Max Wisniewski , Alexander Steen

Tutor: Lena Schlipf

Aufgabe 1 Caching

(a) Zeigen Sie, dass jede Offline-Caching-Strategie durch eine *reduzierte* Ersetzungsstrategie ersetzt werden kann, die nicht die Anzahl der Hauptspeicherzugriffe erhöht.

Beweis:

Sei $Z = z_1 z_2 ... z_m$ eine Anfragefolge auf den Speicher.
Sei $D = d_1 d_2 ... d_n$ eine allgemeine Zugriffsfolge einer Ersetzungsstrategie. Mit zugehörigen $T = t_1 t_2 ... t_n$ Zeitpunkten, so dass d_i bei Anfrage z_{t_i} passiert. Sei $T_{red} = a_1 a_2 ... a_n$ eine Folge mit Zeitpunkten, so dass d_i bei Anfrage z_{a_i} passiert und bei z_{a_i} ein Cache - Miss aufgetreten ist.

Wir wollen nun T' schrittweise konstruieren, so dass T' am Ende eine reduzierte Folge ist induziert. Sei T' initial T.

Sei $i = \min\{i \in \{1, ..., n\} \mid t_i \neq a_i\}$. Insbesondere gilt $t_1 .. t_{i-1} = a_1 ... a_{i-1}$. Bei gleicher Startkonfiguration ist nach $z_{t_{i-1}}$ der Cache bei

(b) Geben Sie eine allgemeine Zugriffsfolge an, so dass die LRU Strategie bei k Wörtern k mal so viele Misses wie die Furthest-in-the-Future Strategie erzeugt.

Lösung:

Aufgabe 2 Union-Find

Betrachten Sie eine Folge von **Union** und **Find** Operationen der Länge m. Gegeben ist die Startpartition $\{\{1\}, \{2\}, ..., \{n\}\}$. Verwendet werden sowohl Union-By-Rank also auch Pfadkompression.

(a) Werden alle **Union**-Operationen vor allen **Find** - Operationen durchgeführt, so ist die Gesamtlaufzeit $O(n + m)$.

Beweis:

Idee: Union : Kosten 2

(b) Wenn alle **Find** - Operationen auf Mengen mit mindestens $\log\ n$ Elementen durchgeführt werden, so ist die Gesamtlaufzeit $O(n + m)$.

Beweis:

Tipp: Im $\log *n$ Teil steht es. Hauptlemma + andere Aufteilung (nicht der Rang)

Aufgabe 3 Bitmaps

Sie n eine natürtliche Zahl. Eine $n \times n$ Bitmap ist ein Array $B[1...n, 1...n]$ vom Typ `Boolean`, welches ein Schwarz-Weiß-Bild darstellt.

(a) Entwerfen und analysieren Sie einen effizienten Algorithmus, der eine größte Zusammenhangskomponente von schwarzen Pixeln in B bestimt.

Lösung:

(b) Beschreiben und analysieren Sie eine Funktion schwärze(i,j), die das Pixel an der Stelle $B[i, j]$ schwarz färbt und die Größe einer größten Zusammenhangskomponente von schwarzen Pixeln zurückgibt. Nehmen Sie an, dass zu Beginn alle Pixel weiß sind. Die Gesamtlaufzeit für jede Folge von m Aufrufen von schwärze sollte so gering wie möglich sein.

Lösung:

(c) Was ist die worst-case Laufzeit für einen Aufruf Ihrer Funktion schwärze aus (b)?

Lösung:

Aufgabe 4 Matroide

Sei S eine endliche, nichtleere Menge und sei $\mathfrak{I} \subseteq 2^S$ eine nichtleere Menge von Teilmengen von S. Das Paar $M = (S, \mathfrak{I})$ soll ein Matriod, nach Definition aus der Aufgabe sein.

(a) Eine inklusionsmaximale unabhängige Menge heißt *Basis* von M. Zeigen Sie, dass alle Basen von M die gleiche Anzahl von Elementen haben.

Beweis: (Widerspruch)
Angenommen A, B sind inklusionsmaximale unabhängige Mengen von M, mit o.B.d.A. $|A| < |B|$.
Dann wissen wir, dass $B \setminus A \neq \emptyset$ ist, die mindestens die überzähligen im Schnitt liegen und wenigestens ein weiteres Element, da A sonst nicht inklusionsmaximal wäre.

Nun können wir nach Austauscheigenschaft $x \in B \setminus A$ nehmen und $A \cup \{x\}$ bilden, so dass $A \cup \{x\} \in \mathfrak{I}$ sein muss. Nun ist aber A nicht inklusionsmaximal, da $A \subset A \cup \{x\}$ ist.

$\square$

(b) Sei $w \ : S \longrightarrow \mathbb{R}^+$ eine Gewichtsfunktion. Gesucht ist eine Basis von von M mit maximalem Gewicht, wobei das Gewicht einer Teilmenge die Summer der Einzelgewichte ist. Der Algorithmus funktioniert wie folgt:

- Sortiere S absteigend.
- Setzt $B := \emptyset$.
- Gehe S Elementweise durch (nach Ordnung.
  Füge ein $x \in S$ zu B hinzu, wenn B dadurch unabhängig bleibt.

- Gib B zurück.

Zeigen Sie, dass der gierige Algorithmus eine Basis von maximalem Gewicht bestimmt. Was können Sie zur Laufzeit sagen?

Korrektheit:

bla

Laufzeit:

blub