Zufällige Verdrängungsstrategien

Wir betrachten folgenden Algorithmus:

Bei einem Seitenfehler wird eine zufällige gewählte nicht markierte Seite verdrängt.

Satz

Dieser Algorithmus ist $2H_k$-kompetitiv.

$H_k = \ln k + O(1)$

Einige Bezeichnungen:

Eine unmarkierte Seite, auf die in der letzten Phase zugegriffen wurde, nennen wir eine vorherige Seite.

Eine unmarkierte Seite, die nicht vorherig ist nennen wir eine frische Seite.

Es sei

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

mit $f_t =$ Anzahl der frischen Seiten, auf die in der $t$ten Phase zugegriffen wird und $T =$ Anzahl der Phasen.

Lemma

$E(C(I)) \leq F \cdot H_k$.

Beweis

Betrachte die $t$te Phase. Es gibt f_t Seitenfehler bei Zugriffen auf frische Seiten.

Anfang der Phase: Alle k Seiten im Cache vorherig.

In der Phase wird auf k unterschiedliche Seiten zugegriffen. Davon sind $s = k - f_t$ vorherige Seiten.

$$\sum_{i=1}^{s} \frac{f_t}{k+1-i} \leq \sum_{i=2}^{k} \frac{f_t}{i} = f_t(H_k - 1)$$

Lemma

Eine optimale Strategie verursacht mindestens $\frac{1}{2}(F + k)$ Seitenfehler.

Beweis

Übungsaufgabe.

Aus beiden Lemmata folgt der Beweis.

Randomisierte Algorithmen

Ein sehr weites Gebiet.

- Las Vegas– und Monte Carlo–Algorithmen
- Universales Hashing, Bloom-Filter
- Symmetry Breaking
- Routing-Algorithmen
- Derandomisierung
- . . .

Einfaches Beispiel

Eingabe: Zwei Polynome

Frage: Sind sie gleich?

$$
\begin{aligned}
& (x-13)(x-5)(x-3)(x+7)(x^2+12) \\
\stackrel{?}{=} \quad & x^6 - 14x^5 - 16x^4 + 472x^3 - 1701x^2 + 7656x - 16380
\end{aligned}
$$

Ausmultiplizieren: $\Theta(d^2)$ Multiplikationen

Geht es schneller?