RWTH Aachen **SS 2010**
Lehrgebiet Theoretische Informatik Übungsblatt 11
Kneis—Langer—Rossmanith 14.07.2010

Übung zur Vorlesung Effiziente Algorithmen

Tutoraufgabe 25

Wir betrachten das folgende Online-Problem: Zu jedem Zeitpunkt ist Ihr Punktestand p, beginnend mit 0. Sie erhalten der Reihe nach Zahlen x_i und können diese entweder zu p addieren oder von p subtrahieren. Allerdings darf p nie kleiner als 0 sein. Ihr Ziel ist es, das maximal erreichte p möglichst klein zu halten.

Beispiel	Zulässige Lösung	Optimale Lösung
$1, 3, 3, 4, 5, 2, 1$	$1+3+3-4+5-2-1$ (Kosten 8)	$1+3-3+4-5+2-1$ (Kosten 5)

- Entwerfen Sie einen deterministischen Online-Algorithmus, der dieses Problem mit Competitive-Faktor 2 löst.
- Zeigen Sie, daß kein deterministischer Online-Algorithmus mit Competitive-Faktor kleiner als 1.5 existiert.

Lösung:

- Eine mögliche Lösung ist es, wann immer möglich zu subtrahieren. Ist zu einem beliebigen Zeitpunkt die aktuelle Zahl $x \leq p$, so wird x von p subtrahiert.

  Es bezeichne *opt* die Kosten der optimalen Lösung, dann gilt $x_i \leq opt$ für alle x_i aus der Eingabesequenz. Nach dem Bearbeiten von x_i ist p aber höchstens $2x_i - 1$, da sonst x_i subtrahiert worden wäre. Somit ist der Algorithmus 2-competitive.

- Sei A ein beliebiger Online-Algorithmus für dieses Problem. Unabhängig vom Verhalten von A sind die ersten beiden Zahlen beide 1. Addiert der Algorithmus diese beiden Zahlen, so werden in Zukunft nur 1 gesendet, woraus sich ein Competitive-Faktor von 2 ergibt. Falls A dagegen die zweite 1 subtrahiert, so werden als nächstes 2 und 4 gesendet. A muss diese nun beide zu $1-1$ addieren und erreicht somit Kosten 6, optimal wären dagegen 4. Insgesamt ergibt sich eine Schranke von 3/2.

Tutoraufgabe 26

Sei

$$F = \sum_{t=1}^{T} f_t,$$

wobei f_t die Anzahl der frischen Seiten in der t-ten Phase bezeichnet und T die Anzahl der Phasen. Beweisen Sie folgende Aussage aus der Vorlesung:
Eine optimale Caching-Strategie verursacht mindestens $\frac{1}{2}(F)$ viele Seitenfehler.

Lösung

Betrachten wir zuerst eine einzelne Phase t. Sei f die Anzahl der frischen Seiten, die in dieser Phase geladen werden. Sei außerdem S_a der Cache des Markierungsalgorithmus und S_a^* der Cache einer optimalen Strategie am Anfang der Phase, und analog S_e, S_e^* am Ende der Phase. Wir setzen $d_a^t = |S_a^* \setminus S_a|$ und $d_e^t = |S_e^* \setminus S_e|$.

Da der randomisierte Markierungsalgorithmus nur nicht markierte Seiten verdrängen kann, muss er am Anfang der Phase alle vorher markierten Seiten im Cache haben. Er erzeugt in der aktuellen Phase also mindestens f_t viele Seitenfehler, da alle frischen Seiten neugeladen werden müssen. Da der Cache $S*$ der optimalen Strategie genau d_a^t andere Seiten als S enthält, muss die optimale Strategie also mindestens $f_t - d_a^t$ Seitenfehler aufweisen.

Am Ende der Phase enthält S alle Seiten, die in dieser Phase geladen wurden, da es sich um einen Markiergunsalgorithmus handelt. Insbesondere wurde auf jede dieser Seiten in dieser Phase zugegriffen. Da S^* jedoch d_e^t dieser Seiten nicht mehr enthält, müssen mindestens diese d_e^t Seiten aus dem Cache verdrängt worden sein. Es gab also mindestens d_e^t viele Seitenfehler.

Somit treten in der optimalen Strategie mindestens $\max\{f_t - d_a^t, d_e^t\}$ viele Seitenfehler in dieser Phase auf. Wir nutzen die Abschätzung

$$\max\{f_t - d_a^t, d_e^t\} \geq 1/2(f_t - d_a^t + d_e^t)$$

und erhalten die Kosten C der optimalen Lösung

$$C \geq 1/2 \sum_{t=1}^{T} \left(f_t - d_a^t + d_e^t\right).$$

Da offensichtlich $d_e^t = d_a^{t-1}$ gilt, folgt

$$C \geq \sum_{t=1}^{T} f_t - d_a^1 + d_e^T.$$

Die Behauptung folgt nun, da der Cache beider Strategien am Anfang gleich ist und somit $d_a^1 = 0$ ist.

Hausaufgabe 22 (10 Punkte)

Beweisen oder widerlegen Sie:

- Es gibt ein $c \in \mathbf{N}$, so daß LIFO c-kompetitiv ist.

Lösung

Wir beweisen, daß FIFO nicht kompetitiv ist.

Angenommen LIFO wäre c-kompetitv für irgendein festes c. Wir betrachten dann die Eingabe $1, 2, \ldots, k, (k+1, k)^c$. LIFO wird erst den Cache mit den Seiten $1, \ldots, k$ füllen, und muss dann die Seite $k+1$ nachladen. Da die Seite k zuletzt geladen wurde, wird diese überschrieben. Danach wird aber wiederum k geladen, und $k+1$ entfernt. Dies wird c mal wiederholt und resultiert somit in $2c$ Seitenfehlern.

Eine optimale Strategie kommt offensichtlich mit einem Seitenfehler aus, indem einfach Seite 1 verdrängt wird, sobald $k+1$ geladen wird.

Es gibt also kein c, so daß LIFO c-kompetitiv ist.

Hausaufgabe 23 (10 Punkte)
Beweisen oder widerlegen Sie die folgenden Aussagen:

- FIFO ist ein Markierungsalgorithmus.
- FIFO ist k-kompetitiv.

Lösung
Zuerst zeigen wir, daß FIFO kein Markierungsalgorithmus ist. Dazu betrachten wir die Eingabe $1, 2, \ldots, k, k+1, 2, 1$.
Nach dem Einlesen von $1, \ldots, k$ sind genau diese Werte im Cache und die Phase endet. Einlesen von $k+1$ beginnt die nächste Phase, 1 wird verdrängt. Nun wird 2 markiert, danach aber beim lesen von 1 sofort verdrängt.
Wir zeigen nun weiterhin, daß FIFO trotzdem k-kompetitiv ist.
Leider enthält FIFO am Ende einer Phase nicht genau die Seiten, die in der Phase nachgefragt wurden, wie obiges Beispiel zeigt.
Betrachten wir aber nun eine Eingabe $p_1, \ldots, p_n$, und sei P_i der Zeitpunkt, an dem FIFO genau ik Seitenfehler macht. Zwischen P_i und P_{i+1} treten in FIFO also genau k Seitenfehler auf.
Allerdings müssen zwischen P_i und P_{i+1} mindestens k verschiedene Seiten geladen werden. Angenommen es werden nur die Seiten $s_1, \ldots, s_l$ $l < k$ geladen, so bleiben diese im Cache von FIFO und werden nicht verdrängt, FIFO würde also höchstens l Seitenfehler machen. Damit sind die Abstände zwischen P_i und P_{i+1} aber mindestens so lang wie die einzelnen Phasen.
Da die optimale Lösung mindestens soviele Fehler macht wie es Phasen gibt, ist FIFO also k-kompetitiv.