

Algorithmen II

Vorlesung am 16.01.2014

Approximierende Algorithmen: APAS für Bin Packing

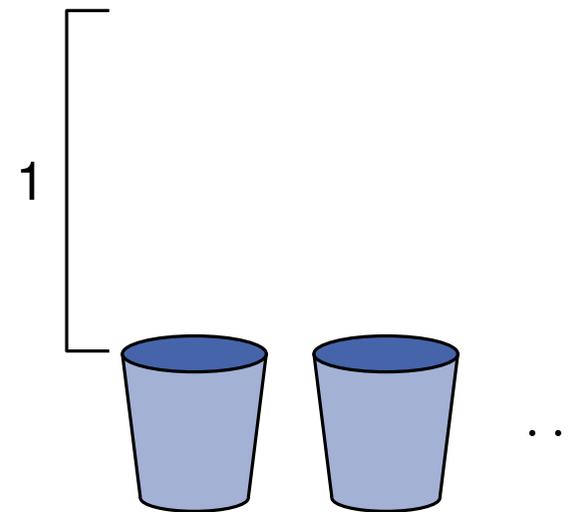
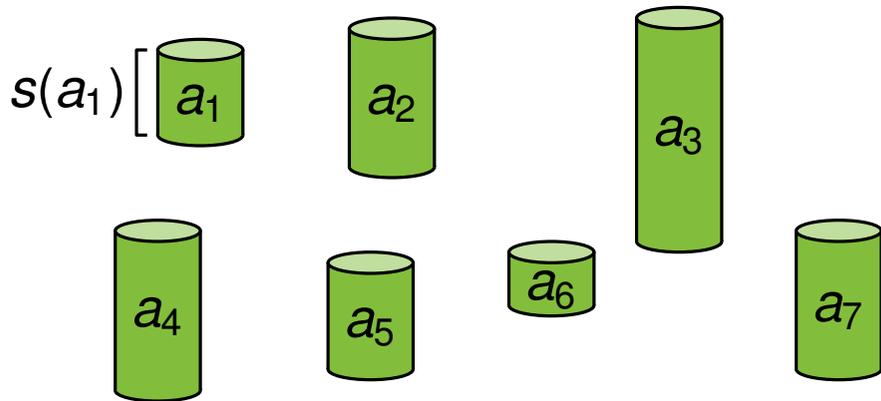
INSTITUT FÜR THEORETISCHE INFORMATIK · PROF. DR. DOROTHEA WAGNER



APAS für Bin Packing

Wiederholung: Bin Packing – Definition

endliche Menge $M = \{a_1, \dots, a_n\}$
mit Gewichtsfunktion $s: M \rightarrow (0, 1]$



Eimer (Bins) mit Fassungsvermögen 1

Schreibe für Elementgröße $s(a_i)$ auch kurz s_i .

Problem: BIN PACKING

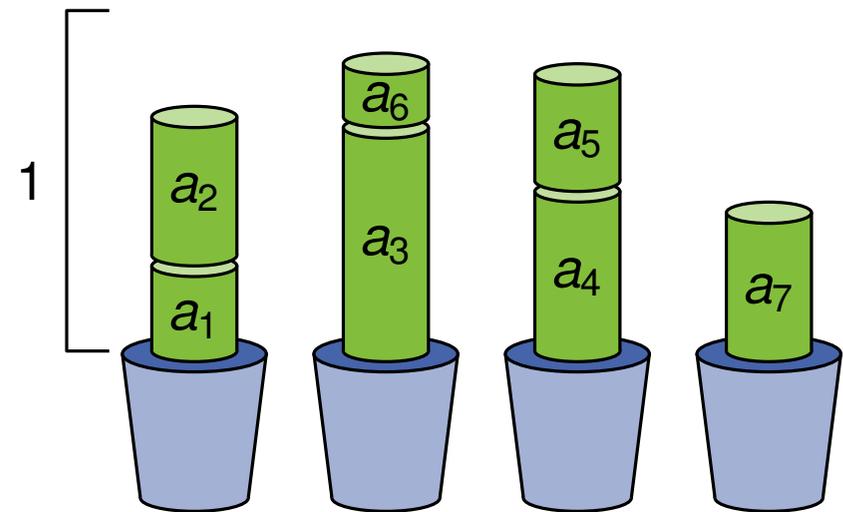
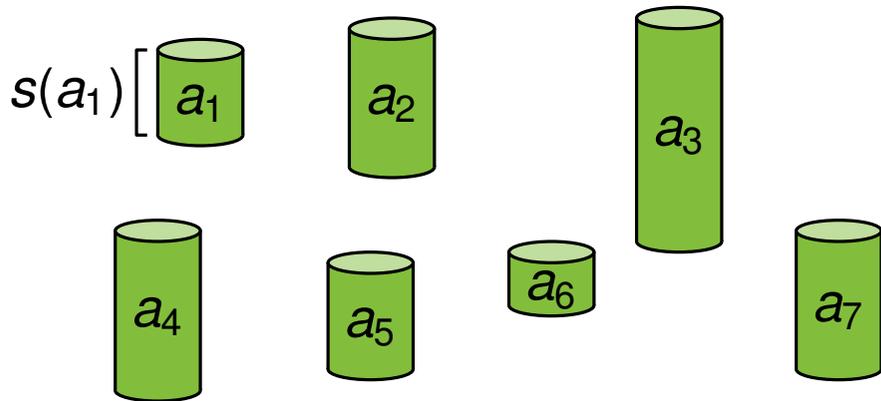
Weise die Elemente in M einer minimalen Anzahl an Bins B_1, \dots, B_m zu, sodass für jeden Bin B gilt:

$$\sum_{a_i \in B} s(a_i) \leq 1$$

BIN PACKING ist \mathcal{NP} -schwer.

Wiederholung: Bin Packing – Definition

endliche Menge $M = \{a_1, \dots, a_n\}$
mit Gewichtsfunktion $s: M \rightarrow (0, 1]$



Eimer (Bins) mit Fassungsvermögen 1

Schreibe für Elementgröße $s(a_i)$ auch kurz s_i .

4 Bins

Problem: BIN PACKING

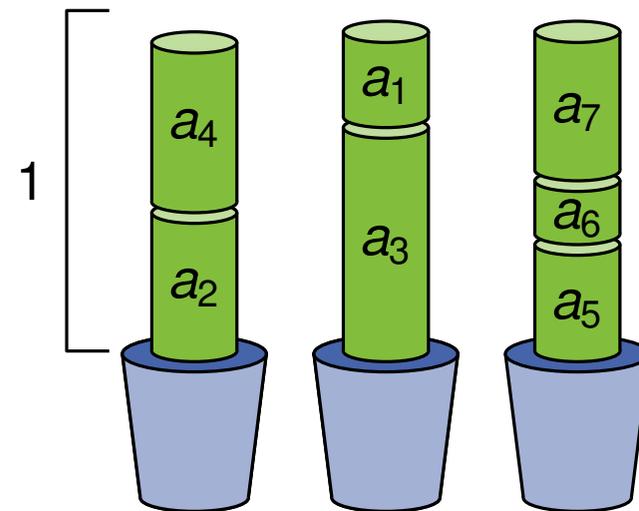
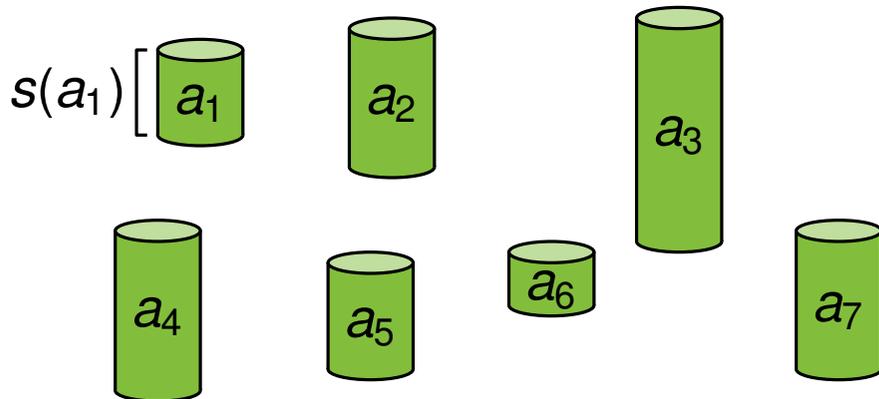
Weise die Elemente in M einer minimalen Anzahl an Bins B_1, \dots, B_m zu, sodass für jeden Bin B gilt:

$$\sum_{a_i \in B} s(a_i) \leq 1$$

BIN PACKING ist \mathcal{NP} -schwer.

Wiederholung: Bin Packing – Definition

endliche Menge $M = \{a_1, \dots, a_n\}$
mit Gewichtsfunktion $s: M \rightarrow (0, 1]$



Eimer (Bins) mit Fassungsvermögen 1

Schreibe für Elementgröße $s(a_i)$ auch kurz s_i .

3 Bins

Problem: BIN PACKING

Weise die Elemente in M einer minimalen Anzahl an Bins B_1, \dots, B_m zu, sodass für jeden Bin B gilt:

$$\sum_{a_i \in B} s(a_i) \leq 1$$

BIN PACKING ist \mathcal{NP} -schwer.

Definition: Asymptotische Approximation

Sei \mathcal{A} ein Algorithmus. Definiere die *asymptotische Gütegarantie* $\mathcal{R}_{\mathcal{A}}^{\infty}$ wie folgt:

$$\mathcal{R}_{\mathcal{A}}^{\infty} := \inf \{ r \geq 1 \mid \text{Es gibt } N > 0, \text{ sodass } \mathcal{R}_{\mathcal{A}}(I) \leq r \text{ für alle } I \text{ mit } \text{OPT}(I) \geq N \}$$

Falls $\mathcal{R}_{\mathcal{A}}^{\infty} \leq 1 + \varepsilon$, so ist \mathcal{A} ein *asymptotisch ε -approximativer* Algorithmus.

Beispiel:

Sei \mathcal{A} ein Algorithmus mit $\mathcal{A}(I) \leq \frac{3}{2} \cdot \text{OPT}(I) + 1$ für alle Instanzen I .

$\Rightarrow \mathcal{A}$ hat die asymptotische Gütegarantie $\mathcal{R}_{\mathcal{A}}^{\infty} = \frac{3}{2}$.

Definition: Asymptotische Approximation

Sei \mathcal{A} ein Algorithmus. Definiere die *asymptotische Gütegarantie* $\mathcal{R}_{\mathcal{A}}^{\infty}$ wie folgt:

$$\mathcal{R}_{\mathcal{A}}^{\infty} := \inf \{ r \geq 1 \mid \text{Es gibt } N > 0, \text{ sodass } \mathcal{R}_{\mathcal{A}}(I) \leq r \text{ für alle } I \text{ mit } \text{OPT}(I) \geq N \}$$

Falls $\mathcal{R}_{\mathcal{A}}^{\infty} \leq 1 + \varepsilon$, so ist \mathcal{A} ein *asymptotisch ε -approximativer* Algorithmus.

Beispiel:

Sei \mathcal{A} ein Algorithmus mit $\mathcal{A}(I) \leq \frac{3}{2} \cdot \text{OPT}(I) + 1$ für alle Instanzen I .

$\Rightarrow \mathcal{A}$ hat die asymptotische Gütegarantie $\mathcal{R}_{\mathcal{A}}^{\infty} = \frac{3}{2}$.

Definition: Asymptotisches PAS (APAS)

(Definition 7.16)

Ein *asymptotisches PAS (APAS)* ist eine Familie von Algorithmen $\{\mathcal{A}_{\varepsilon} \mid \varepsilon > 0\}$, sodass $\mathcal{A}_{\varepsilon}$ ein asymptotisch ε -approximativer Algorithmus ist.

$\{\mathcal{A}_{\varepsilon} \mid \varepsilon > 0\}$ ist ein *asymptotisches vollpolynomiell PAS (AFPAS)*, wenn die Laufzeit von $\mathcal{A}_{\varepsilon}$ polynomiell in $\frac{1}{\varepsilon}$ ist.

Unser Ziel:

Finde APAS $\{\mathcal{A}_\varepsilon \mid \varepsilon > 0\}$, sodass für alle Instanzen I von BIN PACKING gilt:

$$\mathcal{A}_\varepsilon(I) \leq (1 + \varepsilon) \cdot \text{OPT}(I) + 1$$

Damit kommt man beliebig nahe an die optimale Lösung heran, bis auf einen zusätzlichen Bin (daher ein APAS und kein PAS).

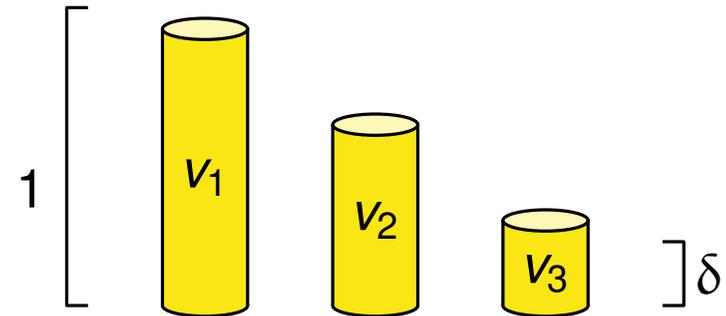
Strategie: Benutze folgende Techniken

- Restriktion von BIN PACKING
- Entfernen kleiner Elemente
- Lineares Gruppieren

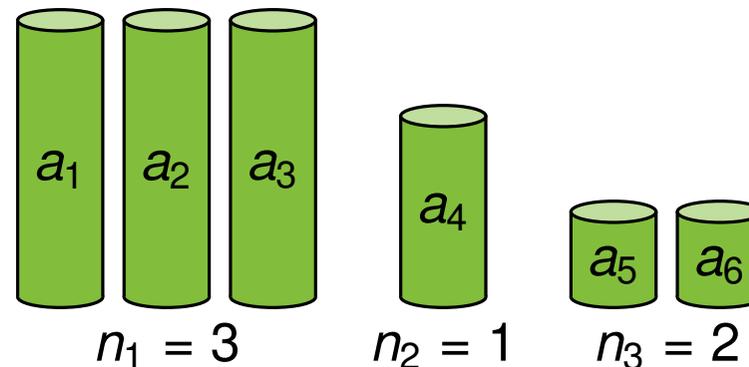
RESTRICTED BIN PACKING (RBP)

Zwei Einschränkungen:

- Die Elementgrößen s_1, \dots, s_n sind auf m erlaubte Größen $V = \{v_1, \dots, v_m\}$ eingeschränkt ($m < n$).
 - $v_1 \geq \dots \geq v_m$
- Für jede erlaubte Größe v_i gilt $v_i \geq \delta$.



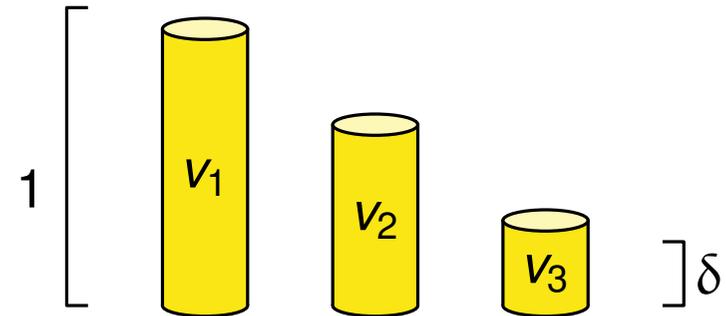
Instanz von $\text{RBP}[\delta, m]$ ist gegeben durch Anzahl n_i jeder Größe v_i .



RESTRICTED BIN PACKING (RBP)

Zwei Einschränkungen:

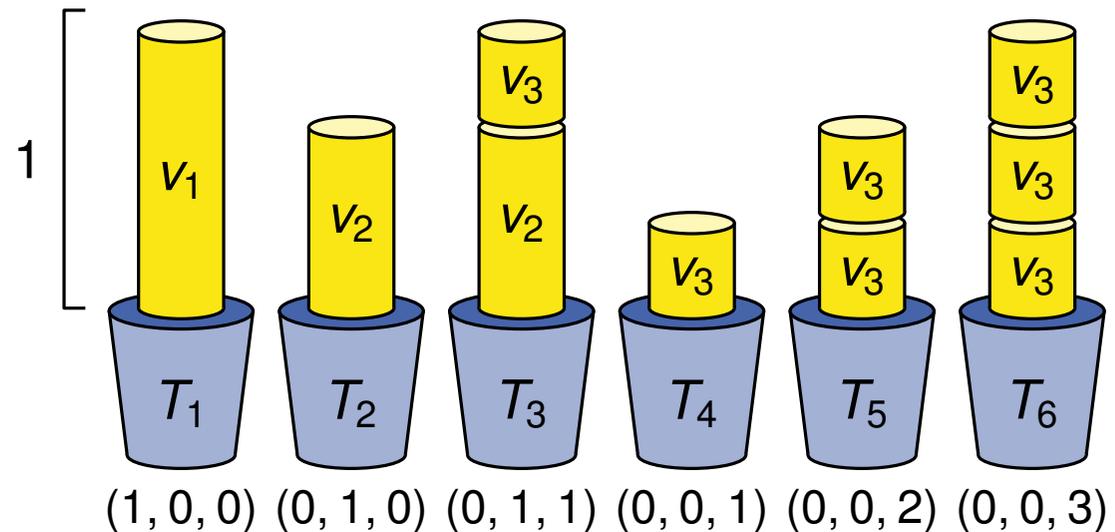
- Die Elementgrößen s_1, \dots, s_n sind auf m erlaubte Größen $V = \{v_1, \dots, v_m\}$ eingeschränkt ($m < n$).
 - $v_1 \geq \dots \geq v_m$
- Für jede erlaubte Größe v_i gilt $v_i \geq \delta$.



Bin-Typen

- Eine Zuweisung von Elementen auf ein Bin ist durch ein m -Tupel $T_t = (T_{t_1}, \dots, T_{t_m})$ gegeben.
- Ein Tupel T_t heißt *Bin-Typ*, falls

$$\sum_{j=1}^m T_{t_j} \cdot v_j \leq 1$$

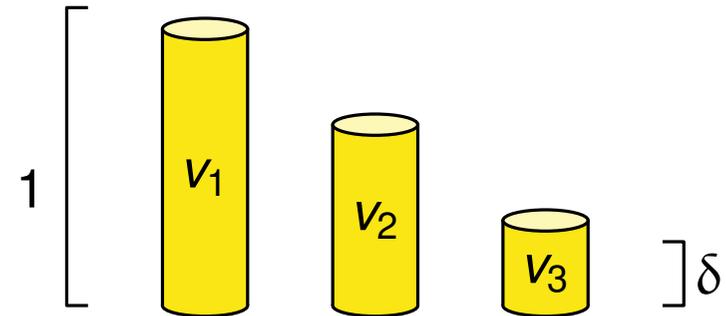


Wie viele Bin-Typen kann es geben?

RESTRICTED BIN PACKING (RBP)

Zwei Einschränkungen:

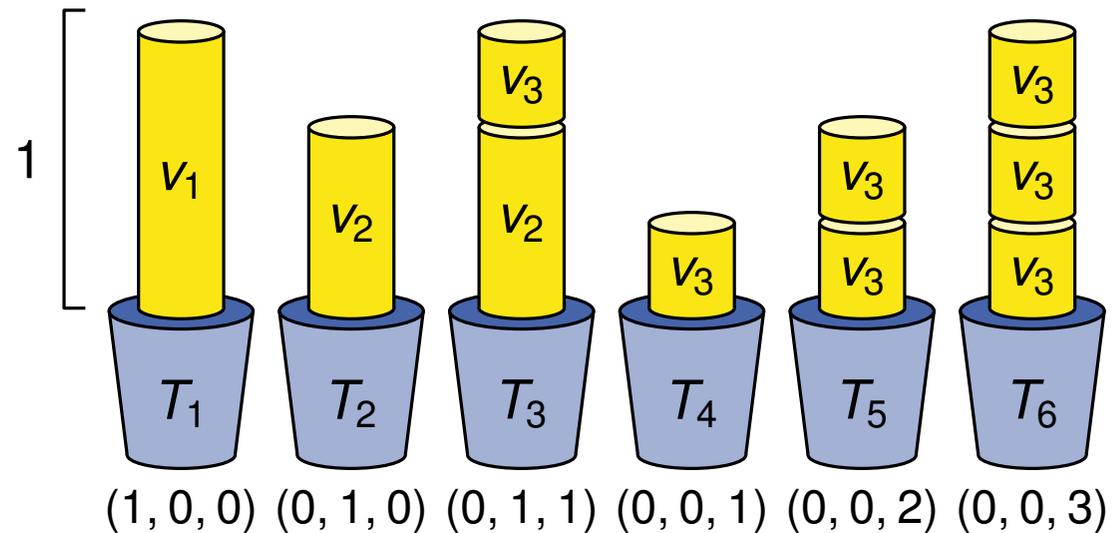
- Die Elementgrößen s_1, \dots, s_n sind auf m erlaubte Größen $V = \{v_1, \dots, v_m\}$ eingeschränkt ($m < n$).
 - $v_1 \geq \dots \geq v_m$
- Für jede erlaubte Größe v_i gilt $v_i \geq \delta$.



Bin-Typen

- Eine Zuweisung von Elementen auf ein Bin ist durch ein m -Tupel $T_t = (T_{t_1}, \dots, T_{t_m})$ gegeben.
- Ein Tupel T_t heißt *Bin-Typ*, falls

$$\sum_{j=1}^m T_{t_j} \cdot v_j \leq 1$$



Lemma: Anzahl Bin-Typen

(Lemma 7.18)

Sei $k = \lfloor \frac{1}{\delta} \rfloor$. Dann gilt für die Anzahl $q(\delta, m)$ an Bin-Typen: $q(\delta, m) \leq \binom{m+k}{k}$

Lemma: Anzahl Bin-Typen

(Lemma 7.18)

Sei $k = \lfloor \frac{1}{\delta} \rfloor$. Dann gilt für die Anzahl $q(\delta, m)$ an Bin-Typen: $q(\delta, m) \leq \binom{m+k}{k}$

Beweis:

- Keine zu kleinen Elemente ($v_i \geq \delta$) \Rightarrow jeder Bin enthält max. $k = \lfloor \frac{1}{\delta} \rfloor$ Elemente.
- Es folgt: $\sum_{j=1}^m T_{t_j} \leq k$

Wie viele Möglichkeiten gibt es, ...

äquivalent

- Anzahlen für m Elemente zu wählen, die sich zu $k' \leq k$ aufsummieren?
- Anzahlen für $m + 1$ Elemente zu wählen, die sich genau zu k aufsummieren?

Lemma: Anzahl Bin-Typen

(Lemma 7.18)

Sei $k = \lfloor \frac{1}{\delta} \rfloor$. Dann gilt für die Anzahl $q(\delta, m)$ an Bin-Typen: $q(\delta, m) \leq \binom{m+k}{k}$

Beweis:

- Keine zu kleinen Elemente ($v_i \geq \delta$) \Rightarrow jeder Bin enthält max. $k = \lfloor \frac{1}{\delta} \rfloor$ Elemente.
- Es folgt: $\sum_{j=1}^m T_{t_j} \leq k$

Wie viele Möglichkeiten gibt es, ...

äquivalent

- Anzahlen für m Elemente zu wählen, die sich zu $k' \leq k$ aufsummieren?
- Anzahlen für $m + 1$ Elemente zu wählen, die sich genau zu k aufsummieren?

Bekannt aus der Kombinatorik: $\binom{m+k}{k}$ Möglichkeiten (Kombinationen mit Wiederholung)

Wie zeigt man das?

Lemma: Anzahl Bin-Typen

(Lemma 7.18)

Sei $k = \lfloor \frac{1}{\delta} \rfloor$. Dann gilt für die Anzahl $q(\delta, m)$ an Bin-Typen: $q(\delta, m) \leq \binom{m+k}{k}$

Beweis:

- Keine zu kleinen Elemente ($v_i \geq \delta$) \Rightarrow jeder Bin enthält max. $k = \lfloor \frac{1}{\delta} \rfloor$ Elemente.
- Es folgt: $\sum_{j=1}^m T_{t_j} \leq k$

Wie viele Möglichkeiten gibt es, ...

äquivalent

- Anzahlen für m Elemente zu wählen, die sich zu $k' \leq k$ aufsummieren?
- Anzahlen für $m + 1$ Elemente zu wählen, die sich genau zu k aufsummieren?

Bekannt aus der Kombinatorik: $\binom{m+k}{k}$ Möglichkeiten (Kombinationen mit Wiederholung)

Wie zeigt man das?

Kodierte $m+1$ Anzahlen unär:

x	x		x			x	x	x			x
2	1	0	3	0	1						

Lemma: Anzahl Bin-Typen

(Lemma 7.18)

Sei $k = \lfloor \frac{1}{\delta} \rfloor$. Dann gilt für die Anzahl $q(\delta, m)$ an Bin-Typen: $q(\delta, m) \leq \binom{m+k}{k}$

Beweis:

- Keine zu kleinen Elemente ($v_i \geq \delta$) \Rightarrow jeder Bin enthält max. $k = \lfloor \frac{1}{\delta} \rfloor$ Elemente.
- Es folgt: $\sum_{j=1}^m T_{t_j} \leq k$

Wie viele Möglichkeiten gibt es, ...

äquivalent

- Anzahlen für m Elemente zu wählen, die sich zu $k' \leq k$ aufsummieren?
- Anzahlen für $m + 1$ Elemente zu wählen, die sich genau zu k aufsummieren?

Bekannt aus der Kombinatorik: $\binom{m+k}{k}$ Möglichkeiten (Kombinationen mit Wiederholung)

Wie zeigt man das?

Kodiere $m+1$ Anzahlen unär:

×	×		×			×	×	×			×
2	1	0	3	0	1						

Man wählt also aus $k + m$ Plätzen k Plätze aus, die ein \times enthalten (die restlichen m enthalten den „Trenner“ $|$)

$\Rightarrow \binom{m+k}{k}$ Möglichkeiten

Darstellung einer Lösung

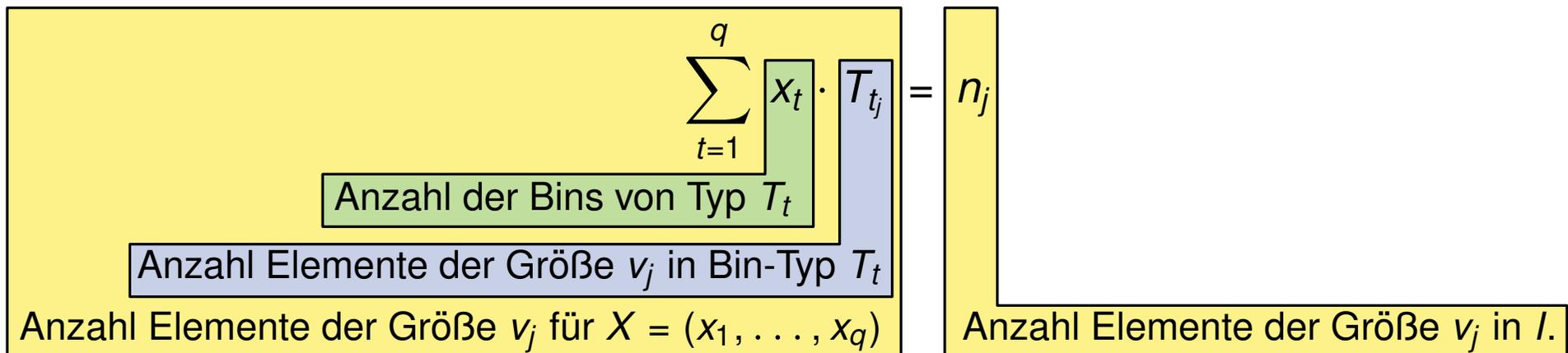
- Eine Lösung einer Instanz von $\text{RBP}[\delta, m]$ kann dadurch charakterisiert werden, wie viele Bins von jedem der $q = q(\delta, m)$ Bin-Typen vorkommen.
- \Rightarrow Lösung ist durch ein q -Tupel $X = (x_1, \dots, x_q)$ gegeben, wobei x_t die Anzahl der Bins vom Typ T_t ist.

Darstellung einer Lösung

- Eine Lösung einer Instanz von RBP[δ, m] kann dadurch charakterisiert werden, wie viele Bins von jedem der $q = q(\delta, m)$ Bin-Typen vorkommen.
- \Rightarrow Lösung ist durch ein q -Tupel $X = (x_1, \dots, x_q)$ gegeben, wobei x_t die Anzahl der Bins vom Typ T_t ist.

Formuliere RBP[δ, m]-Instanz I als ganzzahliges lineares Programm (ILP)

- Benutze die x_i im Tupel X als Variablen.
- Für jede erlaubte Größe v_j (für $j \in \{1, \dots, m\}$) stelle folgende Bedingung auf:


$$\sum_{t=1}^q x_t \cdot T_{tj} = n_j$$

Anzahl der Bins von Typ T_t

Anzahl Elemente der Größe v_j in Bin-Typ T_t

Anzahl Elemente der Größe v_j für $X = (x_1, \dots, x_q)$

n_j

Anzahl Elemente der Größe v_j in I .

- Die Anzahl Bins in der Lösung X ist gerade $\sum_{i=1}^q x_i$ (das soll minimiert werden)

Formulierung als ILP

Wie groß ist das ILP?

- Anzahl Variablen: $q(\delta, m)$.
- Anzahl Nebenbedingungen: m .

→ $q(\delta, m)$ ist exponentiell in m und $\frac{1}{\delta}$ (Erinnerung: $q(\delta, m) \leq \binom{m+k}{k}$, mit $k = \lfloor \frac{1}{\delta} \rfloor$)

Die Größe des ILP ist unabhängig von n und damit konstant, falls m und δ konstant.

Satz: Algorithmus für RBP

(Satz 7.21)

Eine Instanz I von $\text{RBP}[\delta, m]$ kann in $O(n + f(\delta, m))$ gelöst werden, wobei $f(\delta, m)$ eine Konstante ist, die nur von δ und m abhängt.

Formulierung als ILP

Wie groß ist das ILP?

- Anzahl Variablen: $q(\delta, m)$.
- Anzahl Nebenbedingungen: m .

→ $q(\delta, m)$ ist exponentiell in m und $\frac{1}{\delta}$ (Erinnerung: $q(\delta, m) \leq \binom{m+k}{k}$, mit $k = \lfloor \frac{1}{\delta} \rfloor$)

Die Größe des ILP ist unabhängig von n und damit konstant, falls m und δ konstant.

Satz: Algorithmus für RBP

(Satz 7.21)

Eine Instanz I von RBP $[\delta, m]$ kann in $O(n + f(\delta, m))$ gelöst werden, wobei $f(\delta, m)$ eine Konstante ist, die nur von δ und m abhängt.

Strategie: Benutze folgende Techniken

- Restriktion von BIN PACKING ← gerade gesehen
- Entfernen kleiner Elemente ← damit δ nicht zu klein ist
- Lineares Gruppieren ← damit m nicht zu groß ist

Lemma: Kleine Elemente

(Lemma 7.22)

Sei I eine Instanz von BIN PACKING mit einer Teillösung, die alle Elemente der Größe $s_i > \delta$ für $0 < \delta \leq \frac{1}{2}$ in β Bins packt. Die Teillösung kann zu einer Lösung von I erweitert werden, die maximal $\max\{\beta, (1 + 2 \cdot \delta) \cdot \text{OPT}(I) + 1\}$ Bins benötigt.

Beweis: Füge kleine Elemente mittels FIRST FIT ein.

- FIRST FIT fängt nur dann einen neuen Bin an, wenn das einzufügende Element in keinen existierenden Bin passt.
- **Fall 1:** Kein neuer Bin wird angefangen. \Rightarrow Es werden nur β Bins benötigt.
- **Fall 2:** Es werden $\beta' > \beta$ Bins benötigt. \Rightarrow Alle bis auf ein Bin sind mindestens zu $1 - \delta$ gefüllt. Es folgt:

$$\text{OPT}(I) \geq \text{SIZE}(I) \geq (1 - \delta) \cdot (\beta' - 1)$$

($\text{SIZE}(I)$ ist die Gesamtgröße der Elemente)

Lemma: Kleine Elemente

(Lemma 7.22)

Sei I eine Instanz von BIN PACKING mit einer Teillösung, die alle Elemente der Größe $s_i > \delta$ für $0 < \delta \leq \frac{1}{2}$ in β Bins packt. Die Teillösung kann zu einer Lösung von I erweitert werden, die maximal $\max\{\beta, (1 + 2 \cdot \delta) \cdot \text{OPT}(I) + 1\}$ Bins benötigt.

Beweis: Füge kleine Elemente mittels FIRST FIT ein.

- FIRST FIT fängt nur dann einen neuen Bin an, wenn das einzufügende Element in keinen existierenden Bin passt.
- **Fall 1:** Kein neuer Bin wird angefangen. \Rightarrow Es werden nur β Bins benötigt.
- **Fall 2:** Es werden $\beta' > \beta$ Bins benötigt. \Rightarrow Alle bis auf ein Bin sind mindestens zu $1 - \delta$ gefüllt. Es folgt:

$$\text{OPT}(I) \geq \text{SIZE}(I) \geq (1 - \delta) \cdot (\beta' - 1)$$

$$\Rightarrow \beta' \leq \frac{1}{1 - \delta} \cdot \text{OPT}(I) + 1$$

$$\leq \frac{1 + \delta - 2\delta^2}{1 - \delta} \cdot \text{OPT}(I) + 1 \quad \delta \leq \frac{1}{2} \Rightarrow 2\delta^2 \leq \delta \Rightarrow 0 \leq \delta - 2\delta^2$$

($\text{SIZE}(I)$ ist die Gesamtgröße der Elemente)

Lemma: Kleine Elemente

(Lemma 7.22)

Sei I eine Instanz von BIN PACKING mit einer Teillösung, die alle Elemente der Größe $s_i > \delta$ für $0 < \delta \leq \frac{1}{2}$ in β Bins packt. Die Teillösung kann zu einer Lösung von I erweitert werden, die maximal $\max\{\beta, (1 + 2 \cdot \delta) \cdot \text{OPT}(I) + 1\}$ Bins benötigt.

Beweis: Füge kleine Elemente mittels FIRST FIT ein.

- FIRST FIT fängt nur dann einen neuen Bin an, wenn das einzufügende Element in keinen existierenden Bin passt.
- **Fall 1:** Kein neuer Bin wird angefangen. \Rightarrow Es werden nur β Bins benötigt.
- **Fall 2:** Es werden $\beta' > \beta$ Bins benötigt. \Rightarrow Alle bis auf ein Bin sind mindestens zu $1 - \delta$ gefüllt. Es folgt:

$$\text{OPT}(I) \geq \text{SIZE}(I) \geq (1 - \delta) \cdot (\beta' - 1)$$

$$\Rightarrow \beta' \leq \frac{1}{1 - \delta} \cdot \text{OPT}(I) + 1$$

$$\leq \frac{1 + \delta - 2\delta^2}{1 - \delta} \cdot \text{OPT}(I) + 1$$

$\delta \leq \frac{1}{2} \Rightarrow 2\delta^2 \leq \delta \Rightarrow 0 \leq \delta - 2\delta^2$

$$= (1 + 2\delta) \cdot \text{OPT}(I) + 1$$

($\text{SIZE}(I)$ ist die Gesamtgröße der Elemente)

Satz: Algorithmus für RBP

(Satz 7.21)

Eine Instanz I von $\text{RBP}[\delta, m]$ kann in $O(n + f(\delta, m))$ gelöst werden, wobei $f(\delta, m)$ eine Konstante ist, die nur von δ und m abhängt.

Strategie: Benutze folgende Techniken

- Restriktion von BIN PACKING schon gesehen
- Entfernen kleiner Elemente ← damit δ nicht zu klein ist
- Lineares Gruppieren ← damit m nicht zu groß ist das kommt jetzt

Lineares Gruppieren

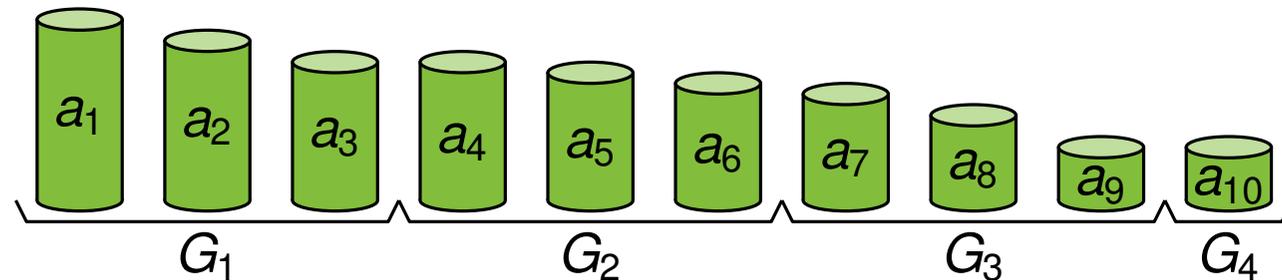
Instanz I von BIN PACKING mit Elementen der Größe $1 \geq s_1 \geq \dots \geq s_n \geq \delta > 0$

Für ein $k < n$ sei $m = \lfloor \frac{n}{k} \rfloor$. Definiere $m + 1$ Gruppen G_1, \dots, G_{m+1} , sodass G_1 die k größten Elemente, G_2 die k Nächstgrößten, usw. enthält.

$$n = 10$$

$$k = 3$$

$$m = 3$$



Lineares Gruppieren

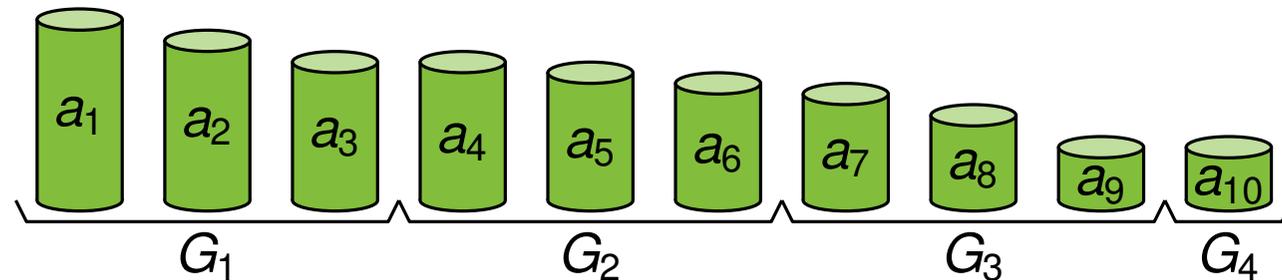
Instanz I von BIN PACKING mit Elementen der Größe $1 \geq s_1 \geq \dots \geq s_n \geq \delta > 0$

Für ein $k < n$ sei $m = \lfloor \frac{n}{k} \rfloor$. Definiere $m + 1$ Gruppen G_1, \dots, G_{m+1} , sodass G_1 die k größten Elemente, G_2 die k Nächstgrößten, usw. enthält.

$$n = 10$$

$$k = 3$$

$$m = 3$$



Definition: Dominanz

(Definition 7.23)

Man definiert zu zwei Instanzen I_1 und I_2 von BIN PACKING mit

I_1 enthält Elemente der Größe $x_1 \geq x_2 \geq \dots \geq x_n$

I_2 enthält Elemente der Größe $y_1 \geq y_2 \geq \dots \geq y_n$

die Relation „ \geq “ durch $I_1 \geq I_2 \Leftrightarrow x_i \geq y_i$ für alle $i \in \{1, \dots, n\}$.

Man sagt dann I_1 *dominiert* I_2 .

Lineares Gruppieren

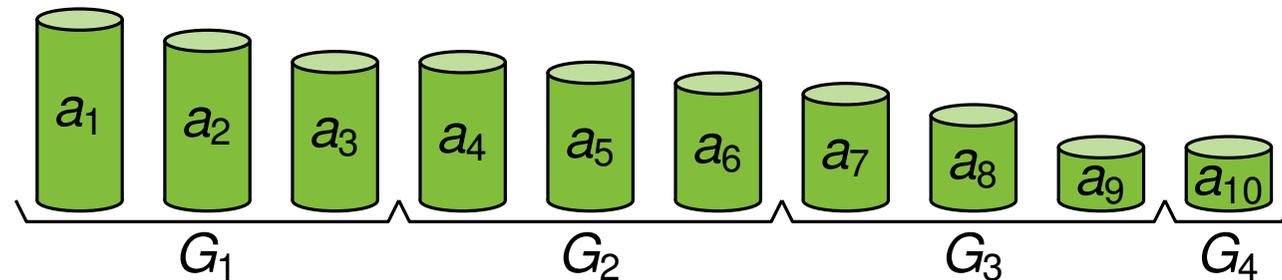
Instanz I von BIN PACKING mit Elementen der Größe $1 \geq s_1 \geq \dots \geq s_n \geq \delta > 0$

Für ein $k < n$ sei $m = \lfloor \frac{n}{k} \rfloor$. Definiere $m + 1$ Gruppen G_1, \dots, G_{m+1} , sodass G_1 die k größten Elemente, G_2 die k Nächstgrößten, usw. enthält.

$$n = 10$$

$$k = 3$$

$$m = 3$$



Definition: Dominanz

(Definition 7.23)

Man definiert zu zwei Instanzen I_1 und I_2 von BIN PACKING mit

I_1 enthält Elemente der Größe $x_1 \geq x_2 \geq \dots \geq x_n$

I_2 enthält Elemente der Größe $y_1 \geq y_2 \geq \dots \geq y_n$

die Relation „ \geq “ durch $I_1 \geq I_2 \Leftrightarrow x_i \geq y_i$ für alle $i \in \{1, \dots, n\}$.

Man sagt dann I_1 *dominiert* I_2 .

Folgerung:

(Folgerung 7.24)

Falls $I_1 \geq I_2$, so gilt $\text{SIZE}(I_1) \geq \text{SIZE}(I_2)$ und $\text{OPT}(I_1) \geq \text{OPT}(I_2)$.

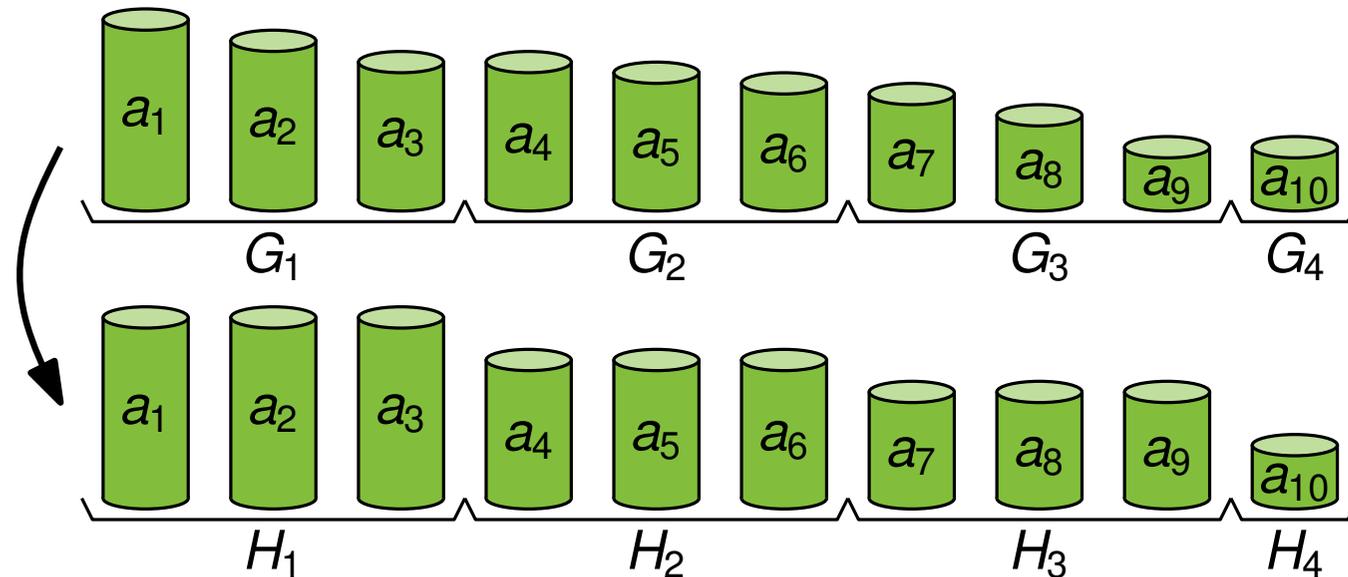
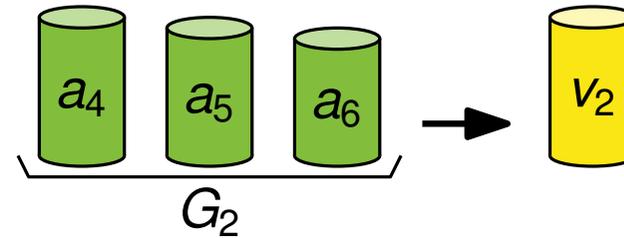
Offensichtlich: $G_1 \geq G_2 \geq \dots \geq G_m$

($\text{SIZE}(I)$ ist die Gesamtgröße der Elemente)

Lineares Gruppieren

Definition neuer Instanzen:

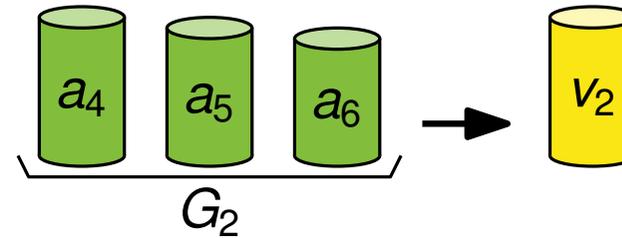
- Sei v_j das größte Element in G_j .
- Für G_j definiere H_j durch Erhöhung aller Elemente in G_j auf v_j .



Lineares Gruppieren

Definition neuer Instanzen:

- Sei v_j das größte Element in G_j .
- Für G_j definiere H_j durch Erhöhung aller Elemente in G_j auf v_j .

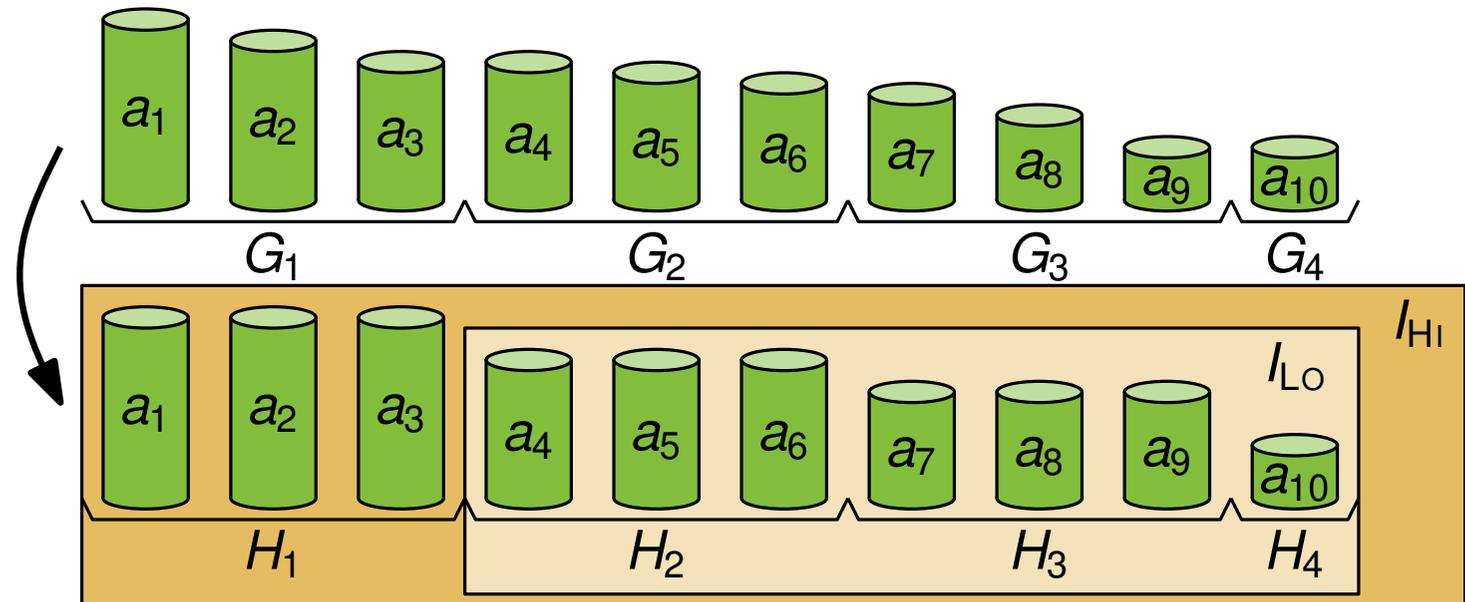


Instanz I_{Lo} :

$$H_2 \cup \dots \cup H_{m+1}$$

Instanz I_{Hi} :

$$H_1 \cup \dots \cup H_{m+1}$$



Lemma: Einordnung der Originalinstanz I

(Lemma 7.25)

$$\text{OPT}(I_{Lo}) \leq \text{OPT}(I) \leq \text{OPT}(I_{Hi}) \leq \text{OPT}(I_{Lo}) + k$$

$$\text{SIZE}(I_{Lo}) \leq \text{SIZE}(I) \leq \text{SIZE}(I_{Hi}) \leq \text{SIZE}(I_{Lo}) + k$$

Lemma: Einordnung der Originalinstanz I

(Lemma 7.25)

$$\begin{aligned} \text{OPT}(I_{L_0}) \leq \text{OPT}(I) &\leq \text{OPT}(I_{H_1}) \leq \text{OPT}(I_{L_0}) + k \\ \text{SIZE}(I_{L_0}) \leq \text{SIZE}(I) &\leq \text{SIZE}(I_{H_1}) \leq \text{SIZE}(I_{L_0}) + k \end{aligned}$$

Beweis:

$$H_1 \geq G_1 \geq H_2 \geq G_2 \cdots \geq H_{m+1} \geq G_{m+1}$$

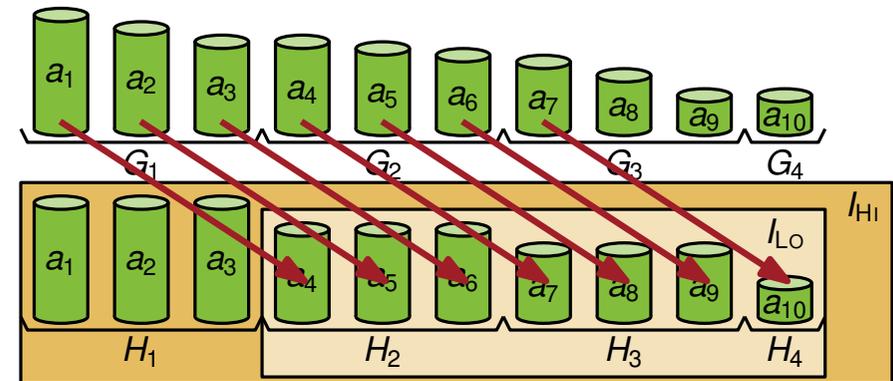
- $I_{L_0} \leq I$

- $I \leq I_{H_1}$

- Letzte Ungleichung gilt ebenfalls, denn:

- Die k Elemente in H_1 haben Gesamtgröße maximal k .

- Man benötigt maximal k zusätzlichen Bins um sie in einer Lösung von I_{L_0} einzufügen.



Lemma: Einordnung der Originalinstanz I

(Lemma 7.25)

$$\text{OPT}(I_{L_0}) \leq \text{OPT}(I) \leq \text{OPT}(I_{H_1}) \leq \text{OPT}(I_{L_0}) + k$$

$$\text{SIZE}(I_{L_0}) \leq \text{SIZE}(I) \leq \text{SIZE}(I_{H_1}) \leq \text{SIZE}(I_{L_0}) + k$$

Beweis:

$$H_1 \geq G_1 \geq H_2 \geq G_2 \cdots \geq H_{m+1} \geq G_{m+1}$$

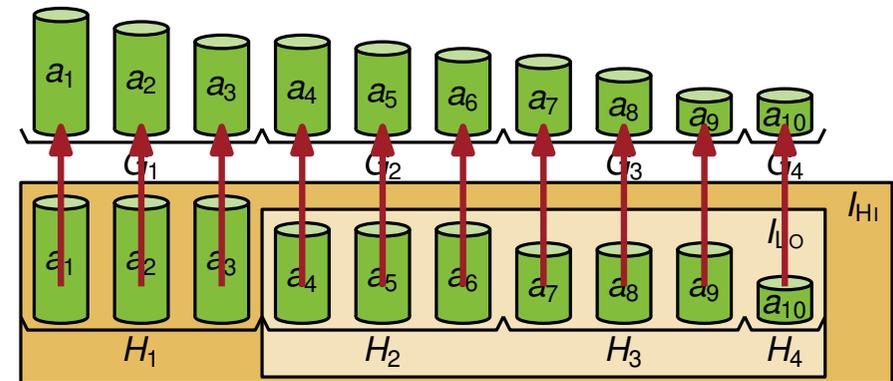
- $I_{L_0} \leq I$

- $I \leq I_{H_1}$

- Letzte Ungleichung gilt ebenfalls, denn:

- Die k Elemente in H_1 haben Gesamtgröße maximal k .

- Man benötigt maximal k zusätzlichen Bins um sie in einer Lösung von I_{L_0} einzufügen.



Lemma: Einordnung der Originalinstanz I

(Lemma 7.25)

$$\text{OPT}(I_{L_0}) \leq \text{OPT}(I) \leq \text{OPT}(I_{H_1}) \leq \text{OPT}(I_{L_0}) + k$$

$$\text{SIZE}(I_{L_0}) \leq \text{SIZE}(I) \leq \text{SIZE}(I_{H_1}) \leq \text{SIZE}(I_{L_0}) + k$$

Beweis:

$$H_1 \geq G_1 \geq H_2 \geq G_2 \cdots \geq H_{m+1} \geq G_{m+1}$$

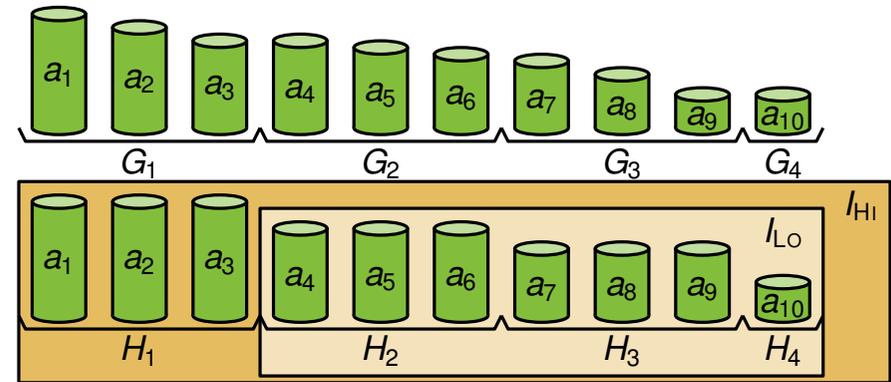
- $I_{L_0} \leq I$

- $I \leq I_{H_1}$

- Letzte Ungleichung gilt ebenfalls, denn:

- Die k Elemente in H_1 haben Gesamtgröße maximal k .

- Man benötigt maximal k zusätzlichen Bins um sie in einer Lösung von I_{L_0} einzufügen.



Lemma: Einordnung der Originalinstanz I

(Lemma 7.25)

$$\text{OPT}(I_{L_0}) \leq \text{OPT}(I) \leq \text{OPT}(I_{H_1}) \leq \text{OPT}(I_{L_0}) + k$$

$$\text{SIZE}(I_{L_0}) \leq \text{SIZE}(I) \leq \text{SIZE}(I_{H_1}) \leq \text{SIZE}(I_{L_0}) + k$$

Beweis:

$$H_1 \geq G_1 \geq H_2 \geq G_2 \cdots \geq H_{m+1} \geq G_{m+1}$$

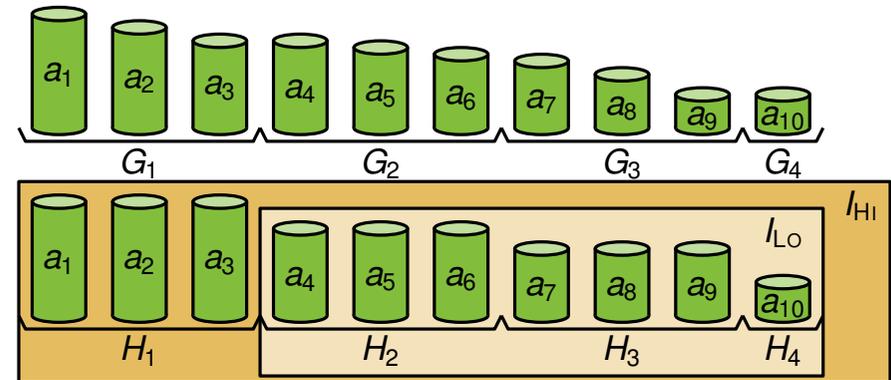
- $I_{L_0} \leq I$

- $I \leq I_{H_1}$

- Letzte Ungleichung gilt ebenfalls, denn:

- Die k Elemente in H_1 haben Gesamtgröße maximal k .

- Man benötigt maximal k zusätzlichen Bins um sie in einer Lösung von I_{L_0} einzufügen.



Folgerung:

Man kann Lösung für I mit maximal $\text{OPT}(I) + k$ Bins finden, indem man I_{L_0} löst.

Beachte: I_{L_0} ist Instanz von $\text{RBP}[\delta, m]$.

Außerdem kann I_{L_0} in $O(n \log n)$ Zeit aus I berechnet werden (Sortieren).

APAS $\mathcal{A}_\varepsilon(I = (s_1 \geq \dots \geq s_n))$

$$\delta \leftarrow \frac{\varepsilon}{2}$$

$J \leftarrow$ Instanz von RBP $[\delta, n']$ bestehend aus allen Elementen in I der Größe $\geq \delta$

$$k \leftarrow \left\lceil \frac{\varepsilon^2}{2} \cdot n' \right\rceil$$

$(H_1, \dots, H_{m+1}) \leftarrow$ Gruppen der vergrößerten Elemente, wobei $m = \left\lfloor \frac{n'}{k} \right\rfloor$

$J_{LO} \leftarrow$ Instanz von RBP $[\delta, m]$ bestehend aus H_2, \dots, H_{m+1}

$J_{HI} \leftarrow$ Instanz von BIN PACKING bestehend aus H_1, \dots, H_{m+1}

Berechne optimale Lösung von J_{LO} (mittels ILP)

Füge die k Elemente aus H_1 in maximal k zusätzliche Bins ein \rightarrow Lösung von J_{HI}

Berechne daraus Lösung für J ohne zusätzliche Bins

Erweitere Lösung von J mittels FIRST FIT zu Lösung von I .

Satz: APAS

(Satz 7.26)

Für den Algorithmus \mathcal{A}_ε gilt: $\mathcal{A}_\varepsilon(I) \leq (1 + \varepsilon) \cdot \text{OPT}(I) + 1$

\mathcal{A}_ε hat Laufzeit $O(n \cdot \log n + c_\varepsilon)$, wobei c_ε eine „Konstante“ ist, die von ε abhängt.

Satz: APAS

(Satz 7.26)

Für den Algorithmus \mathcal{A}_ε gilt: $\mathcal{A}_\varepsilon(I) \leq (1 + \varepsilon) \cdot \text{OPT}(I) + 1$

\mathcal{A}_ε hat Laufzeit $O(n \cdot \log n + c_\varepsilon)$, wobei c_ε eine „Konstante“ ist, die von ε abhängt.

Beweis Approximation:

Übersicht des Algorithmus:

$J \leftarrow$ Instanz von RBP[δ, n'] mit Elementen der Größe $\geq \delta = \frac{\varepsilon}{2}$

Gruppieren J in $m = \left\lfloor \frac{n'}{k} \right\rfloor$ Gruppen der Größe $k = \left\lceil \frac{\varepsilon^2}{2} \cdot n' \right\rceil$

Berechne optimale Lösung von J_{L_0} (mittels ILP)

Erweitere Lösung von J_{L_0} zu Lösung von J_{H_1} zu Lösung von J

Füge kleine Elemente mit FIRST FIT ein \rightarrow Lösung von I

Lemma: Einordnung der Originalinstanz I (Lemma 7.25)

$$\text{OPT}(I_{L_0}) \leq \text{OPT}(I) \leq \text{OPT}(I_{H_1}) \leq \text{OPT}(I_{L_0}) + k$$

$$\text{SIZE}(I_{L_0}) \leq \text{SIZE}(I) \leq \text{SIZE}(I_{H_1}) \leq \text{SIZE}(I_{L_0}) + k$$

Lemma: Kleine Elemente (Lemma 7.22)

Sei I eine Instanz von BIN PACKING mit einer Teillösung, die alle Elemente der Größe $s_i > \delta$ für $0 < \delta \leq \frac{1}{2}$ in β Bins packt. Die Teillösung kann zu einer Lösung von I erweitert werden, die maximal $\max\{\beta, (1 + 2 \cdot \delta) \cdot \text{OPT}(I) + 1\}$ Bins benötigt.

Satz: APAS

(Satz 7.26)

Für den Algorithmus \mathcal{A}_ε gilt: $\mathcal{A}_\varepsilon(I) \leq (1 + \varepsilon) \cdot \text{OPT}(I) + 1$

\mathcal{A}_ε hat Laufzeit $O(n \cdot \log n + c_\varepsilon)$, wobei c_ε eine „Konstante“ ist, die von ε abhängt.

Beweis Approximation:

■ Lösung von J hat max. $\text{OPT}(J_{L_0}) + k$ Bins.

■ Elemente in J haben Größe mind. $\delta = \frac{\varepsilon}{2}$

$$\Rightarrow \frac{\varepsilon}{2} \cdot n' \leq \text{SIZE}(J)$$

$$\Rightarrow k \leq \frac{\varepsilon^2}{2} \cdot n' + 1 \leq \varepsilon \cdot \text{SIZE}(J) + 1 \leq \varepsilon \cdot \text{OPT}(J) + 1$$

Übersicht des Algorithmus:

$J \leftarrow$ Instanz von RBP[δ, n'] mit Elementen der Größe $\geq \delta = \frac{\varepsilon}{2}$

Gruppieren J in $m = \lfloor \frac{n'}{k} \rfloor$ Gruppen der Größe $k = \lceil \frac{\varepsilon^2}{2} \cdot n' \rceil$

Berechne optimale Lösung von J_{L_0} (mittels ILP)

Erweitere Lösung von J_{L_0} zu Lösung von J_{H_1} zu Lösung von J

Füge kleine Elemente mit FIRST FIT ein \rightarrow Lösung von I

Lemma: Einordnung der Originalinstanz I (Lemma 7.25)

$$\text{OPT}(I_{L_0}) \leq \text{OPT}(I) \leq \text{OPT}(I_{H_1}) \leq \text{OPT}(I_{L_0}) + k$$

$$\text{SIZE}(I_{L_0}) \leq \text{SIZE}(I) \leq \text{SIZE}(I_{H_1}) \leq \text{SIZE}(I_{L_0}) + k$$

Lemma: Kleine Elemente (Lemma 7.22)

Sei I eine Instanz von BIN PACKING mit einer Teillösung, die alle Elemente der Größe $s_i > \delta$ für $0 < \delta \leq \frac{1}{2}$ in β Bins packt. Die Teillösung kann zu einer Lösung von I erweitert werden, die maximal $\max\{\beta, (1 + 2 \cdot \delta) \cdot \text{OPT}(I) + 1\}$ Bins benötigt.

Satz: APAS

(Satz 7.26)

Für den Algorithmus \mathcal{A}_ε gilt: $\mathcal{A}_\varepsilon(I) \leq (1 + \varepsilon) \cdot \text{OPT}(I) + 1$

\mathcal{A}_ε hat Laufzeit $O(n \cdot \log n + c_\varepsilon)$, wobei c_ε eine „Konstante“ ist, die von ε abhängt.

Beweis Approximation:

■ Lösung von J hat max. $\text{OPT}(J_{L_0}) + k$ Bins.

■ Elemente in J haben Größe mind. $\delta = \frac{\varepsilon}{2}$

$$\Rightarrow \frac{\varepsilon}{2} \cdot n' \leq \text{SIZE}(J)$$

$$\Rightarrow k \leq \frac{\varepsilon^2}{2} \cdot n' + 1 \leq \varepsilon \cdot \text{SIZE}(J) + 1 \leq \varepsilon \cdot \text{OPT}(J) + 1$$

■ Es folgt:

$$\begin{aligned} \text{OPT}(J_{L_0}) + k &\leq \text{OPT}(J) + \varepsilon \cdot \text{OPT}(J) + 1 \\ &= (1 + \varepsilon) \cdot \text{OPT}(J) + 1 \end{aligned}$$

■ Lösung von J hat max. $(1 + \varepsilon) \cdot \text{OPT}(J) + 1$ Bins.

Übersicht des Algorithmus:

$J \leftarrow$ Instanz von RBP[δ, n'] mit Elementen der Größe $\geq \delta = \frac{\varepsilon}{2}$

Gruppier J in $m = \lfloor \frac{n'}{k} \rfloor$ Gruppen der Größe $k = \lfloor \frac{\varepsilon^2}{2} \cdot n' \rfloor$

Berechne optimale Lösung von J_{L_0} (mittels ILP)

Erweitere Lösung von J_{L_0} zu Lösung von J_{H_1} zu Lösung von J

Füge kleine Elemente mit FIRST FIT ein \rightarrow Lösung von I

Lemma: Einordnung der Originalinstanz I (Lemma 7.25)

$$\text{OPT}(I_{L_0}) \leq \text{OPT}(I) \leq \text{OPT}(I_{H_1}) \leq \text{OPT}(I_{L_0}) + k$$

$$\text{SIZE}(I_{L_0}) \leq \text{SIZE}(I) \leq \text{SIZE}(I_{H_1}) \leq \text{SIZE}(I_{L_0}) + k$$

Lemma: Kleine Elemente (Lemma 7.22)

Sei I eine Instanz von BIN PACKING mit einer Teillösung, die alle Elemente der Größe $s_i > \delta$ für $0 < \delta \leq \frac{1}{2}$ in β Bins packt. Die Teillösung kann zu einer Lösung von I erweitert werden, die maximal $\max\{\beta, (1 + 2 \cdot \delta) \cdot \text{OPT}(I) + 1\}$ Bins benötigt.

Satz: APAS

(Satz 7.26)

Für den Algorithmus \mathcal{A}_ε gilt: $\mathcal{A}_\varepsilon(I) \leq (1 + \varepsilon) \cdot \text{OPT}(I) + 1$

\mathcal{A}_ε hat Laufzeit $O(n \cdot \log n + c_\varepsilon)$, wobei c_ε eine „Konstante“ ist, die von ε abhängt.

Beweis Approximation:

- Lösung von J hat max. $(1 + \varepsilon) \cdot \text{OPT}(J) + 1$ Bins.

Übersicht des Algorithmus:

$J \leftarrow$ Instanz von RBP[δ, n'] mit Elementen der Größe $\geq \delta = \frac{\varepsilon}{2}$

Gruppieren J in $m = \lfloor \frac{n'}{k} \rfloor$ Gruppen der Größe $k = \lceil \frac{\varepsilon^2}{2} \cdot n' \rceil$

Berechne optimale Lösung von J_{L_0} (mittels ILP)

Erweitere Lösung von J_{L_0} zu Lösung von J_{H_1} zu Lösung von J

Füge kleine Elemente mit FIRST FIT ein \rightarrow Lösung von I

Lemma: Einordnung der Originalinstanz I (Lemma 7.25)

$$\text{OPT}(I_{L_0}) \leq \text{OPT}(I) \leq \text{OPT}(I_{H_1}) \leq \text{OPT}(I_{L_0}) + k$$

$$\text{SIZE}(I_{L_0}) \leq \text{SIZE}(I) \leq \text{SIZE}(I_{H_1}) \leq \text{SIZE}(I_{L_0}) + k$$

Lemma: Kleine Elemente (Lemma 7.22)

Sei I eine Instanz von BIN PACKING mit einer Teillösung, die alle Elemente der Größe $s_i > \delta$ für $0 < \delta \leq \frac{1}{2}$ in β Bins packt. Die Teillösung kann zu einer Lösung von I erweitert werden, die maximal $\max\{\beta, (1 + 2 \cdot \delta) \cdot \text{OPT}(I) + 1\}$ Bins benötigt.

Satz: APAS

(Satz 7.26)

Für den Algorithmus \mathcal{A}_ε gilt: $\mathcal{A}_\varepsilon(I) \leq (1 + \varepsilon) \cdot \text{OPT}(I) + 1$

\mathcal{A}_ε hat Laufzeit $O(n \cdot \log n + c_\varepsilon)$, wobei c_ε eine „Konstante“ ist, die von ε abhängt.

Beweis Approximation:

- Lösung von J hat max. $(1 + \varepsilon) \cdot \text{OPT}(J) + 1$ Bins. $\geq \beta$

- Einfügen der kleinen Elemente ergibt:

$$\begin{aligned} \mathcal{A}_\varepsilon(I) &\leq \max\{\beta, (1 + 2 \cdot \delta) \cdot \text{OPT}(I) + 1\} \\ &\leq \max\{(1 + \varepsilon) \cdot \text{OPT}(J) + 1, \\ &\quad (1 + \varepsilon) \cdot \text{OPT}(I) + 1\} \end{aligned}$$

- Offensichtlich gilt: $\text{OPT}(J) \leq \text{OPT}(I)$

- Und damit: $\mathcal{A}_\varepsilon(I) \leq (1 + \varepsilon) \cdot \text{OPT}(I) + 1$

Übersicht des Algorithmus:

$J \leftarrow$ Instanz von RBP[δ, n'] mit Elementen der Größe $\geq \delta = \frac{\varepsilon}{2}$

Gruppieren J in $m = \lfloor \frac{n'}{k} \rfloor$ Gruppen der Größe $k = \lfloor \frac{\varepsilon^2}{2} \cdot n' \rfloor$

Berechne optimale Lösung von J_{L_0} (mittels ILP)

Erweitere Lösung von J_{L_0} zu Lösung von J_{H_1} zu Lösung von J

Füge kleine Elemente mit FIRST FIT ein \rightarrow Lösung von I

Lemma: Einordnung der Originalinstanz I (Lemma 7.25)

$$\text{OPT}(I_{L_0}) \leq \text{OPT}(I) \leq \text{OPT}(I_{H_1}) \leq \text{OPT}(I_{L_0}) + k$$

$$\text{SIZE}(I_{L_0}) \leq \text{SIZE}(I) \leq \text{SIZE}(I_{H_1}) \leq \text{SIZE}(I_{L_0}) + k$$

Lemma: Kleine Elemente (Lemma 7.22)

Sei I eine Instanz von BIN PACKING mit einer Teillösung, die alle Elemente der Größe $s_i > \delta$ für $0 < \delta \leq \frac{1}{2}$ in β Bins packt. Die Teillösung kann zu einer Lösung von I erweitert werden, die maximal $\max\{\beta, (1 + 2 \cdot \delta) \cdot \text{OPT}(I) + 1\}$ Bins benötigt.

Satz: APAS

(Satz 7.26)

Für den Algorithmus \mathcal{A}_ε gilt: $\mathcal{A}_\varepsilon(I) \leq (1 + \varepsilon) \cdot \text{OPT}(I) + 1$

\mathcal{A}_ε hat Laufzeit $O(n \cdot \log n + c_\varepsilon)$, wobei c_ε eine „Konstante“ ist, die von ε abhängt.

Beweis Approximation:

- Lösung von J hat max. $(1 + \varepsilon) \cdot \text{OPT}(J) + 1$ Bins. $\geq \beta$

- Einfügen der kleinen Elemente ergibt:

$$\begin{aligned} \mathcal{A}_\varepsilon(I) &\leq \max\{\beta, (1 + 2 \cdot \delta) \cdot \text{OPT}(I) + 1\} \\ &\leq \max\{(1 + \varepsilon) \cdot \text{OPT}(J) + 1, \\ &\quad (1 + \varepsilon) \cdot \text{OPT}(I) + 1\} \end{aligned}$$

- Offensichtlich gilt: $\text{OPT}(J) \leq \text{OPT}(I)$

- Und damit: $\mathcal{A}_\varepsilon(I) \leq (1 + \varepsilon) \cdot \text{OPT}(I) + 1$

Laufzeit:

- Sortieren + Lösen von $\text{RBP}[\delta, m]$

Beachte: c_ε ist exponentiell in $\frac{1}{\varepsilon}$ (\mathcal{A}_ε ist also ein APAS aber kein AFPAS)

Übersicht des Algorithmus:

$J \leftarrow$ Instanz von $\text{RBP}[\delta, n']$ mit Elementen der Größe $\geq \delta = \frac{\varepsilon}{2}$

Gruppieren J in $m = \lfloor \frac{n'}{k} \rfloor$ Gruppen der Größe $k = \lfloor \frac{\varepsilon^2}{2} \cdot n' \rfloor$

Berechne optimale Lösung von J_{L_0} (mittels ILP)

Erweitere Lösung von J_{L_0} zu Lösung von J_{H_1} zu Lösung von J

Füge kleine Elemente mit FIRST FIT ein \rightarrow Lösung von I

Lemma: Einordnung der Originalinstanz I (Lemma 7.25)

$$\text{OPT}(I_{L_0}) \leq \text{OPT}(I) \leq \text{OPT}(I_{H_1}) \leq \text{OPT}(I_{L_0}) + k$$

$$\text{SIZE}(I_{L_0}) \leq \text{SIZE}(I) \leq \text{SIZE}(I_{H_1}) \leq \text{SIZE}(I_{L_0}) + k$$

Lemma: Kleine Elemente (Lemma 7.22)

Sei I eine Instanz von BIN PACKING mit einer Teillösung, die alle Elemente der Größe $s_i > \delta$ für $0 < \delta \leq \frac{1}{2}$ in β Bins packt. Die Teillösung kann zu einer Lösung von I erweitert werden, die maximal $\max\{\beta, (1 + 2 \cdot \delta) \cdot \text{OPT}(I) + 1\}$ Bins benötigt.

Satz: Algorithmus für RBP (Satz 7.21)

Eine Instanz I von $\text{RBP}[\delta, m]$ kann in $O(n + f(\delta, m))$ gelöst werden, wobei $f(\delta, m)$ eine Konstante ist, die nur von δ und m abhängt.

AFPAS für Bin Packing

Relaxierung des ILP

Problem: Die Laufzeit des vorgestellten Algorithmus \mathcal{A}_ε ist exponentiell in $\frac{1}{\varepsilon}$.

- Lösen von RBP[δ, m] mittels ILP ist teuer.
- Ziel: löse RBP[δ, m] schneller mittels *Relaxierung*.

Formulierung als ILP (in Matrixschreibweise):

- $X = (x_1 \dots x_q)$ (x_t ist Anzahl Bins von Typ T_t)
- $q \times m$ -Matrix A (Eintrag $a_{t,j}$ beschreibt Anzahl Elemente der Größe v_j in T_t)
- $N^T = (n_1 \dots n_m)$ (n_j beschreibt Anzahl Elemente der Größe v_j in der Eingabe)

$$\text{Minimiere } (1 \dots 1) \cdot X^T = \sum_{t=1}^q x_t$$

Unter den Bedingungen

$$X \cdot A = N \text{ und } x_t \geq 0 \text{ (für } 1 \leq t \leq q)$$

Relaxierung des ILP

Problem: Die Laufzeit des vorgestellten Algorithmus \mathcal{A}_ε ist exponentiell in $\frac{1}{\varepsilon}$.

- Lösen von RBP[δ, m] mittels ILP ist teuer.
- Ziel: löse RBP[δ, m] schneller mittels *Relaxierung*.

Formulierung als ILP (in Matrixschreibweise):

- $X = (x_1 \dots x_q)$ (x_t ist Anzahl Bins von Typ T_t)
- $q \times m$ -Matrix A (Eintrag $a_{t,j}$ beschreibt Anzahl Elemente der Größe v_j in T_t)
- $N^T = (n_1 \dots n_m)$ (n_j beschreibt Anzahl Elemente der Größe v_j in der Eingabe)

$$\text{Minimiere } (1 \dots 1) \cdot X^T = \sum_{t=1}^q x_t$$

Unter den Bedingungen

$$X \cdot A = N \text{ und } x_t \geq 0 \text{ (für } 1 \leq t \leq q)$$

Relaxierung: Weglassen der Forderung nach Ganzzahligkeit liefert ein LP.

→ Sei $\text{LIN}(I)$ der Wert einer optimalen Lösung des LP. (nicht immer ganzzahlig)

Beispiel: $x_t = 3\frac{3}{4}$ bedeutet, dass von Bin-Typ T_t 3 ganze und $\frac{3}{4}$ eines weiteren Bins verwendet werden. → kann man daraus eine „richtige“ Lösung machen?

Lemma: Lösung LP \rightarrow Lösung ILP (Lemma 7.28)

Für alle Instanzen I von RBP $[\delta, m]$ gilt:

$$\text{SIZE}(I) \leq \text{LIN}(I) \leq \text{OPT}(I) \leq \text{LIN}(I) + \frac{m+1}{2}$$

Erinnerung:

m – Anzahl Elementgrößen

δ – Mindestgröße der Elemente

$\text{SIZE}(I)$ – Gesamtgröße der Elemente

$\text{LIN}(I)$ – Optimallösung des LP

$\text{OPT}(I)$ – Optimallösung (des ILP)

Beweis:

- $\text{SIZE}(I) \leq \text{LIN}(I)$: Klar, da jeder Bin im besten Fall nur Elemente der Gesamtgröße 1 enthalten kann.
- $\text{LIN}(I) \leq \text{OPT}(I)$: Klar, da eine optimale Lösung des ILP insbesondere auch eine Lösung des LP ist.
- $\text{OPT}(I) \leq \text{LIN}(I) + \frac{m+1}{2}$: Zeige: Aus einer Lösung des LP kann mit nur $\frac{m+1}{2}$ zusätzlichen Bins eine Lösung des ILP gemacht werden.

Lemma: Lösung LP \rightarrow Lösung ILP (Lemma 7.28)

Für alle Instanzen I von RBP $[\delta, m]$ gilt:

$$\text{SIZE}(I) \leq \text{LIN}(I) \leq \text{OPT}(I) \leq \text{LIN}(I) + \frac{m+1}{2}$$

Erinnerung:

m – Anzahl Elementgrößen

δ – Mindestgröße der Elemente

$\text{SIZE}(I)$ – Gesamtgröße der Elemente

$\text{LIN}(I)$ – Optimallösung des LP

$\text{OPT}(I)$ – Optimallösung (des ILP)

Beweis: Zeige: Aus einer Lösung des LP kann mit nur $\frac{m+1}{2}$ zusätzlichen Bins eine Lösung des ILP gemacht werden.

Aus der Theorie der linearen Programmierung:

- Ohne Einschränkungen gilt $\text{rang}(A) = m$ und die ersten m Zeilen von A bilden eine Basis.
- Eine Lösung X^* mit $x_t = 0$ für $t > m$ heißt *Basislösung*.
- Es gibt immer eine Basislösung, die optimal ist (also Wert $\text{LIN}(I)$ hat).

Anschaulich formuliert:

Man kann davon ausgehen, dass die Lösung des LP nur m verschiedene Bin-Typen verwendet.

Relaxierung des ILP

Lemma: Lösung LP \rightarrow Lösung ILP (Lemma 7.28)

Für alle Instanzen I von RBP $[\delta, m]$ gilt:

$$\text{SIZE}(I) \leq \text{LIN}(I) \leq \text{OPT}(I) \leq \text{LIN}(I) + \frac{m+1}{2}$$

Erinnerung:

m – Anzahl Elementgrößen

δ – Mindestgröße der Elemente

$\text{SIZE}(I)$ – Gesamtgröße der Elemente

$\text{LIN}(I)$ – Optimallösung des LP

$\text{OPT}(I)$ – Optimallösung (des ILP)

Beweis: Zeige: Aus einer Lösung des LP kann mit nur $\frac{m+1}{2}$ zusätzlichen Bins eine Lösung des ILP gemacht werden.

Annahme: Lösung des LP verwendet nur m verschiedene Bin-Typen.

- Zerlege Instanz I in ganzzahligen Teil $I - J$ und Rest J .
(Sei $\text{LIN}_I(J / I - J)$ der Wert der resultierenden Lösung für $J / I - J$.)

Beispiel:

3 Bin-Typen:

	v_2	v_3	v_3
	v_1	v_1	v_3
			v_3
			v_2
I :	$9 \times v_1$	$8 \times v_2$	$6 \times v_3$
LP-Lösung:	$6 \frac{3}{4}$	$2 \frac{1}{4}$	$1 \frac{1}{4}$
$I - J$:	$8 \times v_1$	$7 \times v_2$	$5 \times v_3$
LP-Lösung:	6	2	1
J :	$1 \times v_1$	$1 \times v_2$	$1 \times v_3$
LP-Lösung:	$\frac{3}{4}$	$\frac{1}{4}$	$\frac{1}{4}$

Relaxierung des ILP

Lemma: Lösung LP \rightarrow Lösung ILP (Lemma 7.28)

Für alle Instanzen I von RBP $[\delta, m]$ gilt:

$$\text{SIZE}(I) \leq \text{LIN}(I) \leq \text{OPT}(I) \leq \text{LIN}(I) + \frac{m+1}{2}$$

Erinnerung:

m – Anzahl Elementgrößen

δ – Mindestgröße der Elemente

$\text{SIZE}(I)$ – Gesamtgröße der Elemente

$\text{LIN}(I)$ – Optimallösung des LP

$\text{OPT}(I)$ – Optimallösung (des ILP)

Beweis: Zeige: Aus einer Lösung des LP kann mit nur $\frac{m+1}{2}$ zusätzlichen Bins eine Lösung des ILP gemacht werden.

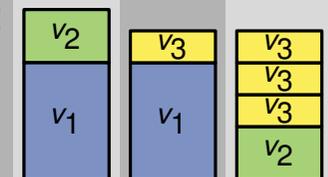
Annahme: Lösung des LP verwendet nur m verschiedene Bin-Typen.

■ Zerlege Instanz I in ganzzahligen Teil $I - J$ und Rest J .
(Sei $\text{LIN}_I(J / I - J)$ der Wert der resultierenden Lösung für $J / I - J$.)

■ Es gilt: $\text{OPT}(J) \leq m$ jeder Bin-Typ einmal („Aufrunden“)

Beispiel:

3 Bin-Typen:



I : $9 \times v_1, 8 \times v_2, 6 \times v_3$

LP-Lösung: $6 \frac{3}{4} \quad 2 \frac{1}{4} \quad 1 \frac{1}{4}$

$I - J$: $8 \times v_1, 7 \times v_2, 5 \times v_3$

LP-Lösung: $6 \quad 2 \quad 1$

J : $1 \times v_1, 1 \times v_2, 1 \times v_3$

LP-Lösung: $\frac{3}{4} \quad \frac{1}{4} \quad \frac{1}{4}$

Relaxierung des ILP

Lemma: Lösung LP \rightarrow Lösung ILP (Lemma 7.28)

Für alle Instanzen I von RBP $[\delta, m]$ gilt:

$$\text{SIZE}(I) \leq \text{LIN}(I) \leq \text{OPT}(I) \leq \text{LIN}(I) + \frac{m+1}{2}$$

Erinnerung:

m – Anzahl Elementgrößen

δ – Mindestgröße der Elemente

$\text{SIZE}(I)$ – Gesamtgröße der Elemente

$\text{LIN}(I)$ – Optimallösung des LP

$\text{OPT}(I)$ – Optimallösung (des ILP)

Beweis: Zeige: Aus einer Lösung des LP kann mit nur $\frac{m+1}{2}$ zusätzlichen Bins eine Lösung des ILP gemacht werden.

Annahme: Lösung des LP verwendet nur m verschiedene Bin-Typen.

■ Zerlege Instanz I in ganzzahligen Teil $I - J$ und Rest J .
(Sei $\text{LIN}_I(J / I - J)$ der Wert der resultierenden Lösung für $J / I - J$.)

■ Es gilt: $\text{OPT}(J) \leq m$ jeder Bin-Typ einmal („Aufrunden“)

$$\text{OPT}(J) \leq 2 \cdot \text{SIZE}(J) + 1 \quad \text{FIRST FIT}$$

Wurde im Beweis zur Approximationsgüte von FIRST FIT gezeigt (im Schnitt ist jeder Bin, bis auf einer, zumindest halb voll).

Beispiel:

3 Bin-Typen:

v_2	v_3	v_3
v_1	v_1	v_3
		v_3
		v_2

I : $9 \times v_1, 8 \times v_2, 6 \times v_3$

LP-Lösung: $6 \frac{3}{4} \quad 2 \frac{1}{4} \quad 1 \frac{1}{4}$

$I - J$: $8 \times v_1, 7 \times v_2, 5 \times v_3$

LP-Lösung: $6 \quad 2 \quad 1$

J : $1 \times v_1, 1 \times v_2, 1 \times v_3$

LP-Lösung: $\frac{3}{4} \quad \frac{1}{4} \quad \frac{1}{4}$

Relaxierung des ILP

Lemma: Lösung LP \rightarrow Lösung ILP (Lemma 7.28)

Für alle Instanzen I von RBP[δ, m] gilt:

$$\text{SIZE}(I) \leq \text{LIN}(I) \leq \text{OPT}(I) \leq \text{LIN}(I) + \frac{m+1}{2}$$

Erinnerung:

m – Anzahl Elementgrößen

δ – Mindestgröße der Elemente

$\text{SIZE}(I)$ – Gesamtgröße der Elemente

$\text{LIN}(I)$ – Optimallösung des LP

$\text{OPT}(I)$ – Optimallösung (des ILP)

Beweis: Zeige: Aus einer Lösung des LP kann mit nur $\frac{m+1}{2}$ zusätzlichen Bins eine Lösung des ILP gemacht werden.

Annahme: Lösung des LP verwendet nur m verschiedene Bin-Typen.

■ Zerlege Instanz I in ganzzahligen Teil $I - J$ und Rest J .
(Sei $\text{LIN}_I(J / I - J)$ der Wert der resultierenden Lösung für $J / I - J$.)

■ Es gilt: $\text{OPT}(J) \leq m$ jeder Bin-Typ einmal („Aufrunden“)

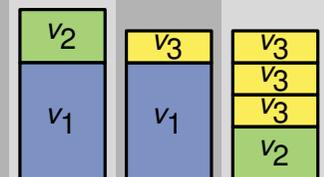
$$\text{OPT}(J) \leq 2 \cdot \text{SIZE}(J) + 1 \quad \text{FIRST FIT}$$

$$2 \cdot \text{OPT}(J) \leq m + 2 \cdot \text{SIZE}(J) + 1 \quad \text{Summe}$$

$$\text{OPT}(J) \leq \text{SIZE}(J) + \frac{m+1}{2} \quad \text{umstellen}$$

Beispiel:

3 Bin-Typen:



I : $9 \times v_1, 8 \times v_2, 6 \times v_3$

LP-Lösung: $6 \frac{3}{4} \quad 2 \frac{1}{4} \quad 1 \frac{1}{4}$

$I - J$: $8 \times v_1, 7 \times v_2, 5 \times v_3$

LP-Lösung: $6 \quad 2 \quad 1$

J : $1 \times v_1, 1 \times v_2, 1 \times v_3$

LP-Lösung: $\frac{3}{4} \quad \frac{1}{4} \quad \frac{1}{4}$

Relaxierung des ILP

Lemma: Lösung LP \rightarrow Lösung ILP (Lemma 7.28)

Für alle Instanzen I von RBP $[\delta, m]$ gilt:

$$\text{SIZE}(I) \leq \text{LIN}(I) \leq \text{OPT}(I) \leq \text{LIN}(I) + \frac{m+1}{2}$$

Erinnerung:

m – Anzahl Elementgrößen

δ – Mindestgröße der Elemente

$\text{SIZE}(I)$ – Gesamtgröße der Elemente

$\text{LIN}(I)$ – Optimallösung des LP

$\text{OPT}(I)$ – Optimallösung (des ILP)

Beweis: Zeige: Aus einer Lösung des LP kann mit nur $\frac{m+1}{2}$ zusätzlichen Bins eine Lösung des ILP gemacht werden.

Annahme: Lösung des LP verwendet nur m verschiedene Bin-Typen.

■ Zerlege Instanz I in ganzzahligen Teil $I - J$ und Rest J .
(Sei $\text{LIN}_I(J / I - J)$ der Wert der resultierenden Lösung für $J / I - J$.)

■ Es gilt: $\text{OPT}(J) \leq \text{SIZE}(J) + \frac{m+1}{2}$

■ Zusammensetzen der Lösungen für $I - J$ und J :

$$\text{OPT}(I) \leq \text{OPT}(I - J) + \text{OPT}(J)$$

Die Kombination von Teillösungen für $I - J$ und J zu einer Lösung von I kann nicht besser sein als die optimale Lösung für I .

Beispiel:

3 Bin-Typen:

v_2	v_3	v_3
v_1	v_1	v_3
		v_3
		v_2

I : $9 \times v_1, 8 \times v_2, 6 \times v_3$

LP-Lösung: $6 \frac{3}{4} \quad 2 \frac{1}{4} \quad 1 \frac{1}{4}$

$I - J$: $8 \times v_1, 7 \times v_2, 5 \times v_3$

LP-Lösung: $6 \quad 2 \quad 1$

J : $1 \times v_1, 1 \times v_2, 1 \times v_3$

LP-Lösung: $\frac{3}{4} \quad \frac{1}{4} \quad \frac{1}{4}$

Relaxierung des ILP

Lemma: Lösung LP \rightarrow Lösung ILP (Lemma 7.28)

Für alle Instanzen I von RBP $[\delta, m]$ gilt:

$$\text{SIZE}(I) \leq \text{LIN}(I) \leq \text{OPT}(I) \leq \text{LIN}(I) + \frac{m+1}{2}$$

Erinnerung:

m – Anzahl Elementgrößen

δ – Mindestgröße der Elemente

$\text{SIZE}(I)$ – Gesamtgröße der Elemente

$\text{LIN}(I)$ – Optimallösung des LP

$\text{OPT}(I)$ – Optimallösung (des ILP)

Beweis: Zeige: Aus einer Lösung des LP kann mit nur $\frac{m+1}{2}$ zusätzlichen Bins eine Lösung des ILP gemacht werden.

Annahme: Lösung des LP verwendet nur m verschiedene Bin-Typen.

■ Zerlege Instanz I in ganzzahligen Teil $I - J$ und Rest J .
(Sei $\text{LIN}_I(J / I - J)$ der Wert der resultierenden Lösung für $J / I - J$.)

■ Es gilt: $\text{OPT}(J) \leq \text{SIZE}(J) + \frac{m+1}{2}$

■ Zusammensetzen der Lösungen für $I - J$ und J :

$$\begin{aligned} \text{OPT}(I) &\leq \text{OPT}(I - J) + \text{OPT}(J) \\ &\leq \text{LIN}_I(I - J) + \text{SIZE}(J) + \frac{m+1}{2} \end{aligned}$$

$\text{LIN}_I(I - J)$ repräsentiert gültige (ganzzahlige) Lösung für $I - J$.

Beispiel:

3 Bin-Typen:

v_2	v_3	v_3
v_1	v_1	v_3
		v_3
		v_2

I : $9 \times v_1, 8 \times v_2, 6 \times v_3$

LP-Lösung: $6 \frac{3}{4} \quad 2 \frac{1}{4} \quad 1 \frac{1}{4}$

$I - J$: $8 \times v_1, 7 \times v_2, 5 \times v_3$

LP-Lösung: $6 \quad 2 \quad 1$

J : $1 \times v_1, 1 \times v_2, 1 \times v_3$

LP-Lösung: $\frac{3}{4} \quad \frac{1}{4} \quad \frac{1}{4}$

Relaxierung des ILP

Lemma: Lösung LP \rightarrow Lösung ILP (Lemma 7.28)

Für alle Instanzen I von RBP $[\delta, m]$ gilt:

$$\text{SIZE}(I) \leq \text{LIN}(I) \leq \text{OPT}(I) \leq \text{LIN}(I) + \frac{m+1}{2}$$

Erinnerung:

m – Anzahl Elementgrößen

δ – Mindestgröße der Elemente

$\text{SIZE}(I)$ – Gesamtgröße der Elemente

$\text{LIN}(I)$ – Optimallösung des LP

$\text{OPT}(I)$ – Optimallösung (des ILP)

Beweis: Zeige: Aus einer Lösung des LP kann mit nur $\frac{m+1}{2}$ zusätzlichen Bins eine Lösung des ILP gemacht werden.

Annahme: Lösung des LP verwendet nur m verschiedene Bin-Typen.

■ Zerlege Instanz I in ganzzahligen Teil $I - J$ und Rest J .
(Sei $\text{LIN}_I(J / I - J)$ der Wert der resultierenden Lösung für $J / I - J$.)

■ Es gilt: $\text{OPT}(J) \leq \text{SIZE}(J) + \frac{m+1}{2}$

■ Zusammensetzen der Lösungen für $I - J$ und J :

$$\begin{aligned} \text{OPT}(I) &\leq \text{OPT}(I - J) + \text{OPT}(J) \\ &\leq \text{LIN}_I(I - J) + \text{SIZE}(J) + \frac{m+1}{2} \\ &\leq \text{LIN}_I(I - J) + \text{LIN}_I(J) + \frac{m+1}{2} \end{aligned}$$

Klar, da Bins mit Gesamtgröße x nur Elemente der Gesamtgröße maximal x fassen können.

Beispiel:

3 Bin-Typen:

v_2	v_3	v_3
v_1	v_1	v_3
		v_3
		v_2

I : $9 \times v_1, 8 \times v_2, 6 \times v_3$

LP-Lösung: $6 \frac{3}{4} \quad 2 \frac{1}{4} \quad 1 \frac{1}{4}$

$I - J$: $8 \times v_1, 7 \times v_2, 5 \times v_3$

LP-Lösung: $6 \quad 2 \quad 1$

J : $1 \times v_1, 1 \times v_2, 1 \times v_3$

LP-Lösung: $\frac{3}{4} \quad \frac{1}{4} \quad \frac{1}{4}$

Relaxierung des ILP

Lemma: Lösung LP \rightarrow Lösung ILP (Lemma 7.28)

Für alle Instanzen I von RBP[δ, m] gilt:

$$\text{SIZE}(I) \leq \text{LIN}(I) \leq \text{OPT}(I) \leq \text{LIN}(I) + \frac{m+1}{2}$$

Erinnerung:

m – Anzahl Elementgrößen

δ – Mindestgröße der Elemente

$\text{SIZE}(I)$ – Gesamtgröße der Elemente

$\text{LIN}(I)$ – Optimallösung des LP

$\text{OPT}(I)$ – Optimallösung (des ILP)

Beweis: Zeige: Aus einer Lösung des LP kann mit nur $\frac{m+1}{2}$ zusätzlichen Bins eine Lösung des ILP gemacht werden.

Annahme: Lösung des LP verwendet nur m verschiedene Bin-Typen.

■ Zerlege Instanz I in ganzzahligen Teil $I - J$ und Rest J .
(Sei $\text{LIN}_I(J / I - J)$ der Wert der resultierenden Lösung für $J / I - J$.)

■ Es gilt: $\text{OPT}(J) \leq \text{SIZE}(J) + \frac{m+1}{2}$

■ Zusammensetzen der Lösungen für $I - J$ und J :

$$\begin{aligned} \text{OPT}(I) &\leq \text{OPT}(I - J) + \text{OPT}(J) \\ &\leq \text{LIN}_I(I - J) + \text{SIZE}(J) + \frac{m+1}{2} \\ &\leq \text{LIN}_I(I - J) + \text{LIN}_I(J) + \frac{m+1}{2} \\ &\leq \text{LIN}(I) + \frac{m+1}{2} \end{aligned}$$

Beispiel:

3 Bin-Typen:

v_2	v_3	v_3
v_1	v_1	v_3
		v_3
		v_2

I : $9 \times v_1, 8 \times v_2, 6 \times v_3$

LP-Lösung: $6 \frac{3}{4} \quad 2 \frac{1}{4} \quad 1 \frac{1}{4}$

$I - J$: $8 \times v_1, 7 \times v_2, 5 \times v_3$

LP-Lösung: $6 \quad 2 \quad 1$

J : $1 \times v_1, 1 \times v_2, 1 \times v_3$

LP-Lösung: $\frac{3}{4} \quad \frac{1}{4} \quad \frac{1}{4}$

Lemma: Lösung LP \rightarrow Lösung ILP (Lemma 7.28)

Für alle Instanzen I von RBP $[\delta, m]$ gilt:

$$\text{SIZE}(I) \leq \text{LIN}(I) \leq \text{OPT}(I) \leq \text{LIN}(I) + \frac{m+1}{2}$$

Erinnerung:

m – Anzahl Elementgrößen

δ – Mindestgröße der Elemente

$\text{SIZE}(I)$ – Gesamtgröße der Elemente

$\text{LIN}(I)$ – Optimallösung des LP

$\text{OPT}(I)$ – Optimallösung (des ILP)

Bedeutung des Lemmas:

- Konstruktiver Beweis: Gegeben eine Lösung für das LP; man kann daraus mit $\frac{m+1}{2}$ zusätzlichen Bins eine ganzzahlige Lösung bauen.
- Problem: Anzahl der Bedingungen im LP ist (wie im ILP) exponentiell in $\frac{1}{\varepsilon}$.

Lemma: Lösung LP \rightarrow Lösung ILP (Lemma 7.28)

Für alle Instanzen I von RBP[δ, m] gilt:

$$\text{SIZE}(I) \leq \text{LIN}(I) \leq \text{OPT}(I) \leq \text{LIN}(I) + \frac{m+1}{2}$$

Erinnerung:

m – Anzahl Elementgrößen

δ – Mindestgröße der Elemente

$\text{SIZE}(I)$ – Gesamtgröße der Elemente

$\text{LIN}(I)$ – Optimallösung des LP

$\text{OPT}(I)$ – Optimallösung (des ILP)

Bedeutung des Lemmas:

- Konstruktiver Beweis: Gegeben eine Lösung für das LP; man kann daraus mit $\frac{m+1}{2}$ zusätzlichen Bins eine ganzzahlige Lösung bauen.
- Problem: Anzahl der Bedingungen im LP ist (wie im ILP) exponentiell in $\frac{1}{\varepsilon}$.

Satz (ohne Beweis): Vollpolynomieller Algorithmus (Satz 7.30)

Es gibt einen vollpolynomiellen Algorithmus \mathcal{A} zur Lösung einer Instanz von RBP[δ, m] mit

$$\mathcal{A}(I) \leq \text{LIN}(I) + \frac{m+1}{2} + 1$$

(Karmaker & Karp 1982)

Anmerkung zum Beweis:

- Der Algorithmus basiert auf einem polynomiellen Algorithmus für LPs.
- Laufzeit: $O\left(m^8 \log m \log^2\left(\frac{mn}{\delta}\right) + m^4 n \log m \log\left(\frac{mn}{\delta}\right)\right)$

APAS $\mathcal{A}_\varepsilon(I = (s_1 \geq \dots \geq s_n))$

$$\delta \leftarrow \frac{\varepsilon}{2}$$

$J \leftarrow$ Instanz von RBP $[\delta, n']$ bestehend aus allen Elementen in I der Größe $\geq \delta$

$$k \leftarrow \left\lceil \frac{\varepsilon^2}{2} \cdot n' \right\rceil$$

$(H_1, \dots, H_{m+1}) \leftarrow$ Gruppen der vergrößerten Elemente, wobei $m = \left\lfloor \frac{n'}{k} \right\rfloor$

$J_{LO} \leftarrow$ Instanz von RBP $[\delta, m]$ bestehend aus H_2, \dots, H_{m+1}

$J_{HI} \leftarrow$ Instanz von BIN PACKING bestehend aus H_1, \dots, H_{m+1}

Berechne optimale Lösung von J_{LO} (mittels ILP)

Füge die k Elemente aus H_1 in maximal k zusätzliche Bins ein \rightarrow Lösung von J_{HI}

Berechne daraus Lösung für J ohne zusätzliche Bins

Erweitere Lösung von J mittels FIRST FIT zu Lösung von I .

Satz: APAS

(Satz 7.26)

Für den Algorithmus \mathcal{A}_ε gilt:

$$\mathcal{A}_\varepsilon(I) \leq (1 + \varepsilon) \cdot \text{OPT}(I) + 1$$

~~AFPAS~~ für BIN PACKING

~~AFPAS~~

~~AFPAS~~ $\mathcal{A}_\varepsilon(I = (s_1 \geq \dots \geq s_n))$

$$\delta \leftarrow \frac{\varepsilon}{2}$$

$J \leftarrow$ Instanz von RBP $[\delta, n']$ bestehend aus allen Elementen in I der Größe $\geq \delta$

$$k \leftarrow \left\lceil \frac{\varepsilon^2}{2} \cdot n' \right\rceil$$

$(H_1, \dots, H_{m+1}) \leftarrow$ Gruppen der vergrößerten Elemente, wobei $m = \left\lfloor \frac{n'}{k} \right\rfloor$

$J_{LO} \leftarrow$ Instanz von RBP $[\delta, m]$ bestehend aus H_2, \dots, H_{m+1}

$J_{HI} \leftarrow$ Instanz von BIN PACKING bestehend aus H_1, \dots, H_{m+1}

~~Berechne optimale Lösung von J_{LO} (mittels ILP)~~

Berechne Lösung mit Wert $\leq \text{LIN}(I) + \frac{m+1}{2} + 1$ (mittels LP Relaxierung)

Füge die k Elemente aus H_1 in maximal k zusätzliche Bins ein \rightarrow Lösung von J_{HI}

Berechne daraus Lösung für J ohne zusätzliche Bins

Erweitere Lösung von J mittels FIRST FIT zu Lösung von I .

Satz: ~~AFPAS~~ AFPAS

(Satz 7.26)

32

Für den Algorithmus \mathcal{A}_ε gilt:

$$\mathcal{A}_\varepsilon(I) \leq (1 + \varepsilon) \cdot \text{OPT}(I) + 1 + \frac{1}{\varepsilon^2} + 2 \frac{1}{2}$$

Satz: ~~APAS~~ AFPAS

(Satz 7.26)

32

Für den Algorithmus A_ϵ gilt:

$$\mathcal{A}_\epsilon(I) \leq (1 + \epsilon) \cdot \text{OPT}(I) + 1 + \frac{1}{\epsilon^2} + 2\frac{1}{2}$$

Beweis:

- Analog zu Beweis von Satz 7.26.
- Lösen mittels LP Relaxierung erfordert $\frac{m+1}{2} + 1$ zusätzliche Bins.
- Es gilt also:

$$\begin{aligned} \mathcal{A}_\epsilon(I) &\leq (1 + \epsilon) \cdot \text{OPT}(I) + 1 + \frac{m+1}{2} + 1 \\ &= (1 + \epsilon) \cdot \text{OPT}(I) + \frac{m}{2} + 2\frac{1}{2} \end{aligned}$$

AFPAS

~~APAS~~ $\mathcal{A}_\epsilon(I = (s_1 \geq \dots \geq s_n))$

$\delta \leftarrow \frac{\epsilon}{2}$

$J \leftarrow$ Instanz von RBP $[\delta, n']$ bestehend aus allen Elementen in I der Größe $\geq \delta$

$k \leftarrow \left\lceil \frac{\epsilon^2}{2} \cdot n' \right\rceil$

$(H_1, \dots, H_{m+1}) \leftarrow$ Gruppen der vergrößerten Elemente, wobei $m = \left\lfloor \frac{n'}{k} \right\rfloor$

$J_{L_0} \leftarrow$ Instanz von RBP $[\delta, m]$ bestehend aus H_2, \dots, H_{m+1}

$J_{H_1} \leftarrow$ Instanz von BIN PACKING bestehend aus H_1, \dots, H_{m+1}

~~Berechne optimale Lösung von J_{L_0} (mittels ILP)~~

Berechne Lösung mit Wert $\leq \text{LIN}(I) + \frac{m+1}{2} + 1$ (mittels LP Relaxierung)

Füge die k Elemente aus H_1 in maximal k zusätzliche Bins ein \rightarrow Lösung von J_{H_1}

Berechne daraus Lösung für J ohne zusätzliche Bins

Erweitere Lösung von J mittels FIRST FIT zu Lösung von I .

Satz: ~~APAS~~ AFPAS

(Satz 7.26)

32

Für den Algorithmus A_ϵ gilt:

$$\mathcal{A}_\epsilon(I) \leq (1 + \epsilon) \cdot \text{OPT}(I) + 1 + \frac{1}{\epsilon^2} + 2\frac{1}{2}$$

Beweis:

- Analog zu Beweis von Satz 7.26.
- Lösen mittels LP Relaxierung erfordert $\frac{m+1}{2} + 1$ zusätzliche Bins.
- Es gilt also:

$$\begin{aligned} \mathcal{A}_\epsilon(I) &\leq (1 + \epsilon) \cdot \text{OPT}(I) + 1 + \frac{m+1}{2} + 1 \\ &= (1 + \epsilon) \cdot \text{OPT}(I) + \frac{m}{2} + 2\frac{1}{2} \\ &\leq (1 + \epsilon) \cdot \text{OPT}(I) + \frac{1}{\epsilon^2} + 2\frac{1}{2} \end{aligned}$$

$$m \leq \frac{n'}{k} = \frac{n'}{\left\lceil \frac{\epsilon^2}{2} \cdot n' \right\rceil} \leq \frac{2}{\epsilon^2}$$

AFPAS

~~APAS~~ $\mathcal{A}_\epsilon(I = (s_1 \geq \dots \geq s_n))$

$\delta \leftarrow \frac{\epsilon}{2}$

$J \leftarrow$ Instanz von RBP $[\delta, n']$ bestehend aus allen Elementen in I der Größe $\geq \delta$

$k \leftarrow \left\lceil \frac{\epsilon^2}{2} \cdot n' \right\rceil$

$(H_1, \dots, H_{m+1}) \leftarrow$ Gruppen der vergrößerten Elemente, wobei $m = \left\lfloor \frac{n'}{k} \right\rfloor$

$J_{LO} \leftarrow$ Instanz von RBP $[\delta, m]$ bestehend aus H_2, \dots, H_{m+1}

$J_{HI} \leftarrow$ Instanz von BIN PACKING bestehend aus H_1, \dots, H_{m+1}

~~Berechne optimale Lösung von J_{LO} (mittels ILP)~~

Berechne Lösung mit Wert $\leq \text{LIN}(I) + \frac{m+1}{2} + 1$ (mittels LP Relaxierung)

Füge die k Elemente aus H_1 in maximal k zusätzliche Bins ein \rightarrow Lösung von J_{HI}

Berechne daraus Lösung für J ohne zusätzliche Bins

Erweitere Lösung von J mittels FIRST FIT zu Lösung von I .