7. Übung zur Vorlesung: **Felsner/ Heldt**
Algorithmische Geometrie (KG II) 04. Dezember
Abgabe: 11. Dezember

---

(1) Gegeben sei eine n–elementige Punktmenge P in der Ebene. Gib einen effizienten, randomisierten Algorithmus an, der die Kreisscheibe mit kleinstem Radius K_P bestimmt, die P enthält.

(a) Zeige dazu zunächst, dass der Punkt p, falls er nicht in dem kleinsten, die Menge P umfassenden Kreis K_P liegt, auf dem Rand des kleinsten Kreises liegt, der $P \cup \{p\}$ enthält. Weise weiterhin nach, dass stets zwei oder drei der Punkte von P auf dem Rand von K_P liegen und K_P durch diese eindeutig bestimmt ist.

(b) Entwirf eine Methode, die den kleinsten Kreis $K_{P\cup\{p,q\}}$ berechnet, so dass $P \cup \{p, q\}$ in $K_{P\cup\{p,q\}}$ enthalten ist und p und q auf dem Rand des Kreises liegen.

(c) Verwende (b) um eine Methode zu entwickeln, die den kleinsten Kreis $K_{P\cup\{p\}}$ bestimmt, so dass $P \cup \{p\}$ in $K_{P\cup\{p\}}$ enthalten ist und p auf dem Rand des Kreises liegt.

(d) Nutze (c) um den allgemeinen Algorithmus zur Berechnung von K_P anzugeben. Analysiere die erwartete Laufzeit Deines Algorithmus mit Hilfe einer Rückwärtsanalyse.

(2) Zur Implementierung von randomisierten Algorithmen sind Zufallszahlen eine wichtiges Element. Im Computer werden dazu Pseudozufallszahlen generiert und verwendet. Eine einfache Methode dies zu tun sind *lineare Kongruenz Generatoren*. Diese berechnen iterativ aus einer Zufallszahl X_n eine Zufallszahl

$$X_{n+1} := (a \cdot X_n + c) \mod m.$$

Dabei sind $a, c, m \in \mathbb{N} \setminus \{0\}$, sowie $a < m$ und $c < m$. Üblicherweise ist dabei $m = 2^k$ für ein $k \in \mathbb{N}$ um die Modulo–Operationen effizienter implementieren zu können.

(a) Seien a und m teilerfremd. Zeige $X_{i+1} = X_{i+m}$ für alle $i > m$; also dass die Folge der Pseudozufallszahlen (X_i) mit Ausnahme der ersten Elementen die Periode m hat.

(b) Seien c und $m = 2^k$ teilerfremd und a ungerade. Zeige, dass X_{i+1} genau dann ungerade ist, wenn X_i gerade ist.

(c) Angenommen, wir nutzen einen linearen Kongruenz Generator mit Periode m zur Auswahl einer Permutation $X = (X_0, X_1, \ldots, X_{m-1}) \in S_m$. Bestimme obere und untere Schranken für $\Pr(X = \sigma)$ für alle $\sigma \in S_n$. In wie weit wirkt sich dies auf die erwarteten Laufzeiten der randomisierten Algorithmen der Vorlesung (an einem Beispiel Deiner Wahl) aus?

(3) Ein Range–Tree benötigt $O(n \log n)$ Speicher, wie in der Vorlesung gezeigt wurde. Um diesen Speicheraufwand zu verringern, speichern wir die Suchbäume in y Richtung nur auf jeder zweiten Ebene (also bei allen Knoten in dem Suchbaum bezüglich der x–Koordinate mit Tiefe 0,2,4, …). Wie kann nun korrekt beantwortet werden, welche Punkte in dem Abfrage–Viereck liegen und wie verändert sich die Größe des benötigten Speichers und die Laufzeiten für Abfragen?

(4) Implementiere den SamplingLP–Algorithmus aus der Vorlesung und nutze ihn um konvexe Hüllen zu berechnen; d.h. implementiere auch einen inkrementellen Algorithmus zur Berechnung konvexer Hüllen, der auf SamplingLP basiert.