Praktische Informatik 3 WS 10/11

Christoph Lüth
Bernd Gersdorf
Christian Maeder
René Wagner
Dennis Walter
Diedrich Wolter
Simon Ulbricht

7. Übungsblatt

Ausgabe: 06.01.11
Abgabe: 17.01.11

7.1 *Graham's Scan* — *10 Punkte*

Die Berechnung der *konvexen Hülle* einer Menge von Punkten (in zwei Dimensionen) ist eine Standardaufgabe der algorithmischen Geometrie, die in dieser Übung durch den als Graham's Scan bekannten Algorithmus gelöst werden soll.

Mathematisch versteht man unter der konvexen Hülle $\mathcal{C}(M)$ einer Punktmenge M die kleinste konvexe Obermenge von M. Eine Punktmenge M ist *konvex*, wenn für alle Punkte p und q in M auch die Verbindungslinie zwischen p und q in M liegt:

$$konvex(X) \Longleftrightarrow \forall p \in X.\, \forall q \in X.\, \{p + s \cdot (q - p) \mid 0 \leq s \leq 1\} \subseteq X \tag{1}$$

$$\mathcal{C}(M) = \bigcap \{N \mid M \subseteq N \wedge konvex(N)\}. \tag{2}$$

Die konvexe Hülle endlicher Mengen ist immer die Fläche eines konvexen Polygons, dessen Eckpunkte in der Menge enthalten sind.[1] Ein (nicht unbedingt konvexes) Polygon wird über die Punkte des seinen Rand definierenden Linienzuges repräsentiert. Punkte stellen wir als komplexe Zahlen in kartesischer Form dar. Dies motiviert die folgenden Datentypdefinitionen:

```
type Point = Complex Double
type Polygon = [Point]
```

Die Definitionen (1) und (2) sind formal elegant, eignen sich aber nicht gut als Grundlage für eine Implementierung; stattdessen machen wir uns die folgende Beobachtung zunutze: in einem konvexen Polygon mit n Eckpunkten gilt für drei benachbarte Punkte $p_{i-1 \bmod n}, p_i, p_{i+1 \bmod n}$ immer, dass $p_{i+1 \bmod n}$ *links* von der in $p_{i-1 \bmod n}$ beginnenden und durch p_i verlaufenden Halbgeraden liegt. Dies gilt allgemeiner für *alle* Punkte, die innerhalb des Polygons liegen. Ein Punkt c liegt genau dann links von a und b (wir schreiben hierfür $links(a, b, c)$), wenn gilt[2]

$$\langle b - a; (a - c)i \rangle \geq 0. \tag{3}$$

Hierbei bezeichnet $\langle x; y \rangle$ das Skalarprodukt von x und y, definiert als $\langle x; y \rangle = Re(x) * Re(y) + Im(x) * Im(y)$, wobei $Re(x)$ und $Im(x)$ der Real- bzw. Imaginärteil der komplexen Zahl x sind. i ist die imaginäre Einheit; eine Multiplikation mit i entspricht geometrisch einer Vektorrotation um 90° im mathematisch positiven Drehsinn.

1. Implementieren Sie zwei Funktionen auf Basis von (3), die bestimmen, ob ein Polygon konvex ist, und ob ein Punkt in einem konvexen Polygon liegt.

```
isConvex :: Polygon → Bool
isInside :: Point → Polygon → Bool
```

Graham's Scan zur Berechnung der konvexen Hülle einer endlichen Punktmenge M arbeitet wie folgt:

a. Finde einen Punkt p_{min}, der in jedem Fall ein Eckpunkt des Polygons ist. Dies trifft u.a. auf den Punkt mit der minimalen x-Koordinate zu (warum?)

b. Sortiere alle übrigen Punkte $M \backslash \{p_{min}\}$ aufsteigend bezüglich der Ordnung, bei der p genau dann kleiner ist als q, wenn $links(p_{min}, p, q)$ gilt. Die sortierte Punktliste sei $[p_1, p_2, \ldots, p_{n-1}]$.

[1] Anschaulich ist sofort klar, dass die "Randpunkte" der Punktmenge das Polygon definieren. Stellen wir uns die Punkte als Nägel auf einem Brett vor, dann erhalten wir die Randpunkte, indem wir ein Gummiband um alle Nägel herumlegen und dieses spannen. Die vom Band berührten Punkte sind Randpunkte.

[2] Siehe `http://en.wikipedia.org/wiki/Dot_product#Geometric_interpretation`

c. Die Hülle $H = \mathit{reverse}\ H_{n-2}$ wird iterativ aufgebaut. Starte mit der Liste $H_1 = [p_2, p_1, p_{min}]$. Betrachte in jeder Iteration $i \geq 1$ den Punkt p_{i+2} und entferne aus H_i alle vorderen Punkte, bis p_{i+2} links von den vorderen zwei Punkten p_j, p_k ($j > k$) liegt, d.h. bis $\mathit{links}(p_k, p_j, p_{i+2})$ gilt. Füge p_{i+2} dieser Liste vorn an und setze H_{i+1} auf die so entstandene Liste. Fahre mit Iteration $i + 1$ fort.

Diese Beschreibung ignoriert Sonderfälle wie die leere Punktmenge, Punktmengen, bei denen alle Punkte auf einer Geraden liegen, oder solche bei denen kein eindeutiger Punkt mit minimaler x-Koordinate existiert. Dokumentieren Sie das Verhalten Ihrer Implementierung diesbezüglich!

2. Implementieren Sie das oben beschriebene Verfahren in einer Funktion

```
convexHull :: Set Point → Polygon
```

Hinweis: Der Test Ihrer Implementierung erfolgt mittels der nachfolgenden Aufgabe; Sie müssen für diese Teilaufgabe keine zusätzlichen Testfälle schreiben.

$\boxed{7.2}$ *Schnell mal prüfen* *10+2 Punkte*

Konvexe Mengen weisen viele strukturelle Eigenschaften auf, die sich ideal zum Testen der oben angegebenen Funktionen mithilfe von QuickCheck eignen. So gilt (auf mathematischer Ebene) folgendes:

$$\mathit{konvex}(\mathcal{C}(M)) \tag{4}$$
$$M \subseteq \mathcal{C}(M) \tag{5}$$
$$\mathcal{C}(M) \cup \mathcal{C}(N) \subseteq \mathcal{C}(M \cup N) \tag{6}$$
$$\mathcal{C}(M \cap N) \subseteq \mathcal{C}(M) \cap \mathcal{C}(N) \tag{7}$$
$$\mathcal{C}(\mathcal{C}(M)) = \mathcal{C}(M) \tag{8}$$

In Worten: die konvexe Hülle einer Menge M ist konvex (4) und enthält M vollständig (5); die Vereinigung zweier konvexer Hüllen ist Teilmenge der konvexen Hülle der Vereinigungen (6), während die konvexe Hülle der Schnittmenge eine Teilmenge des Schnitts der einzelnen konvexen Hüllen ist (7); schließlich ist die Bildung der konvexen Hülle eine idempotente Operation (8).

1. Implementieren Sie die Eigenschaften (4) bis (8) als QuickCheck Properties unter Verwendung der Funktionen `convexHull`, `isInside` und `isConvex` sowie `Set.union`, `Set.intersection` und Freunden.

2. Drücken Sie drei weitere Eigenschaften ihrer Implementierung als QuickCheck Properties aus. Eine davon soll den Problemfall der kolinearen Punkte erfassen, d.h. die Bildung der konvexen Hülle einer Menge, die mehrere auf einer Linie liegende Punkte enthält. (2 Bonuspunkte)

Hinweis: Es stehen zwar keine Funktionen zur Bestimmung der Teilmengenbeziehung zwischen Polygonen bzw. für die Bildung der Vereinigung und dem Durchschnitt über diesen bereit. Sie können sich jedoch die Äquivalenzen

$$M \subseteq N \iff (\forall x.\, x \in M \implies x \in N) \tag{9}$$
$$x \in (M \cup N) \iff (x \in M \vee x \in N) \tag{10}$$
$$x \in (M \cap N) \iff (x \in M \wedge x \in N) \tag{11}$$

zunutze machen, um gleichwertige Eigenschaften mit den vorhandenen Operationen zu spezifizieren.

Gutes Gelingen!

Revision 1290 vom 2011-01-12