

Übungsblatt 4 - Lineare Programmierung

Ausgabe: 08. Mai 2011

Abgabe: 15. Mai 2011

1 Korrektheitsbeweis

In der Vorlesung wurde der Algorithmus `RANDOMPERMUTATION(A)` vorgestellt.

- a) Beweise seine Korrektheit, indem du zeigst, dass jede mögliche Permutation von A gleich wahrscheinlich ist.
- b) Zeige außerdem, dass die Aussage aus a) nicht mehr stimmt, wenn wir $r \leftarrow \text{Random}(k)$ durch $r \leftarrow \text{Random}(n)$ ersetzen.

2 Paranoia

Algorithmus 1: ParanoidMax

Eingabe : Endliche Menge $A \subset \mathbb{R}$

Ausgabe : Maximum $\max_{a \in A} a$ der Menge

if $|A| = 1$ **then**

\perp **return** einziges Element $a \in A$

else

a = zufällig gewähltes Element aus A

$b = \text{ParanoidMax}(A \setminus \{a\})$

if $b \geq a$ **then**

\perp **return** b

else

 prüfe unnötigerweise jedes Element aus $A \setminus \{a\}$, um sicherzugehen, dass a wirklich größer ist

\perp **return** a

Betrachte Algorithmus 1, der das Maximum einer Menge von Zahlen berechnet. Schätze die asymptotische Worst-Case-Laufzeit des Algorithmus scharf ab. Beachte die zufällige Auswahl des Elementes a und zeige, dass die erwartete Laufzeit echt besser ist.

Bitte umblättern

3 Züge

Gegeben seien n Züge, die auf parallelen Gleisen fahren. Jeder Zug z_i ($i = 1, \dots, n$) fahre dabei mit konstanter Geschwindigkeit v_i und nehme zum Startzeitpunkt $t = 0$ (hier startet nur die Zeit, die Züge haben schon die volle Geschwindigkeit) die Position k_i auf der Strecke ein. Gebe einen Algorithmus an, der in $O(n \log n)$ Zeit berechnet, welche Züge bis zu einem gegebenen Zeitpunkt $t_{\text{stop}} > 0$ mindestens einmal in Führung waren.

Hinweis: Der in der Vorlesung vorgestellte Algorithmus zu Berechnung vom Schnitt mehrerer Halbebenen könnte dabei nützlich sein.

4 Rückblick: Polygon Partitionieren

In der dritten Vorlesung wurde ein Verfahren vorgestellt welches, mit Hilfe von einer doppelt verketteten Kantenliste (DCEL), ein einfaches Polygon in monotone Polygone auftrennt. Im Verlauf des Algorithmus werden neue Kanten zur DCEL hinzugefügt, nämlich Diagonalen um Split- oder Merge-Knoten zu entfernen. Dabei wurde angenommen, dass wir eine Kante in konstanter Zeit zur DCEL hinzufügen können.

Argumentiere, zunächst warum das Hinzufügen von Kanten in eine DCEL im Allgemeinen *nicht* in konstanter Zeit möglich ist. Zeigen dann, wie man im Fall des in der Vorlesung vorgestellten Algorithmus zur Polygon-Partitionierung Kanten trotzdem in $O(1)$ in die DCEL einfügen kann.