technische universität dortmund

Christian Sohler, Melanie Schmidt
Lehrstuhl 2, Fakultät für Informatik
Wintersemester 2009/2010

AuD Übung – Präsenzübung 6

Ausgabedatum: 06.01.2010 — Besprechung: 11.01.2010

Das Listen-Zugriffs-Problem:

Wir haben in der Vorlesung den Algorithmus MTF (Move to Front) für das Listen-Zugriffs-Problem besprochen, der 2-competitive ist. Im Folgenden wollen wir zeigen, dass es keinen viel besseren deterministischen Online-Algorithmus für das Listen-Zugriffs-Problem geben kann. Den Beweis werden wir in Form von einzelnen Teilaufgaben erarbeiten.

Aufgabe 6.1: Deterministische Online-Algorithmen

Die Worst-Case-Laufzeit aller deterministischen Online-Algorithmen für das Listen-Zugriffs-Problem ist gleich. Warum? Wie groß ist sie (für eine Liste der Länge ℓ und eine Sequenz der Länge n)? Betrachte einen grausamen Gegenspieler.

Aufgabe 6.2: Spezielle Offline-Algorithmen

Betrachte den speziellen Typ von Offline-Algorithmen für das Listen-Zugriffs-Problem, bei dem die Liste nur im ersten Schritt verändert wird und für alle weiteren Anfragen gleich bleibt. Wie viele verschiedene Algorithmen dieser Art gibt es? Gib außerdem eine (einfache) obere Schranke dafür an, wie viel ein solcher Algorithmus für das Umsortieren der Liste im ersten Schritt bezahlen muss.

Aufgabe 6.3: Durchschnittliche Gesamtkosten

Wir wollen jetzt folgenden Trick anwenden: Anstatt die Gesamtkosten eines bestimmten Offline-Algorithmus zu berechnen, berechnen wir die *Summe* der Gesamtkosten aller Algorithmen des Typs aus Aufgabe 6.2. Anschließend berechnen wir die durchschnittlichen Gesamtkosten. Dann muss es mindestens einen Offline-Algorithmus geben (auch wenn wir den nicht kennen), der höchstens diese Gesamtkosten hat. Also: Berechne diese Summe!

Aufgabe 6.4: Eine untere Schranke

Setze die Ergebnisse aus den bisherigen Aufgaben zusammen, um zu zeigen, dass jeder deterministische Online-Algorithmus mindestens ein competitive ratio von $2 - \frac{2}{\ell+1}$ hat.