1]$ nicht enthält, dann ist $CH(P)$ nicht die konvexe Hülle, da der Punkt $p[k+1]$ außerhalb der konvexen Hülle liegen würde. Widerspruch!
- Wenn $CH(P)$ den Punkt $p[k+1]$ enthält, dann hat $CH(P)$ einen Rechtsknick, da der Punkt p rechts von der gerichteten Linie durch $p[k]$ und $p[k+1]$ liegt. Widerspruch!

Insgesamt folgt die Korrektheit des Verfahrens, d.h., $CH(P) = T$. Die Laufzeit ergibt sich direkt aus der Definition des Algorithmus.

Degenerierte Fälle:

1. Der rechteste Punkt ist nicht eindeutig.
   Lösung: Wähle den obersten Punkt.
2. In Zeile 7 des Algorithmus' (s. Algorithm 1) liegen mehrere Punkte auf der Linie durch $p[i]$ und NEXT.
   Lösung: Wähle aus diesen Punkt den Punkt mit größter Distanz.

3 Algorithmus Chan Hull

Gegeben sei ein konvexes Polygon P mit n Knoten und ein Punkt p außerhalb von P; siehe Abbildung 1.

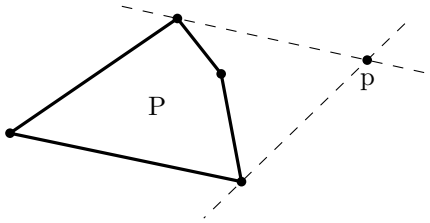

Abbildung 1: Illustration von Aufgabe 3

1. Wie kann eine Tangente an P, die durch p führt, in $O(\log n)$ Zeit berechnet werden, wenn die Ecken von P als (im Uhrzeigersinn) sortierte Liste gegeben sind?
2. An welcher Stelle im Algorithmus *Chan Hull* wird diese Berechnung benötigt?

Lösung: Siehe Folien der Übung vom 30.04.14.

4 Optimalität!

Von einem Algorithmus, der die konvexe Hülle einer gegebenen Punktmenge berechnet, fordern wir, dass er die Punkte als (im Uhrzeigersinn) sortierte Liste ausgibt.

a) Zeigen Sie, dass jeder Algorithmus zur Berechnung der konvexen Hülle von n Punkten im schlimmsten Fall eine Laufzeit von $\Omega(n \log n)$ hat, was bedeutet, dass *Graham Scan* optimal im Sinne der asymptotischen Laufzeit ist.
Hinweis: Benutzen Sie, dass die *Sortierung* von n Schlüsseln (in gewissen Rechnermodellen) eine Laufzeit von $\Omega(n \log n)$ benötigt.
b) Weshalb stellt die Laufzeit von *Grift Wrapping* keinen Widerspruch zum Ergebnis aus Teilaufgabe a) dar?
c) Gegeben sei ein einfaches, nicht notwendigerweise konvexes Polygon in der üblichen Listenrepräsentation. Geben Sie einen Algorithmus an, der die konvexe Hülle der Eckenmenge dieses Polygons in $\mathcal{O}(n)$ Zeit berechnet. Erläutern Sie, weshalb dies keinen Widerspruch zum Ergebnis aus Teilaufgabe a) darstellt.

Lösung: Zu a): Wandle die Eingabe Zahlen $E = \{e_1, \ldots, e_n\}$ in Punkte mit Koordinaten (e_1, e_1^2) um. Berechne von diesen Punkten die konvexe Hülle. Damit ist die Eingabe sortiert.

Zu b): Im schlimmsten Fall hat Gift-Wrapping die Laufzeit $O(n^2)$, da alle Punkte einer Menge in konvexer Lage liegen können.

Zu c): Siehe Folien der Übung vom 30.04.14.