

Mitschrift der Algorithm Engineering Vorlesung vom 12.12.05

Martin Groß

1 Externe Array-Heaps

Die Parameter B, M sind von nun an gewählt wie im EM-Modell, das heißt also

B = Anzahl der Elemente, die in einen Block passen

M = Anzahl der Elemente, die in den Hauptspeicher passen

Ferner sei der Parameter c eine Konstante. Es folgen nun Ergänzungen zu den Beweisen der Lemmata aus der Vorlesung:

Lemma 1. *Es gilt*

$$l_{i+1} = l_i(\mu + 1)$$

wobei

μ = Anzahl der Slots pro Schicht

l_i = Anzahl der Plätze in einem Slot der i -ten Schicht

ist.

Beweis. Die Anzahl Slots / der Plätze in einem Slot der i -ten Schicht ist

$$\mu = \frac{cM}{B} - 1$$
$$l_i = \frac{(cM)^i}{B^{i-1}}$$

Damit lässt sich obige Behauptung schreiben als

$$l_{i+1} = \frac{(cM)^{i+1}}{B^i} = \frac{cM}{B} \cdot \frac{(cM)^i}{B^{i-1}} = (\mu + 1)l_i$$

Somit ist das Lemma gezeigt. □

Lemma 2. *Es befinden sich maximal $cM(2 + L)$ Elemente im Hauptspeicher, wobei L die Anzahl der Slot-Schichten ist.*

Beweis. Wir wissen, dass Heap 1 die neu eingefügten Elemente enthält (maximal $2cM$ von ihnen) und Heap 2 maximal die kleinsten B Elemente jedes belegten Slots enthält. Also muss gelten:

$$\begin{aligned} \text{Elemente im Hauptspeicher} &= \text{Elemente in Heap 1} + \text{Elemente in Heap 2} \\ &\leq 2cM + \mu LB \\ &= 2cM + (\mu + 1)LB - LB \\ &= 2cM + \frac{cM}{B}LB - LB \\ &= 2cM + cML - LB \\ &= cM(2 + L) - LB \\ &\leq cM(2 + L) \end{aligned}$$

Damit ist das Lemma bewiesen. □

Folgerung. Es sind maximal $cM(2+L)$ Elemente im Hauptspeicher, für die Sortier-Operationen wird $(\mu+1)B$ Platz benötigt, es folgt

$$M \geq cM(2+L) + (\mu+1)B = cM(2+L) + cM = cM(3+L)$$

Für $c = \frac{1}{7}$ bedeutet dies beispielsweise, dass $L \leq 4$ sein muss.

Lemma 3. *Das kleinste Element ist immer in Heap 1 oder Heap 2.*

Beweis. Fallunterscheidung:

1. Das kleinste Element befindet sich in einem Slot. Dann ist das kleinste Element notwendigerweise auch das kleinste Element des Slots, der dann offensichtlich nicht leer sein kann. Folglich ist das kleinste Element in Heap 2 enthalten.
2. Das kleinste Element befindet sich in keinem Slot. Dann muss das kleinste Element neu eingefügt worden sein und somit in Heap 1 enthalten sein.

Folglich ist das kleinste Element immer in Heap 1 oder in Heap 2, die Behauptung ist gezeigt. \square

Lemma 4. *Bei Ausführung von $\text{Store}(i, S)$ ist immer garantiert, dass S zwischen $l_i/2$ und l_i Elementen enthält.*

Beweis. \rightarrow siehe nächste Vorlesung \square

Annahme. Im Folgenden sei nun immer $cM > 3B$.

Lemma 5. *Nach N Operationen existieren höchstens*

$$L \leq \log_{cM/B} \left(\frac{N}{B} \right)$$

Schichten.

Beweis. Im Worst-Case sind alle N Operationen Insert-Operationen. In diesem Fall wird bei jedem Überlauf die maximal mögliche Anzahl von Elementen in die nächste Schicht bewegt. Die Anzahl der Elemente in j Slot-Schichten ist also

$$\begin{aligned} \sum_{i=1}^j l_i \mu &= \sum_{i=1}^j \frac{(cM)^i}{B^{i-1}} \mu \\ &= \sum_{i=1}^j \frac{(cM)^i}{B^{i-1}} \left(\frac{cM}{B} - 1 \right) \\ &= \sum_{i=1}^j \frac{(cM)^{i+1}}{B^i} - \sum_{i=1}^j \frac{(cM)^i}{B^{i-1}} \\ &= \frac{(cM)^{j+1}}{B^j} - cM \end{aligned}$$

Gesucht wird nun die kleinste Anzahl von Schichten j , in die alle N Elemente passen:

$$\begin{aligned} N &\leq \text{Max. Elemente in } j \text{ Slot-Schichten} + \text{Max. Elemente im Heap} \\ &= \frac{(cM)^{j+1}}{B^j} - cM + 2cM \\ &= \frac{(cM)^{j+1}}{B^j} + cM \\ &= cM \left(\frac{(cM)^j}{B^j} + 1 \right) \end{aligned}$$

Auflösen nach j liefert

$$\begin{aligned} N &\leq cM \left(\frac{(cM)^j}{B^j} + 1 \right) \\ \Leftrightarrow \frac{N}{cM} - 1 &\leq \frac{(cM)^j}{B^j} \\ \Leftrightarrow \log_{cM/B} \left(\frac{N}{cM} - 1 \right) &\leq j \end{aligned}$$

Wähle nun j minimal, dass heißt

$$j = \left\lceil \log_{cM/B} \left(\frac{N}{cM} - 1 \right) \right\rceil$$

Wegen $cM > 3B$ folgt

$$\frac{N}{cM} - 1 < \frac{N}{3B} - 1 < \frac{N}{B}$$

Folglich ist $\log_{cM/B}(\frac{N}{B})$ eine obere Schranke für die Anzahl der Schichten, das Lemma ist bewiesen. \square

Lemma 6. *Store(i, S) benötigt höchstens $3\frac{l_i}{B}$ I/O's. Compact($i+1$) und Merge-Level(i, S, S') benötigen höchstens $3\frac{l_{i+1}}{B}$ I/O's.*

Beweis. Betrachte den Aufwand für die Operationen (unter der Annahme $cM > 3B$):

1. **Store(i, S)**

Es wird S gelesen und in L_i gespeichert:

$$\left\lceil \frac{|S|}{B} \right\rceil + \left\lceil \frac{l_i}{B} \right\rceil \leq 2 \left\lceil \frac{l_i}{B} \right\rceil \leq 2\frac{l_i}{B} + 2 \leq 3\frac{l_i}{B}$$

2. **Merge-Level(i, S, S')**

Es wird jedes Element von S und jeder Slot der i -ten Schicht maximal einmal gelesen und geschrieben:

$$2 \left\lceil \frac{|S| + \mu l_i}{B} \right\rceil \leq 2 \left\lceil \frac{l_{i+1}}{B} \right\rceil \leq 2\frac{l_{i+1}}{B} + 2 \leq 3\frac{l_{i+1}}{B}$$

3. **Compact($i+1$)**

Seien $|S_1|$ und $|S_2|$ die Anzahlen der Elemente der Slots, die gemischt werden. Diese Slots müssen jeweils einmal gelesen werden, anschließend wird ein Slot geschrieben:

$$\lceil |S_1| \rceil + \lceil |S_2| \rceil + \lceil l_{i+1} \rceil \leq \frac{|S_1| + |S_2| + l_{i+1}}{B} + 3 \leq 2\frac{l_{i+1}}{B} + 3 \leq 3\frac{l_{i+1}}{B}$$

Damit ist das Lemma gezeigt. \square