Satz 2.7

Algorithmus 2.6 findet eine Optimallösung in $O(nC)$.

Beweis:

Die Laufzeit ist klar.

Die Variable $x(i,k)$ beschreibt die kleinstmögliche Gesamtgröße einer Teilmenge $S \subseteq \{1,\dots,i\}$ mit $\sum_{i \in S} c_i = k$. Der Algorithmus berechnet diese Funktion mit der Rekursionsformel

$$x(i,k) = \begin{cases} x(i-1, k-c_i) + l_i & \text{wenn } c_i \le k \\ & \text{und } x(i-1,k-c_i)+l_i \le \min\{l_i, x(i-1,k)\} \\ x(i-1,k) & \text{sonst} \end{cases}$$

Die Variable $s(i,k)$ zeigt an, welcher der beiden Fälle gilt.

Der Algorithmus enumeriert also alle Teilmengen $S \subseteq \{1,\dots,n\}$ – bis auf die unzulässigen oder diejenigen für die es eine bessere Lösung S' gibt, d.h. $\sum_{i \in S} c_i = \sum_{i \in S'} c_i$ und $\sum_{i \in S} l_i > \sum_{i \in S'} l_i$.

In (3) wird die beste Teilmenge ausgewählt.

□

2.3 Heuristiken für Bin Packing

Umzugsproblem: Gegeben: Objekte $l_1, \dots, l_n \in [0,1]$
Umzugskartons der Größe 1

Gesucht: Aufteilung in möglichst wenige Kartons.

Idee: Packe wie es passt!

Algorithmus 2.8

Eingabe: Folge von Objekten $l_1, \dots, l_n \in [0,1]$

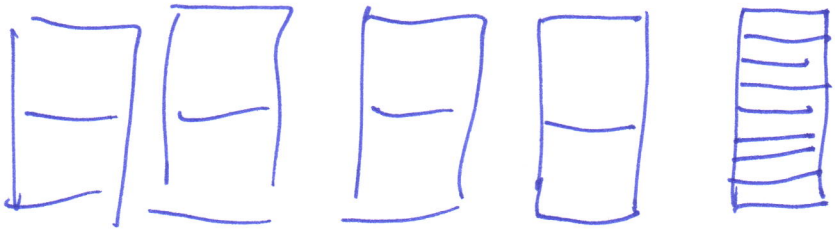
Ausgabe: Aufteilung in Gruppen, die jeweils in einen Karton passen.

(1) Setze $j = 1$
(2) Für jedes l_i tue Folgendes:
(3) IF (l_i hat in Karton K_j keinen Platz mehr) {
(4) Schließe Karton K_j.
(5) Setze $j := j+1$
} {
(6) Packe l_i in Karton K_j.

Beispiel 2.9

$\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{8}, \frac{1}{8}, \frac{1}{8}, \frac{1}{8}, \frac{1}{8}, \frac{1}{8}, \frac{1}{8}, \frac{1}{8}$

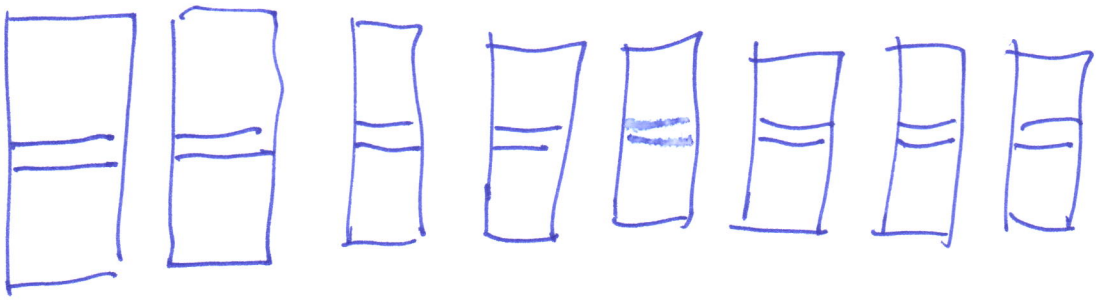
ergibt



→ 5

Beispiel 2.10

$\frac{1}{2}, \frac{1}{8}, \frac{1}{2}, \frac{1}{8}, \frac{1}{2}, \frac{1}{8}, \frac{1}{2}, \frac{1}{8}, \frac{1}{2}, \frac{1}{8}, \frac{1}{2}, \frac{1}{8}, \frac{1}{2}, \frac{1}{8}, \frac{1}{2}, \frac{1}{8}$



→ 8

Also: Next Fit braucht fast doppelt so viel Kartons wie im Optimum!

Das ist auch wirklich der schlechteste Fall:

Satz 2.10

Algorithmus 2.8 liefert in $O(n)$ eine Lösung mit Wert $NF(I)$, der

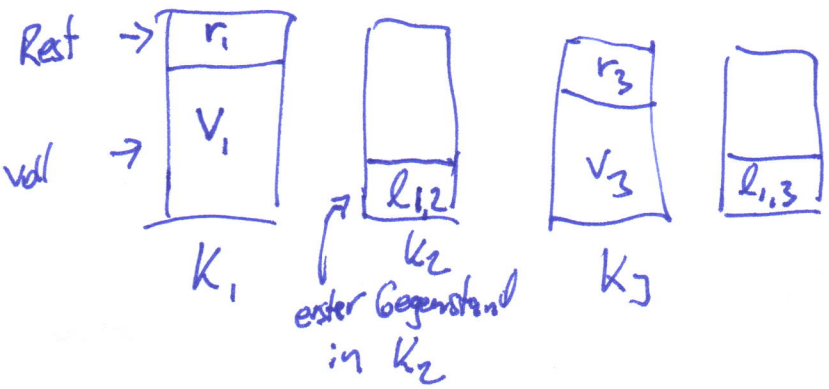
$$NF(I) \leq 2\,OPT(I) - 1$$

erfüllt, d.h. weniger als der doppelte Wert des Optimums $OPT(I)$.

Beweis:

Die Laufzeit ist klar.

Außerdem gilt



(etc.)

also $\ell_{1,2j} > r_{2j-1}$

d.h. $$\sum_{\ell_i \in \{K_{2j-1}, K_{2j}\}} \ell_i > 1 \,.$$

Also gilt $$\left\lfloor \frac{NF(I)}{2} \right\rfloor < \sum_{i=1}^{n} \ell_i \,.$$

Da $\left\lfloor \frac{NF(I)}{2} \right\rfloor$ eine ganze Zahl ist,

gilt

$$\frac{NF(I)-1}{2} \leq \left\lfloor \frac{NF(I)}{2} \right\rfloor \leq \left\lceil \sum_{i=1}^{n} l_i \right\rceil - 1,$$

also gilt

$$NF(I) \leq 2 \left\lceil \sum_{i=1}^{n} l_i \right\rceil - 1.$$

Da offensichtlich

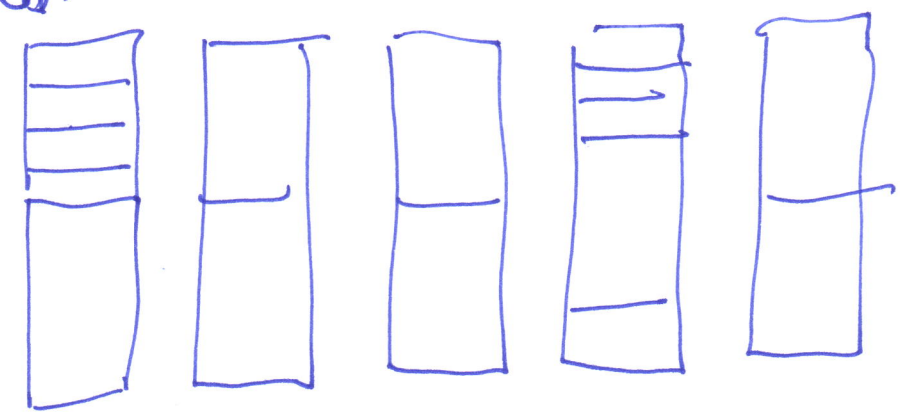
$$\left\lceil \sum_{i=1}^{n} l_i \right\rceil \leq OPT(I),$$

gilt die Behauptung. □

Beispiel 2.9 zeigt, dass NextFit etwas kurzsichtig ist: Es wäre ja besser, die Kartons später noch für kleinere Gegenstände geöffnet zu lassen!

Das liefert:

(Beispiel 2)



– also eine Optimallösung!

Also

Algorithmus 2.11 (FirstFit)

Eingabe: Folge von Objekten $l_1, \ldots, l_n \in [0,1]$

Ausgabe: Aufteilung in Gruppen, die jeweils in einen Karton passen.

(1) FOR i=1 TO n DO {

Packe i in den ersten Karton, in dem noch Platz ist.

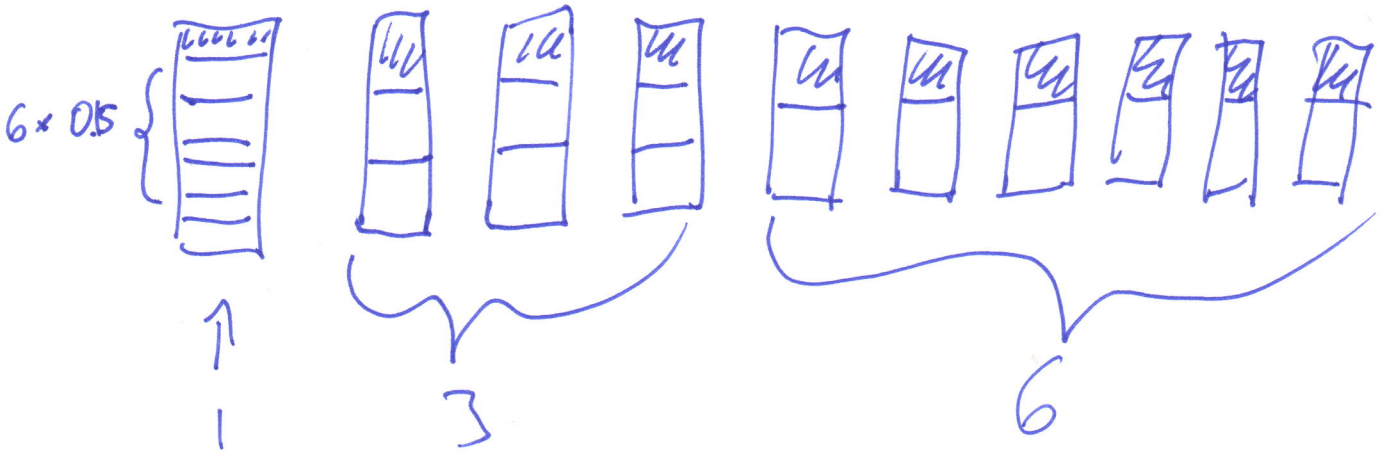
}

Natürlich liefert FIRST FIT nicht immer eine Optimallösung!

Beispiel 2.12

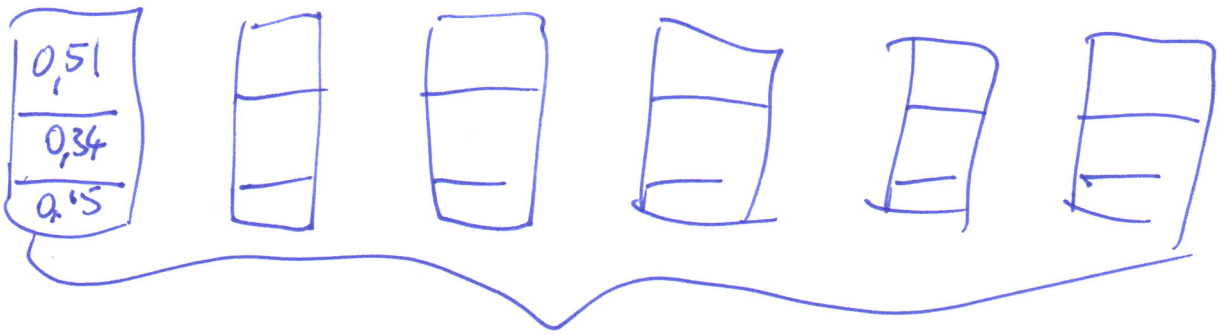
$6 \times 0{,}15$, $6 \times 0{,}34$, $6 \times 0{,}51$!

FirstFit liefert



– also 10.

Optimal:



6

Also Verhältnis $\frac{10}{6} = 1{,}666\ldots$

Satz 2.13 (Johnson+ 1974, Garey+ 1976)

Es gilt

$$FF(I) \leq \left\lceil \frac{17}{10} OPT(I) \right\rceil$$

und es gibt Beispiele I mit $OPT(I)$ beliebig groß und

$$FF(I) \geq \frac{17}{10}(OPT(I)-1).$$

Beweis: Kompliziert! Nicht hier...

Noch etwas cleverer:

Algorithmus 2.14 (First Fit Decreasing (FFD))

Eingabe, Ausgabe } wie First Fit

(1) Sortiere die Objekte in absteigender Größe

(2) Wende First Fit an.

Satz 2.15 (Garey + Johnson 1979)

FFD kann einen Faktor $\frac{11}{9}$ vom Optimum entfernt sein.

Beweis:

$$6m * \left(\tfrac{1}{2}+\varepsilon\right)$$
$$6m * \left(\tfrac{1}{4}+2\varepsilon\right)$$
$$6m * \left(\tfrac{1}{4}+\varepsilon\right)$$
$$12m * \left(\tfrac{1}{4}-2\varepsilon\right)$$

~~FFD~~ liefert Optimal ist

$$6m * \left(\left(\tfrac{1}{2}+\varepsilon\right), \left(\tfrac{1}{4}+\varepsilon\right), \left(\tfrac{1}{4}-2\varepsilon\right)\right)$$
$$3m * \left(\left(\tfrac{1}{4}+2\varepsilon\right), \left(\tfrac{1}{4}+2\varepsilon\right), \left(\tfrac{1}{4}-2\varepsilon\right), \left(\tfrac{1}{4}-2\varepsilon\right)\right)$$

FFD liefert

$$6m * \left(\left(\tfrac{1}{2}+\varepsilon\right), \left(\tfrac{1}{4}+2\varepsilon\right)\right)$$
$$2m * \left(\left(\tfrac{1}{4}+\varepsilon\right), \left(\tfrac{1}{4}+\varepsilon\right), \left(\tfrac{1}{4}+\varepsilon\right)\right)$$
$$3m * \left(\left(\tfrac{1}{4}-2\varepsilon\right), \left(\tfrac{1}{4}-2\varepsilon\right), \left(\tfrac{1}{4}-2\varepsilon\right), \left(\tfrac{1}{4}-2\varepsilon\right)\right)$$

□

Satz 2.16 (Yue 1990)

$$FFD \leq \frac{11}{9} OPT(I) + 1$$

Beweis: kompliziert!