Theoretische Grundlagen der Informatik Übung

6. Übungstermin · 20. Dezember

Guido Brückner

INSTITUT FÜR THEORETISCHE INFORMATIK · PROF. DR. DOROTHEA WAGNER

Übersicht

Organisatorisches

- Unterschiedliche Verfahren in Vorlesung / Übung / Tutorien
- Punktegrenze Notenbonus
- Evaluationsergebnisse

Inhalt

- NP und co-NP
- Pseudopolynomielle Algorithmen
- Approximationsalgorithmen
- Ganzzahlige Programme

Aufgabe

Zeigen Sie, dass falls das Komplement eines NP-vollständigen Problems in NP liegt, dann gilt NP=co-NP.

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Aufgabe

Zeigen Sie, dass falls das Komplement eines NP-vollständigen Problems in NP liegt, dann gilt NP=co-NP.

Wiederholung:

Die Klasse **NP** ist die Menge aller Sprachen L, für die es eine nichtdeterministische Turingmaschine gibt, die L in polynomieller Zeit erkennt.

Die Klasse **co-NP** ist die Menge aller Sprachen deren Komplement in NP enthalten ist:

$$\text{co-NP} = \{L \subseteq \Sigma^\star \mid \Sigma^\star \setminus L \in \text{NP}\}$$

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Aufgabe

Zeigen Sie, dass falls das Komplement eines NP-vollständigen Problems in NP liegt, dann gilt NP=co-NP.

Sei Π_1 ein NP-vollständiges Problem und sei Π_1^c das Komplement hierzu.

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Aufgabe

Zeigen Sie, dass falls das Komplement eines NP-vollständigen Problems in NP liegt, dann gilt NP=co-NP.

Sei Π_1 ein NP-vollständiges Problem und sei Π_1^c das Komplement hierzu.

Annahme: $\Pi_1^c \in \text{NP}$

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Aufgabe

Zeigen Sie, dass falls das Komplement eines NP-vollständigen Problems in NP liegt, dann gilt NP=co-NP.

Sei Π_1 ein NP-vollständiges Problem und sei Π_1^c das Komplement hierzu.

Annahme: $\Pi_1^c \in \text{NP}$

Zeige: Komplement Π_2^c eines beliebigen Problems Π_2 liegt in NP.

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Aufgabe

Zeigen Sie, dass falls das Komplement eines NP-vollständigen Problems in NP liegt, dann gilt NP=co-NP.

Sei Π_1 ein NP-vollständiges Problem und sei Π_1^c das Komplement hierzu.

Annahme: $\Pi_1^c \in \text{NP}$

Zeige: Komplement Π_2^c eines beliebigen Problems Π_2 liegt in NP.

Π_1 ist NP-vollständig: Es gibt Reduktion φ von Π_2 auf Π_1

φ ist auch Reduktion von Π_2^c auf Π_1^c

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Aufgabe

Zeigen Sie, dass falls das Komplement eines NP-vollständigen Problems in NP liegt, dann gilt NP=co-NP.

Sei Π_1 ein NP-vollständiges Problem und sei Π_1^c das Komplement hierzu.

Annahme: $\Pi_1^c \in \text{NP}$

Zeige: Komplement Π_2^c eines beliebigen Problems Π_2 liegt in NP.

Π_1 ist NP-vollständig: Es gibt Reduktion φ von Π_2 auf Π_1

φ ist auch Reduktion von Π_2^c auf Π_1^c

Sei I_2^c eine Instanz von Π_2^c.

Transformiere die Instanz in eine Instanz von Π_1^c: $I_1^c = \varphi(I_2^c)$.

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Aufgabe

Zeigen Sie, dass falls das Komplement eines NP-vollständigen Problems in NP liegt, dann gilt NP=co-NP.

Sei Π_1 ein NP-vollständiges Problem und sei Π_1^c das Komplement hierzu.

Annahme: $\Pi_1^c \in \text{NP}$

Zeige: Komplement Π_2^c eines beliebigen Problems Π_2 liegt in NP.

Π_1 ist NP-vollständig: Es gibt Reduktion φ von Π_2 auf Π_1

φ ist auch Reduktion von Π_2^c auf Π_1^c

Sei I_2^c eine Instanz von Π_2^c.

Transformiere die Instanz in eine Instanz von Π_1^c: $I_1^c = \varphi(I_2^c)$.

Da Π_1^c in NP liegt, kann eine NTM $\mathcal{M}$ wie folgt auf I_1^c arbeiten.

1. $\mathcal{M}$ berechnet eine Lösung von I_1^c
2. $\mathcal{M}$ überprüft ob Ja-Instanz
3. Falls ja, dann auch I_2^c sonst ist I_2^c keine Ja-Instanz.

Pseudopolynomielle Algorithmen

Wiederholung

Ein Algorithmus, der Problem Π löst, heißt pseudopolynomiell, falls seine Laufzeit durch ein Polynom der beiden Variablen

- Eingabegröße und
- Größe der größten in der Eingabe vorkommenden Zahl

beschränkt ist.

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Aufgabe

Problem SUBSETSUM

Gegeben: Menge $M = \{x_1, \ldots, x_n\}$, Funktion $w : M \to \mathbb{N}_0$ und $K \in \mathbb{N}$.

Frage: Existiert Teilmenge $M' \subseteq M$ mit

$$\sum_{a \in M'} w(a) = K$$

Aufgabe: Geben Sie einen pseudopolynomiellen Algorithmus an.

Aufgabe

Problem SUBSETSUM

Gegeben: Menge $M = \{x_1, \ldots, x_n\}$, Funktion $w\colon M \to \mathbb{N}_0$ und $K \in \mathbb{N}$.

Frage: Existiert Teilmenge $M' \subseteq M$ mit

$$\sum_{a \in M'} w(a) = K$$

Aufgabe: Geben Sie einen pseudopolynomiellen Algorithmus an.

Idee Verwende 2-dimensionale Tabelle T der Größe $(n+1) \times (K+1)$, die Wahrheitswerte enthält

	0	1	2	3	4	5
	w					
x_1			f			
x_2						
x_3						

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Aufgabe

Problem SUBSETSUM

Gegeben: Menge $M = \{x_1, \ldots, x_n\}$, Funktion $w : M \to \mathbb{N}_0$ und $K \in \mathbb{N}$.

Frage: Existiert Teilmenge $M' \subseteq M$ mit

$$\sum_{a \in \mathcal{M}'} w(a) = K$$

Aufgabe: Geben Sie einen pseudopolynomiellen Algorithmus an.

Idee Verwende 2-dimensionale Tabelle T der Größe $(n+1) \times (K+1)$, die Wahrheitswerte enthält

Interpretation: $T[i, j] = w \quad \Leftrightarrow$

Es gibt Teilmenge $M' \subseteq \{x_1, \cdots, x_i\}$, sodass

$$\sum_{a \in \mathcal{M}'} w(a) = j$$

	0	1	2	3	4	5
	w					
x_1			f			
x_2						
x_3						

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Lösung

Interpretation: $T[i,j] = w \quad \Leftrightarrow$

Es gibt Teilmenge $M' \subseteq \{x_1, \cdots, x_i\}$, sodass

$$\sum_{a \in \mathcal{M}'} w(a) = j$$

	0	1	2	3	4	5
	w					
x_1			f			
x_2						
x_3						

Wie $T[i,j]$ aus vorherigen Einträgen bestimmen?

$$T[i,j] = T[i-1,j] \vee T[i-1, j-x_i]$$

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Lösung

Interpretation: $T[i,j] = w \quad \Leftrightarrow$

Es gibt Teilmenge $M' \subseteq \{x_1, \cdots, x_i\}$, sodass

$$\sum_{a \in \mathcal{M}'} w(a) = j$$

	0	1	2	3	4	5
	w					
x_1			f			
x_2						
x_3						

Wie $T[i,j]$ aus vorherigen Einträgen bestimmen?

$$T[i,j] = T[i-1,j] \vee T[i-1, j-x_i]$$

Algorithmus

Initialisiere $T[0,0]$ = **w** und $T[i,j]$ = **f** sonst

Für $i = 0, \ldots n$

Für $j = 0, \ldots K$

$T[i,j] = T[i-1,j] \vee T[i-1, j-x_i]$

Gebe $T[n,K]$ zurück.

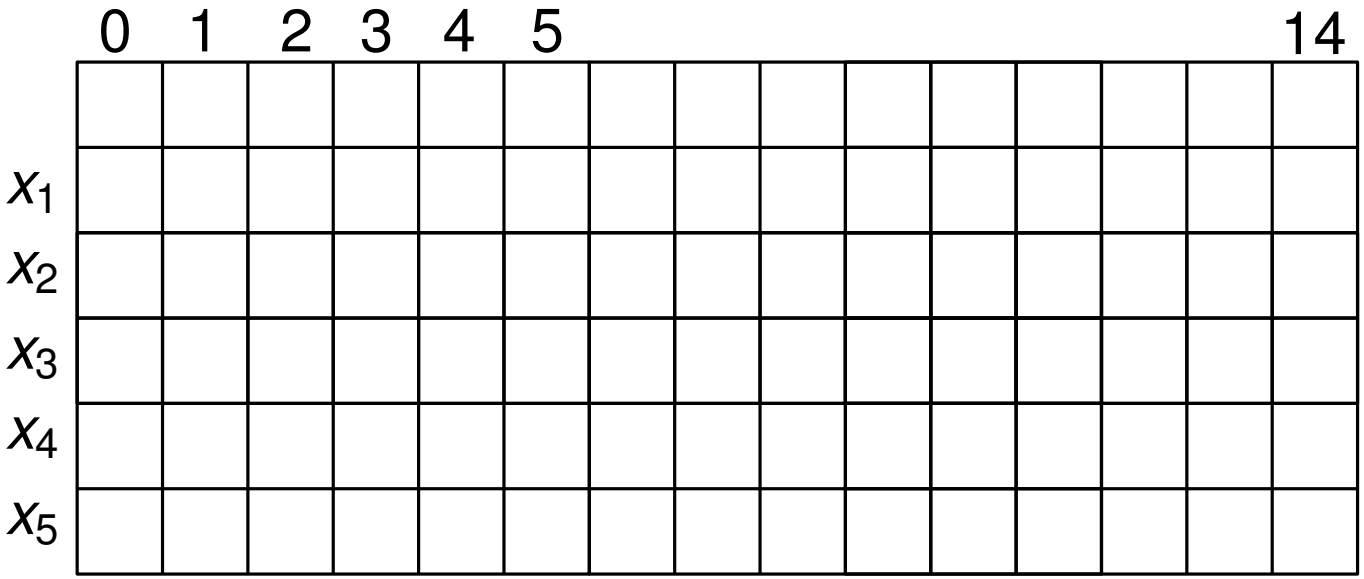
Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Beispiel

$M = \{x_1, x_2, x_3, x_4, x_5\}$

$w(x_1) = 2 \qquad w(x_2) = 3 \quad w(x_3) = 5 \quad w(x_4) = 7 \quad w(x_5) = 10$

$K = 14$



$$T[i,j] = T[i-1,j] \vee T[i-1,j-x_i]$$

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Beispiel

$M = \{x_1, x_2, x_3, x_4, x_5\}$

$w(x_1) = 2 \qquad w(x_2) = 3 \quad w(x_3) = 5 \qquad w(x_4) = 7 \quad w(x_5) = 10$

$K = 14$

	0	1	2	3	4	5									14
	w	f	f	f	f	f	f	f	f	f	f	f	f	f	f
x_1															
x_2															
x_3															
x_4															
x_5															

$$T[i,j] = T[i-1,j] \vee T[i-1,j-x_i]$$

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Beispiel

$M = \{x_1, x_2, x_3, x_4, x_5\}$

$w(x_1) = 2 \qquad w(x_2) = 3 \quad w(x_3) = 5 \quad w(x_4) = 7 \quad w(x_5) = 10$

$K = 14$

	0	1	2	3	4	5									14
	w	f	f	f	f	f	f	f	f	f	f	f	f	f	f
x_1	w	f	w	f	f	f	f	f	f	f	f	f	f	f	f
x_2															
x_3															
x_4															
x_5															

$$T[i,j] = T[i-1,j] \vee T[i-1,j-x_i]$$

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Beispiel

$M = \{x_1, x_2, x_3, x_4, x_5\}$

$w(x_1) = 2 \qquad w(x_2) = 3 \quad w(x_3) = 5 \quad w(x_4) = 7 \quad w(x_5) = 10$

$K = 14$

	0	1	2	3	4	5									14
	w	f	f	f	f	f	f	f	f	f	f	f	f	f	f
x_1	w	f	w	f	f	f	f	f	f	f	f	f	f	f	f
x_2	w	f	w	w	f	w	f	f	f	f	f	f	f	f	f
x_3															
x_4															
x_5															

$$T[i, j] = T[i-1, j] \vee T[i-1, j-x_i]$$

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Beispiel

$M = \{x_1, x_2, x_3, x_4, x_5\}$

$w(x_1) = 2 \qquad w(x_2) = 3 \quad w(x_3) = 5 \quad w(x_4) = 7 \quad w(x_5) = 10$

$K = 14$

	0	1	2	3	4	5									14
	w	f	f	f	f	f	f	f	f	f	f	f	f	f	f
x_1	w	f	w	f	f	f	f	f	f	f	f	f	f	f	f
x_2	w	f	w	w	f	w	f	f	f	f	f	f	f	f	f
x_3	w	f	w	w	f	w	f	w	w	f	w	f	f	f	f
x_4															
x_5															

$$T[i, j] = T[i-1, j] \vee T[i-1, j-x_i]$$

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Beispiel

$M = \{x_1, x_2, x_3, x_4, x_5\}$

$w(x_1) = 2 \qquad w(x_2) = 3 \quad w(x_3) = 5 \quad w(x_4) = 7 \quad w(x_5) = 10$

$K = 14$

	0	1	2	3	4	5									14
	w	f	f	f	f	f	f	f	f	f	f	f	f	f	f
x_1	w	f	w	f	f	f	f	f	f	f	f	f	f	f	f
x_2	w	f	w	w	f	w	f	f	f	f	f	f	f	f	f
x_3	w	f	w	w	f	w	f	w	w	f	w	f	f	f	f
x_4	w	f	w	w	f	w	f	w	w	w	w	f	w	f	w
x_5															

$$T[i, j] = T[i-1, j] \vee T[i-1, j-x_i]$$

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Approximative Algorithmen

Wiederholung

Sei Π ein Optimierungsproblem. Ein polynomialer Algorithmus $\mathcal{A}$, der für jedes $I \in D_\Pi$ einen Wert $\mathcal{A}(I)$ liefert, mit

$$|\mathrm{OPT}(I) - \mathcal{A}(I)| \leq K$$

und $K \in \mathbb{N}_0$ konstant, heißt Approximationsalgorithmus mit Differenzengarantie oder absoluter Approximationsalgorithmus.

Aufgabe

Problem CLIQUE

Gegeben: Ungerichteter Graph $G = (V, E)$.

Gesucht: Möglichst große Clique von G.

Hinweis: $C \subseteq V$ heißt *Clique*, falls für jedes Paar $u, v \in C$ die Kante $\{u, v\} \in E$ existiert.

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Aufgabe

Problem CLIQUE
Gegeben: Ungerichteter Graph $G = (V, E)$.
Gesucht: Möglichst große Clique von G.
Hinweis: $C \subseteq V$ heißt *Clique*, falls für jedes Paar $u, v \in C$ die Kante $\{u, v\} \in E$ existiert.

Zeige: Es gibt keinen Approximationsalgorithmus.

- **Annahme:** Sei $\mathcal{A}$ absoluter Approxalgo mit $|OPT(I) - \mathcal{A}(I)| \leq K \quad \forall I$
- **Idee:** Konstruiere aus $\mathcal{A}$ polynomiellen exakten Algo für CLIQUE
- **Technik:** Nutze Instanz I zum Bau von I', sodass Lösung in I' eine ßu große"Lösung in I induziert.

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Sei $(G = (V, E), K)$ Instanz von CLIQUE

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Sei $(G = (V, E), K)$ Instanz von CLIQUE

Graph G^m ist für $m \in \mathbb{N}$ wie folgt definiert:

1. Kopiere G m-mal
2. Verbinden jeden Knoten einer Kopie mit jedem Knoten aller anderen Kopien

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Sei $(G = (V, E), K)$ Instanz von CLIQUE

Graph G^m ist für $m \in \mathbb{N}$ wie folgt definiert:

1. Kopiere G m-mal
2. Verbinden jeden Knoten einer Kopie mit jedem Knoten aller anderen Kopien

Beob.: $\exists$ Clique C der Größe α in G *gdw.* $\exists$ Clique C^m der Größe αm in G^m.

Sei $(G = (V, E), K)$ Instanz von CLIQUE

Graph G^m ist für $m \in \mathbb{N}$ wie folgt definiert:

1. Kopiere G m-mal
2. Verbinden jeden Knoten einer Kopie mit jedem Knoten aller anderen Kopien

Beob.: $\exists$ Clique C der Größe α in G *gdw.* $\exists$ Clique C^m der Größe αm in G^m.

Annahme: Es gibt Approximationalg. $\mathcal{A}$ mit $|\mathcal{A}(I) - \mathrm{OPT}(I)| \leq K$ (für alle I)

Sei $(G = (V, E), K)$ Instanz von CLIQUE

Graph G^m ist für $m \in \mathbb{N}$ wie folgt definiert:

1. Kopiere G m-mal
2. Verbinden jeden Knoten einer Kopie mit jedem Knoten aller anderen Kopien

Beob.: $\exists$ Clique C der Größe α in G *gdw.* $\exists$ Clique C^m der Größe αm in G^m.

Annahme: Es gibt Approximationalg. $\mathcal{A}$ mit $|\mathcal{A}(I) - \mathrm{OPT}(I)| \leq K$ (für alle I)

Strategie um größte Clique in G zu finden:

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Sei $(G = (V, E), K)$ Instanz von CLIQUE

Graph G^m ist für $m \in \mathbb{N}$ wie folgt definiert:

1. Kopiere G m-mal
2. Verbinden jeden Knoten einer Kopie mit jedem Knoten aller anderen Kopien

Beob.: $\exists$ Clique C der Größe α in G *gdw.* $\exists$ Clique C^m der Größe αm in G^m.

Annahme: Es gibt Approximationalg. $\mathcal{A}$ mit $|\mathcal{A}(I) - \mathrm{OPT}(I)| \leq K$ (für alle I)

Strategie um größte Clique in G zu finden:

1. Erstelle G^{K+1} von G
2. Wende $\mathcal{A}$ auf G^{K+1} an: liefert Clique C^{K+1}
3. Extrahiere aus C^{K+1} eine Clique C für G der Größe $(K+1) \cdot |C| = |C^{K+1}|$

Sei $(G = (V, E), K)$ Instanz von CLIQUE

Graph G^m ist für $m \in \mathbb{N}$ wie folgt definiert:

1. Kopiere G m-mal
2. Verbinden jeden Knoten einer Kopie mit jedem Knoten aller anderen Kopien

Beob.: $\exists$ Clique C der Größe α in G *gdw.* $\exists$ Clique C^m der Größe αm in G^m.

Annahme: Es gibt Approximationalg. $\mathcal{A}$ mit $|\mathcal{A}(I) - \mathrm{OPT}(I)| \leq K$ (für alle I)

Strategie um größte Clique in G zu finden:

1. Erstelle G^{K+1} von G
2. Wende $\mathcal{A}$ auf G^{K+1} an: liefert Clique C^{K+1}
3. Extrahiere aus C^{K+1} eine Clique C für G der Größe $(K+1) \cdot |C| = |C^{K+1}|$

$$|\mathcal{A}(G^{K+1}) - \mathrm{OPT}(G^{K+1})| \leq K \quad \Leftrightarrow \quad |\mathcal{A}(G^{K+1}) - (K+1)\mathrm{OPT}(G)| \leq K$$

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Sei $(G = (V, E), K)$ Instanz von CLIQUE

Graph G^m ist für $m \in \mathbb{N}$ wie folgt definiert:

1. Kopiere G m-mal
2. Verbinden jeden Knoten einer Kopie mit jedem Knoten aller anderen Kopien

Beob.: $\exists$ Clique C der Größe α in G *gdw.* $\exists$ Clique C^m der Größe αm in G^m.

Annahme: Es gibt Approximationalg. $\mathcal{A}$ mit $|\mathcal{A}(I) - \mathrm{OPT}(I)| \leq K$ (für alle I)

Strategie um größte Clique in G zu finden:

1. Erstelle G^{K+1} von G
2. Wende $\mathcal{A}$ auf G^{K+1} an: liefert Clique C^{K+1}
3. Extrahiere aus C^{K+1} eine Clique C für G der Größe $(K+1) \cdot |C| = |C^{K+1}|$

$$|\mathcal{A}(G^{K+1}) - \mathrm{OPT}(G^{K+1})| \leq K \quad \Leftrightarrow \quad |\mathcal{A}(G^{K+1}) - (K+1)\mathrm{OPT}(G)| \leq K$$

$$||C^{K+1}| - (K+1)\mathrm{OPT}(G)| \leq K \quad \Leftrightarrow \quad |(K+1)|C| - (K+1)\mathrm{OPT}(G)| \leq K$$

Sei $(G = (V, E), K)$ Instanz von CLIQUE

Graph G^m ist für $m \in \mathbb{N}$ wie folgt definiert:

1. Kopiere G m-mal
2. Verbinden jeden Knoten einer Kopie mit jedem Knoten aller anderen Kopien

Beob.: $\exists$ Clique C der Größe α in G *gdw.* $\exists$ Clique C^m der Größe αm in G^m.

Annahme: Es gibt Approximationalg. $\mathcal{A}$ mit $|\mathcal{A}(I) - \mathrm{OPT}(I)| \leq K$ (für alle I)

Strategie um größte Clique in G zu finden:

1. Erstelle G^{K+1} von G
2. Wende $\mathcal{A}$ auf G^{K+1} an: liefert Clique C^{K+1}
3. Extrahiere aus C^{K+1} eine Clique C für G der Größe $(K+1) \cdot |C| = |C^{K+1}|$

$$|\mathcal{A}(G^{K+1}) - \mathrm{OPT}(G^{K+1})| \leq K \quad \Leftrightarrow \quad |\mathcal{A}(G^{K+1}) - (K+1)\mathrm{OPT}(G)| \leq K$$

$$||C^{K+1}| - (K+1)\mathrm{OPT}(G)| \leq K \quad \Leftrightarrow \quad |(K+1)|C| - (K+1)\mathrm{OPT}(G)| \leq K$$

$$\big|\,|C| - \mathrm{OPT}(G)\big| \leq \frac{K}{K+1} < 1$$

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Sei $(G = (V, E), K)$ Instanz von CLIQUE

Graph G^m ist für $m \in \mathbb{N}$ wie folgt definiert:

1. Kopiere G m-mal
2. Verbinden jeden Knoten einer Kopie mit jedem Knoten aller anderen Kopien

Beob.: $\exists$ Clique C der Größe α in G *gdw.* $\exists$ Clique C^m der Größe αm in G^m.

Annahme: Es gibt Approximationalg. $\mathcal{A}$ mit $|\mathcal{A}(I) - \mathrm{OPT}(I)| \leq K$ (für alle I)

Strategie um größte Clique in G zu finden:

1. Erstelle G^{K+1} von G
2. Wende $\mathcal{A}$ auf G^{K+1} an: liefert Clique C^{K+1}
3. Extrahiere aus C^{K+1} eine Clique C für G der Größe $(K+1) \cdot |C| = |C^{K+1}|$

$$|\mathcal{A}(G^{K+1}) - \mathrm{OPT}(G^{K+1})| \leq K \quad \Leftrightarrow \quad |\mathcal{A}(G^{K+1}) - (K+1)\mathrm{OPT}(G)| \leq K$$

$$||C^{K+1}| - (K+1)\mathrm{OPT}(G)| \leq K \quad \Leftrightarrow \quad |(K+1)|C| - (K+1)\mathrm{OPT}(G)| \leq K$$

$$||C| - \mathrm{OPT}(G)| \leq \frac{K}{K+1} < 1 \quad \longrightarrow \quad |C| = \mathrm{OPT}(G)$$

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Wiederholung

Sei Π ein Optimierungsproblem. Ein polynomialer Algorithmus $\mathcal{A}$, der für jedes $I \in D_\Pi$ einen Wert $\mathcal{A}(I)$ liefert mit $R_\mathcal{A}(I) \leq K$, wobei $K \geq 1$ eine Konstante, und

$$\mathcal{R}_\mathcal{A}(I) := \begin{cases} \frac{\mathcal{A}(I)}{\mathrm{OPT}(I)} & \text{falls } \Pi \text{ Minimierungsproblem} \\ \\ \frac{\mathrm{OPT}(I)}{\mathcal{A}(I)} & \text{falls } \Pi \text{ Maximierungsproblem} \end{cases}$$

heißt Approximationsalgorithmus mit relativer Gütegarantie. $\mathcal{A}$ heißt ε–approximativ, falls $\mathcal{R}_\mathcal{A}(I) \leq 1 + \varepsilon$ für alle $I \in D_\Pi$.

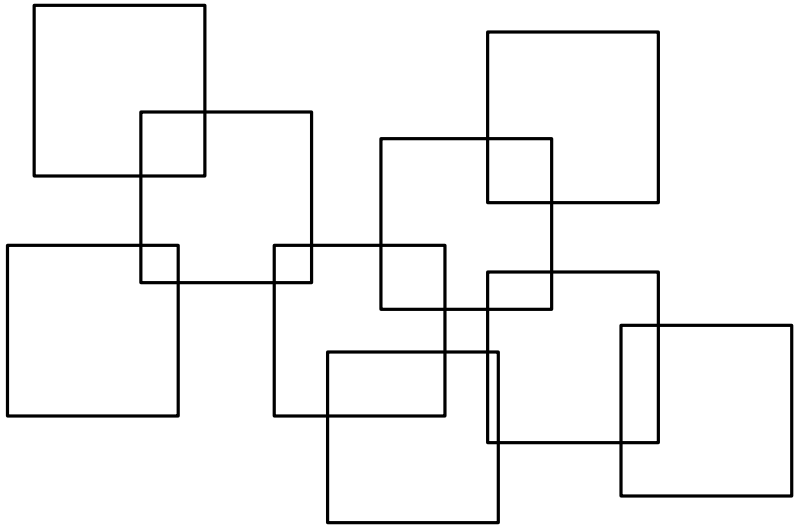
Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Aufgabe

INDEPENDENTSQUARES

Gegeben: Menge $Q = \{q_1, \ldots, q_n\}$ gleichgroßer, achsenparalleler Quadrate in der Ebene.

Gesucht: Möglichst große unabhängige Menge $S \subseteq Q$. Dabei heißt $S \subseteq Q$ *unabhängig*, falls für alle $q_i, q_j \in S$ mit $i \neq j$ gilt, dass q_i und q_j sich nicht schneiden.

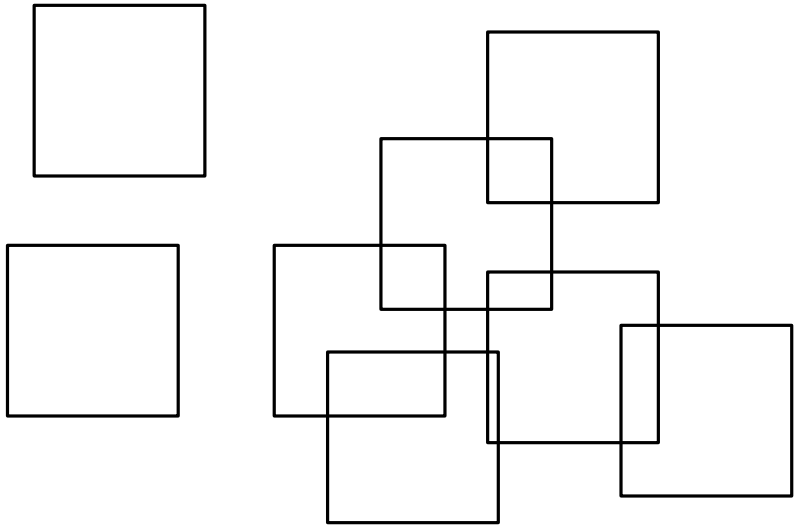


Aufgabe

INDEPENDENTSQUARES

Gegeben: Menge $Q = \{q_1, \ldots, q_n\}$ gleichgroßer, achsenparalleler Quadrate in der Ebene.

Gesucht: Möglichst große unabhängige Menge $S \subseteq Q$. Dabei heißt $S \subseteq Q$ *unabhängig*, falls für alle $q_i, q_j \in S$ mit $i \neq j$ gilt, dass q_i und q_j sich nicht schneiden.

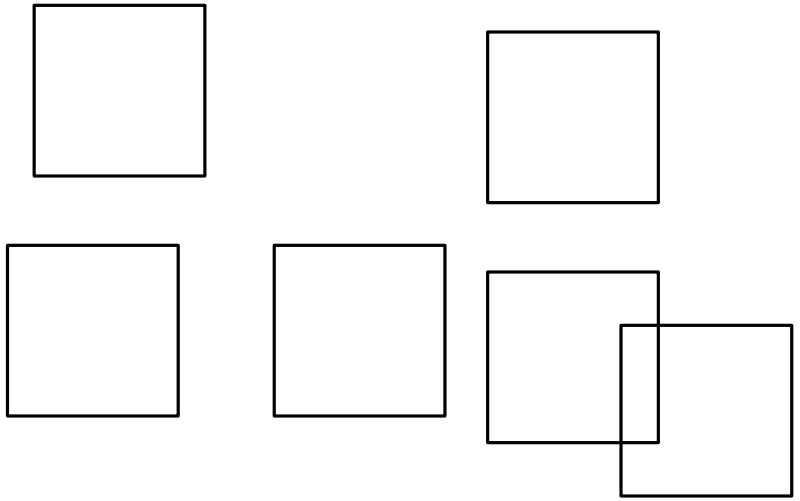


Aufgabe

INDEPENDENTSQUARES

Gegeben: Menge $Q = \{q_1, \ldots, q_n\}$ gleichgroßer, achsenparalleler Quadrate in der Ebene.

Gesucht: Möglichst große unabhängige Menge $S \subseteq Q$. Dabei heißt $S \subseteq Q$ *unabhängig*, falls für alle $q_i, q_j \in S$ mit $i \neq j$ gilt, dass q_i und q_j sich nicht schneiden.



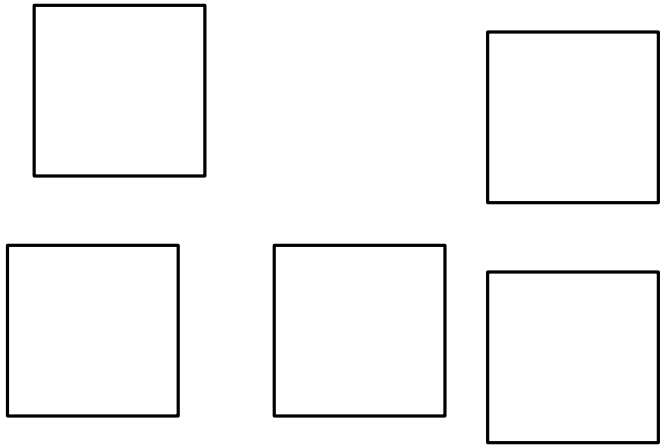
Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Aufgabe

INDEPENDENTSQUARES

Gegeben: Menge $Q = \{q_1, \ldots, q_n\}$ gleichgroßer, achsenparalleler Quadrate in der Ebene.

Gesucht: Möglichst große unabhängige Menge $S \subseteq Q$. Dabei heißt $S \subseteq Q$ *unabhängig*, falls für alle $q_i, q_j \in S$ mit $i \neq j$ gilt, dass q_i und q_j sich nicht schneiden.



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Aufgabe

INDEPENDENTSQUARES

Gegeben: Menge $Q = \{q_1, \ldots, q_n\}$ gleichgroßer, achsenparalleler Quadrate in der Ebene.

Gesucht: Möglichst große unabhängige Menge $S \subseteq Q$. Dabei heißt $S \subseteq Q$ *unabhängig*, falls für alle $q_i, q_j \in S$ mit $i \neq j$ gilt, dass q_i und q_j sich nicht schneiden.

Eingabe : $Q = \{q_1, \ldots, q_n\}$ gleichgroßer, achsenparalleler Quadrate mit Mittelpunkten $c_1, \ldots, c_n$, sodass für die x-Koordinaten der Mittelpunkte gilt $x(c_1) \leq \ldots \leq x(c_n)$.

Ausgabe : Unabhängige Menge $S \subseteq Q$.

$S \leftarrow \emptyset$;

für $i = 1, \ldots, n$ **tue**

  wenn $q_i \in Q$ **dann**

    $S \leftarrow S \cup \{q_i\}$;

    $Q \leftarrow Q \setminus (\{q_i\} \cup \{q_j \in Q \mid q_j$ und q_i schneiden sich.$\})$

return S;

Algorithmus 1 : SWEEPLINE

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Aufgabe

Gesucht: Familie $Q_1, Q_2, Q_3, \ldots$ gleichgroßer, achsenparalleler Quadrate an, sodass gilt

$$|Q_n| \in \Theta(n) \text{ und } |\textsc{SweepLine}(Q_n)| = \frac{1}{2}|\text{OPT}(Q_n)|$$

für alle $n \in \mathbb{N}$. Dabei bezeichnet $\text{OPT}(Q)$ die kardinalitätsmaximale unabhängige Menge von Q. Begründen Sie Ihre Antwort.

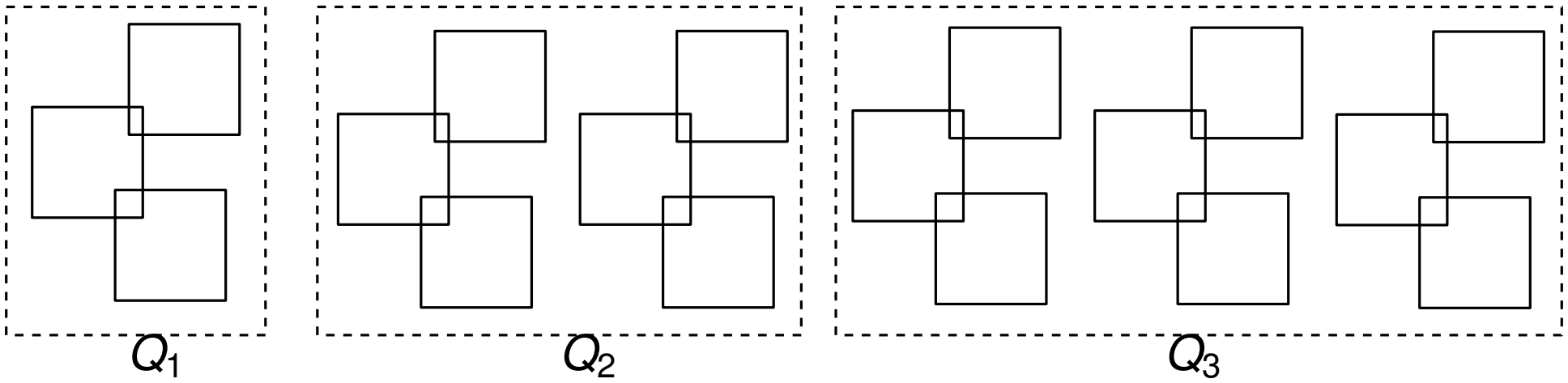
Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Aufgabe

Gesucht: Familie $Q_1, Q_2, Q_3, \ldots$ gleichgroßer, achsenparalleler Quadrate an, sodass gilt

$$|Q_n| \in \Theta(n) \text{ und } |\textsc{SweepLine}(Q_n)| = \frac{1}{2}|\text{OPT}(Q_n)|$$

für alle $n \in \mathbb{N}$. Dabei bezeichnet $\text{OPT}(Q)$ die kardinalitätsmaximale unabhängige Menge von Q. Begründen Sie Ihre Antwort.



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Aufgabe

Zeigen Sie, dass SWEEPLINE für INDEPENDENTSQUARES ein Approximationsalgorithmus mit relativer Gütegarantie 2 ist.

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Aufgabe

Zeigen Sie, dass SweepLine für IndependentSquares ein Approximationsalgorithmus mit relativer Gütegarantie 2 ist.

Betrachte den Fall, dass q_i im i-ten Schritt in S eingefügt wird.

Bezeichne Q_i die Menge Q direkt vor dem i-ten Schritt.

Sei $K = \{q_j \in Q_i \mid q_j \text{ und } q_i \text{ schneiden sich}\}$.

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Aufgabe

Zeigen Sie, dass SWEEPLINE für INDEPENDENTSQUARES ein Approximationsalgorithmus mit relativer Gütegarantie 2 ist.

Betrachte den Fall, dass q_i im i-ten Schritt in S eingefügt wird.

Bezeichne Q_i die Menge Q direkt vor dem i-ten Schritt.

Sei $K = \{q_j \in Q_i \mid q_j \text{ und } q_i \text{ schneiden sich}\}$.

Alle Quadrate gleichgroß und achsenparallel

$\longrightarrow$ Jedes Quadrat $q_j \in K$ überdeckt mindestens eine Ecke von q_i.

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Aufgabe

Zeigen Sie, dass SWEEPLINE für INDEPENDENTSQUARES ein Approximationsalgorithmus mit relativer Gütegarantie 2 ist.

Betrachte den Fall, dass q_i im i-ten Schritt in S eingefügt wird.

Bezeichne Q_i die Menge Q direkt vor dem i-ten Schritt.

Sei $K = \{q_j \in Q_i \mid q_j \text{ und } q_i \text{ schneiden sich}\}$.

Alle Quadrate gleichgroß und achsenparallel

→ Jedes Quadrat $q_j \in K$ überdeckt mindestens eine Ecke von q_i.

$x(c_j) \geq x(c_i)$ für alle $q_j \in Q$:

→ Obere-rechte oder untere-rechte Ecke von q_i muss überdeckt sein.

→ $|\text{OPT}(K)| \leq 2$

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Aufgabe

Zeigen Sie, dass SweepLine für IndependentSquares ein Approximationsalgorithmus mit relativer Gütegarantie 2 ist.

Betrachte den Fall, dass q_i im i-ten Schritt in S eingefügt wird.

Bezeichne Q_i die Menge Q direkt vor dem i-ten Schritt.

Sei $K = \{q_j \in Q_i \mid q_j \text{ und } q_i \text{ schneiden sich}\}$.

Alle Quadrate gleichgroß und achsenparallel

→ Jedes Quadrat $q_j \in K$ überdeckt mindestens eine Ecke von q_i.

$x(c_j) \geq x(c_i)$ für alle $q_j \in Q$:

→ Obere-rechte oder untere-rechte Ecke von q_i muss überdeckt sein.

→ $|\mathrm{OPT}(K)| \leq 2$

Schlimmster Fall: Zwei Quadrate der optimalen Lösung gehen verloren, während eins zur Lösung hinzugenommen wird.

→ Relative Gütegarantie

Aufgabe

Sei $G = (V, E)$ ein gerichteter Graph und sei $G_1 = (V, E_1 \subseteq E)$ ein inklusionsmaximaler azyklischer Teilgraph von G. Zudem sei $G_2 = (V, E_2 = E \setminus E_1)$ das Komplement zu G_1.

Zeige: Für jede Kante $(u, v) \in E_2$ gibt es in G_1 einen gerichteten Pfad von v nach u.

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Aufgabe

Sei $G = (V, E)$ ein gerichteter Graph und sei $G_1 = (V, E_1 \subseteq E)$ ein inklusionsmaximaler azyklischer Teilgraph von G. Zudem sei $G_2 = (V, E_2 = E \setminus E_1)$ das Komplement zu G_1.

Zeige: Für jede Kante $(u, v) \in E_2$ gibt es in G_1 einen gerichteten Pfad von v nach u.

Annahme: Es gibt Kante $(u, v) \in E_2$, sodass es keinen gerichteten Pfad von v nach u in G_1 gibt.

⟶ Kante (u, v) kann zu E_1 hinzu genommen werden, ohne das ein gerichteter Kreis entsteht.

G_1 inklusionsmaximal, azyklischer Graph ist.

Aufgabe

Sei $G = (V, E)$ ein gerichteter Graph und sei $G_1 = (V, E_1 \subseteq E)$ ein inklusionsmaximaler azyklischer Teilgraph von G. Zudem sei $G_2 = (V, E_2 = E \setminus E_1)$ das Komplement zu G_1.

Zeige: G_2 ist azyklisch.

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Aufgabe

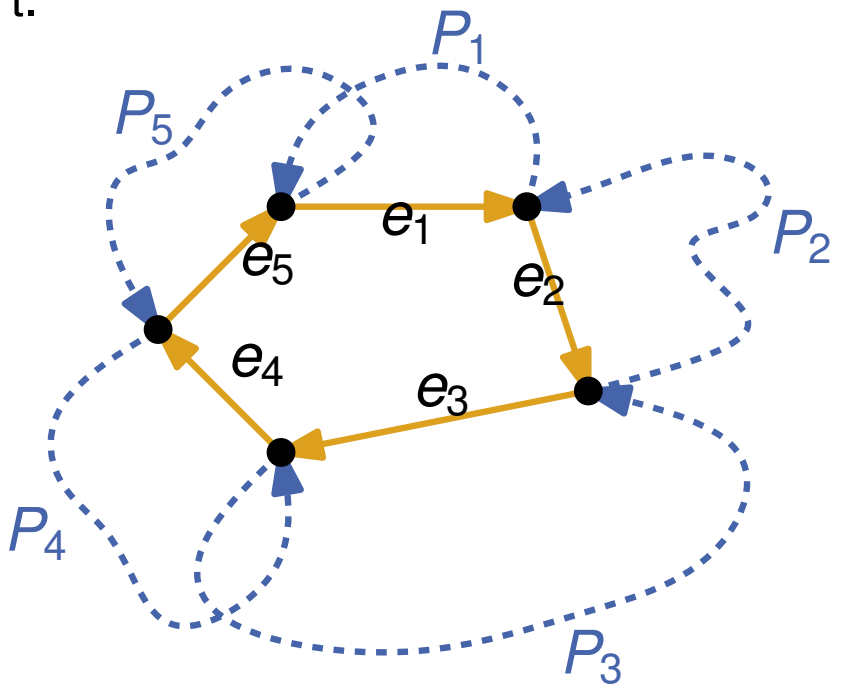
Sei $G = (V, E)$ ein gerichteter Graph und sei $G_1 = (V, E_1 \subseteq E)$ ein inklusionsmaximaler azyklischer Teilgraph von G. Zudem sei $G_2 = (V, E_2 = E \setminus E_1)$ das Komplement zu G_1.

Zeige: G_2 ist azyklisch.

Annahme: G_2 enthält einen Kreis $P = (e_1, e_2, \ldots, e_k)$.

Teilaufgabe (a): Für jede dieser Kanten $e_i \in P$ gibt es gerichteten Pfad P_i in G_1, der vom Zielknoten von e_i zum Startknoten von e_i führt.

Verbinde Pfade zu einem Kreis.



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

MAXIMUM ACYCLIC GRAPH:

Gegeben: Gerichteter Graph $G = (V, E)$.

Gesucht: Kardinalitätsmaximaler azyklischer Teilgraph von G.

Gesucht: Approx.algo. für MAXIMUM ACYCLIC GRAPH mit relativer Gütegarantie 2.

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

MAXIMUM ACYCLIC GRAPH:

Gegeben: Gerichteter Graph $G = (V, E)$.
Gesucht: Kardinalitätsmaximaler azyklischer Teilgraph von G.

Gesucht: Approx.algo. für MAXIMUM ACYCLIC GRAPH mit relativer Gütegarantie 2.

Berechne inklusionsmaximalen azyklischen Teilgraph $G_1 = (V, E_1)$ von G.;
Berechne Komplementgraph $G_2 = (V, E \setminus E_1)$ zu G_1.;
wenn $|E_1| \geq |E_2|$ **dann**
    return G_1;
sonst
    return G_2;

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

MAXIMUM ACYCLIC GRAPH:

Gegeben: Gerichteter Graph $G = (V, E)$.
Gesucht: Kardinalitätsmaximaler azyklischer Teilgraph von G.

Gesucht: Approx.algo. für MAXIMUM ACYCLIC GRAPH mit relativer Gütegarantie 2.

Berechne inklusionsmaximalen azyklischen Teilgraph $G_1 = (V, E_1)$ von G.;
Berechne Komplementgraph $G_2 = (V, E \setminus E_1)$ zu G_1.;
wenn $|E_1| \geq |E_2|$ **dann**
  return G_1;
sonst
  return G_2;

- G_2 ist azyklisch.
- $|E_1| + |E_2| = |E|$ und $E_1 \cap E_2 \neq \emptyset$

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

MAXIMUM ACYCLIC GRAPH:

Gegeben: Gerichteter Graph $G = (V, E)$.
Gesucht: Kardinalitätsmaximaler azyklischer Teilgraph von G.

Gesucht: Approx.algo. für MAXIMUM ACYCLIC GRAPH mit relativer Gütegarantie 2.

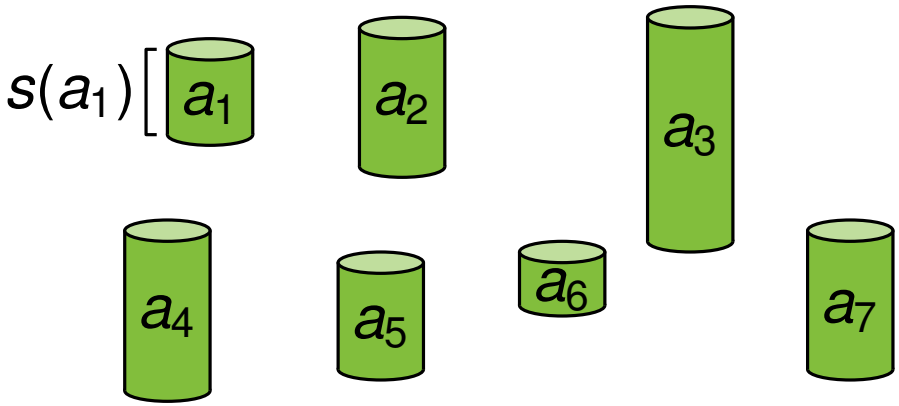
Berechne inklusionsmaximalen azyklischen Teilgraph $G_1 = (V, E_1)$ von G.;
Berechne Komplementgraph $G_2 = (V, E \setminus E_1)$ zu G_1.;
wenn $|E_1| \geq |E_2|$ **dann**
  return G_1;
sonst
  return G_2;

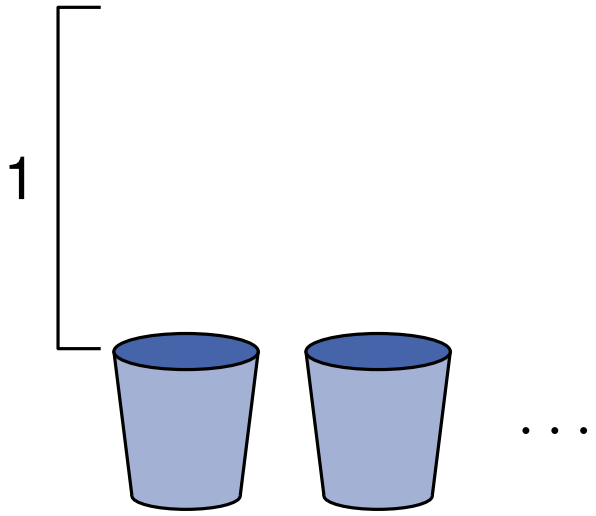
- G_2 ist azyklisch.
- $|E_1| + |E_2| = |E|$ und $E_1 \cap E_2 \neq \emptyset$
  → $|E_1| \geq \frac{1}{2}|E|$ oder $|E_2| \geq \frac{1}{2}|E|$
  → Optimale Lösung kann nicht mehr als $|E|$ Kanten enthalten.

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Bin Packing – Definition

endliche Menge $M = \{a_1, \ldots, a_n\}$
mit Gewichtsfunktion
$s\colon M \longrightarrow (0, 1]$





Eimer (Bins) mit Fassungsvermögen 1

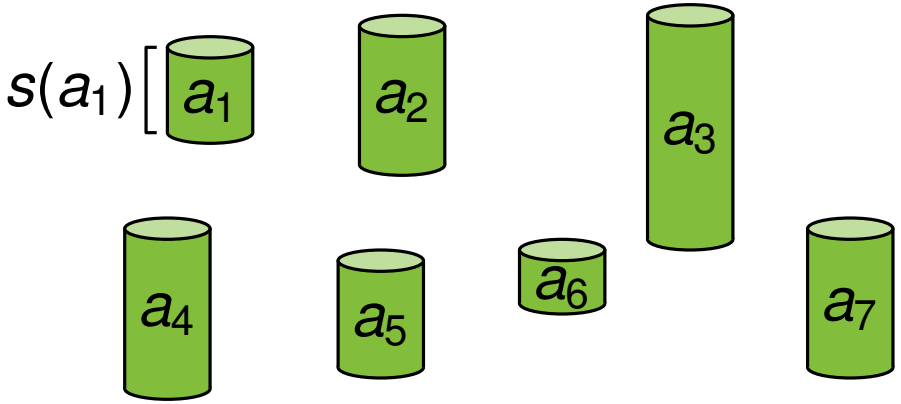
Problem: BIN PACKING
Weise die Elemente in M einer minimalen Anzahl an Bins $B_1, \ldots, B_m$ zu, sodass für jeden Bin B gilt:

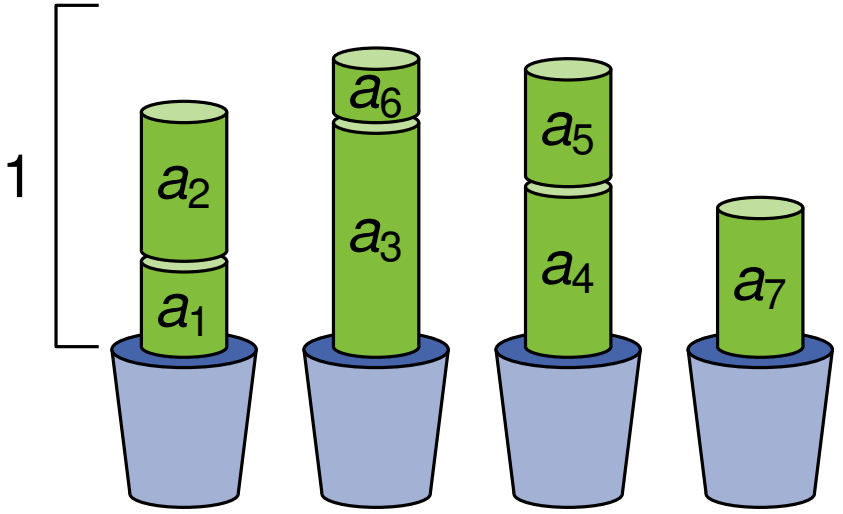
$$\sum_{a_i \in B} s(a_i) \leq 1$$

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Bin Packing – Definition

endliche Menge $M = \{a_1, \ldots, a_n\}$
mit Gewichtsfunktion
$s\colon M \longrightarrow (0, 1]$





Eimer (Bins) mit Fassungsvermögen 1

4 Bins

Problem: BIN PACKING

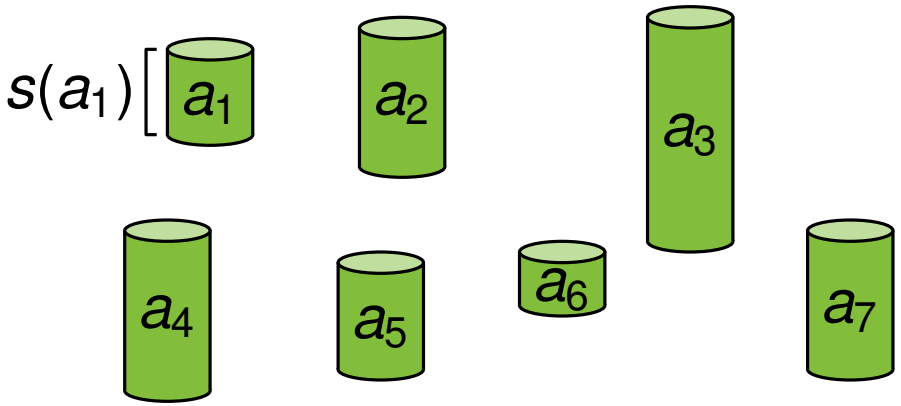
Weise die Elemente in M einer minimalen Anzahl an Bins $B_1, \ldots, B_m$ zu, sodass für jeden Bin B gilt:

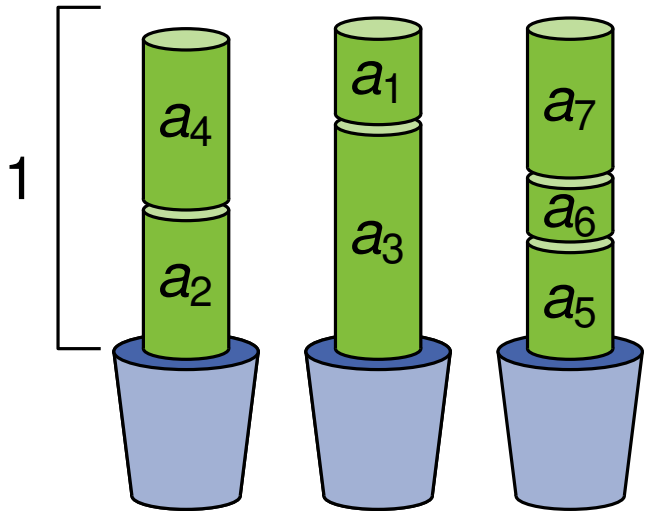
$$\sum_{a_i \in B} s(a_i) \leq 1$$

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Bin Packing – Definition

endliche Menge $M = \{a_1, \ldots, a_n\}$
mit Gewichtsfunktion
$s\colon M \longrightarrow (0, 1]$





Eimer (Bins) mit Fassungsvermögen 1

3 Bins

Problem: BIN PACKING

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

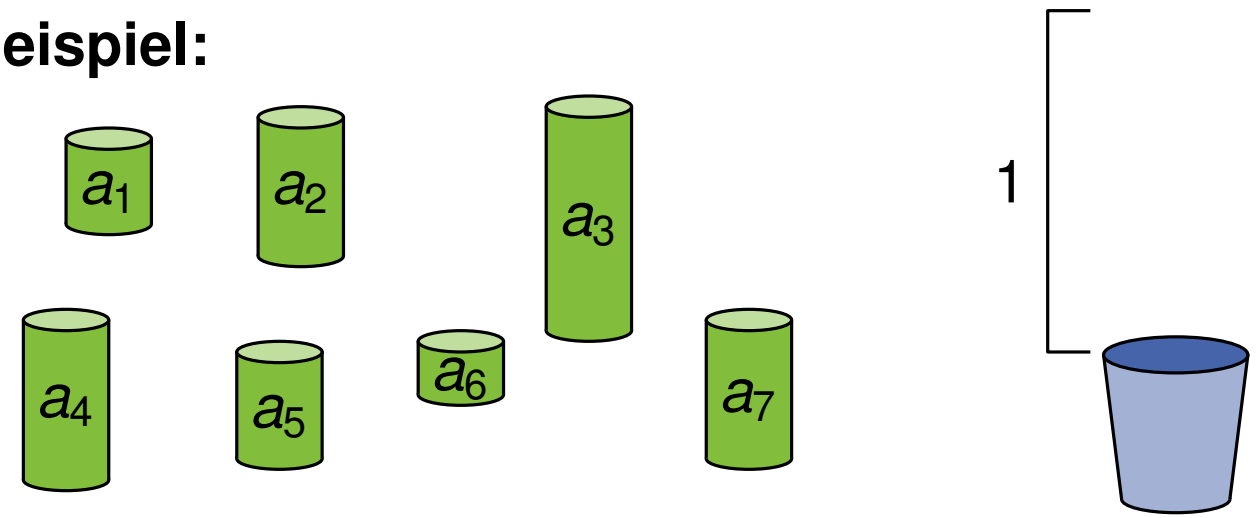
$$\sum_{a_i \in B} s(a_i) \leq 1$$

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Bin-Packing

Strategie:
Füge Elemente nacheinander in den aktuellen Bin ein. Wenn ein Element nicht mehr passt, schließe den Bin ab und nimm einen neuen.

Beispiel:

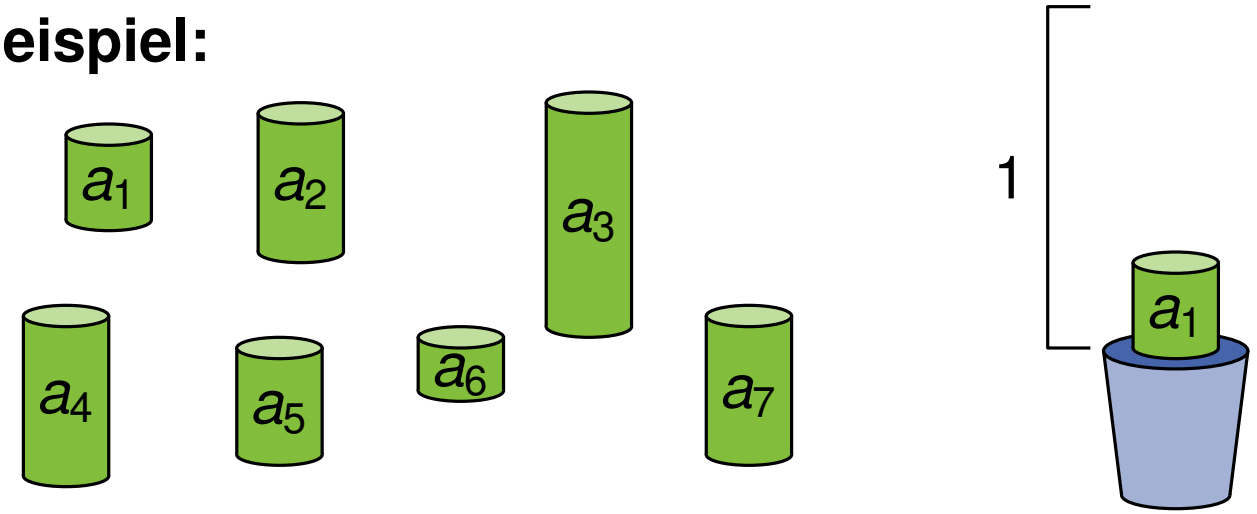


Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Bin-Packing

Strategie:
Füge Elemente nacheinander in den aktuellen Bin ein. Wenn ein Element nicht mehr passt, schließe den Bin ab und nimm einen neuen.

Beispiel:

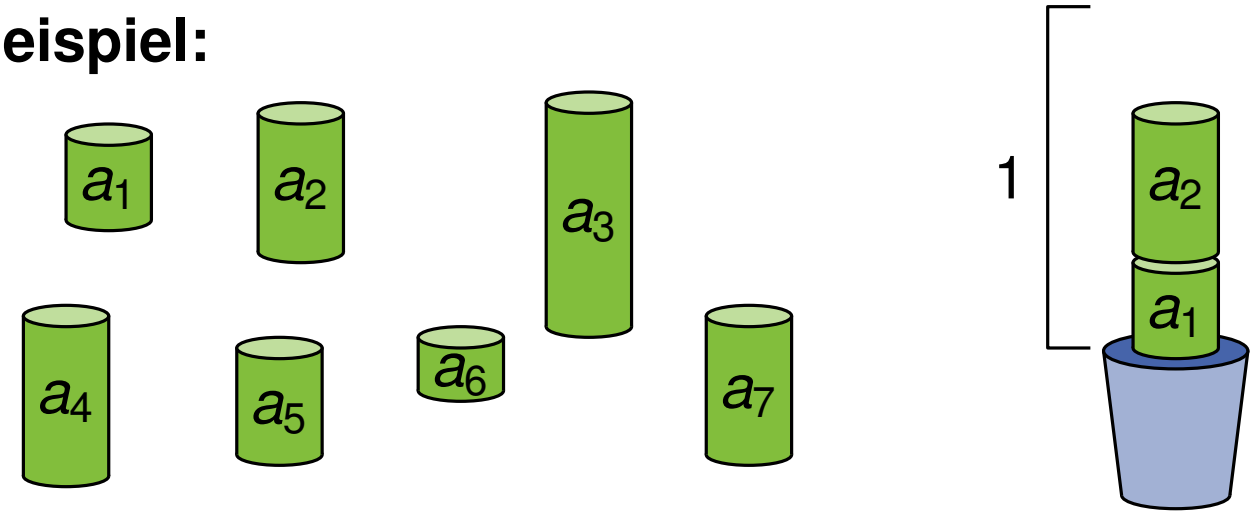


Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Bin-Packing

Strategie:
Füge Elemente nacheinander in den aktuellen Bin ein. Wenn ein Element nicht mehr passt, schließe den Bin ab und nimm einen neuen.

Beispiel:

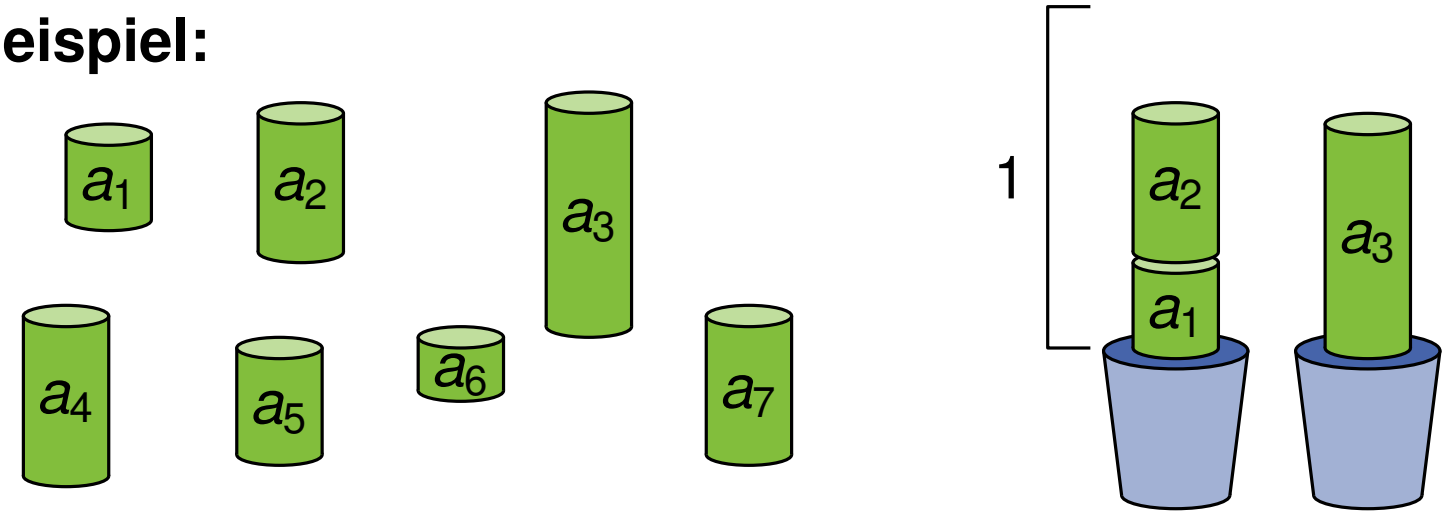


Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Bin-Packing

Strategie:
Füge Elemente nacheinander in den aktuellen Bin ein. Wenn ein Element nicht mehr passt, schließe den Bin ab und nimm einen neuen.

Beispiel:

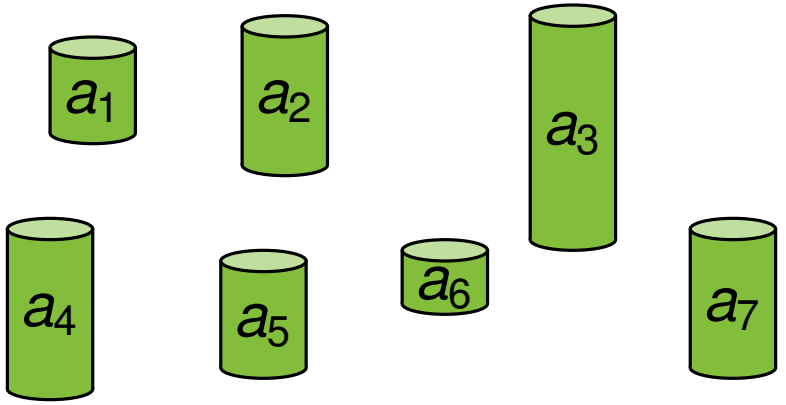


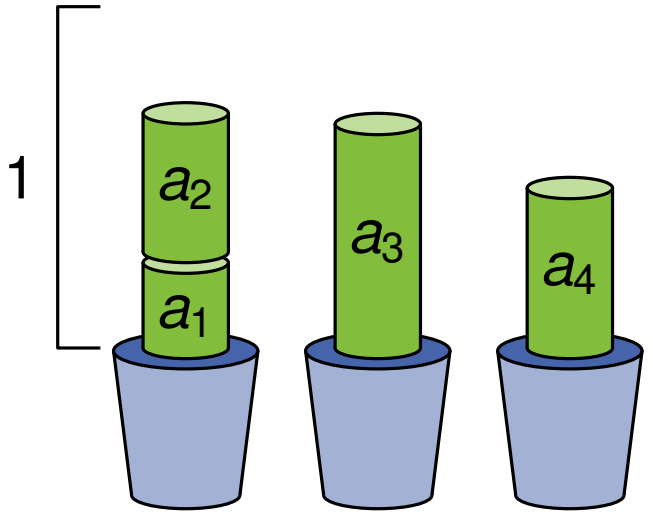
Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Bin-Packing

Strategie:
Füge Elemente nacheinander in den aktuellen Bin ein. Wenn ein Element nicht mehr passt, schließe den Bin ab und nimm einen neuen.

Beispiel:



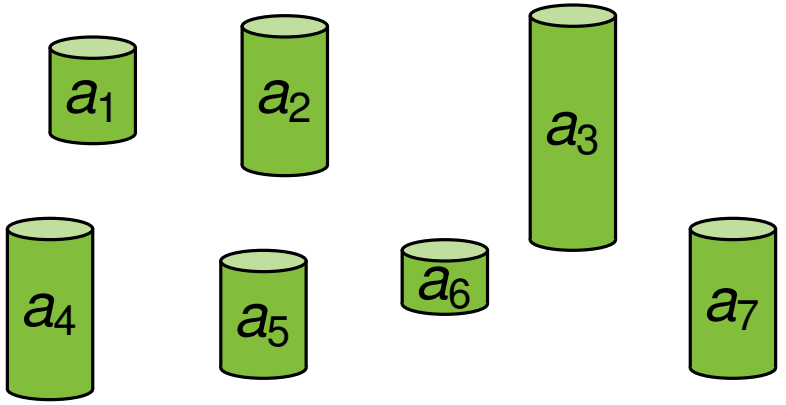


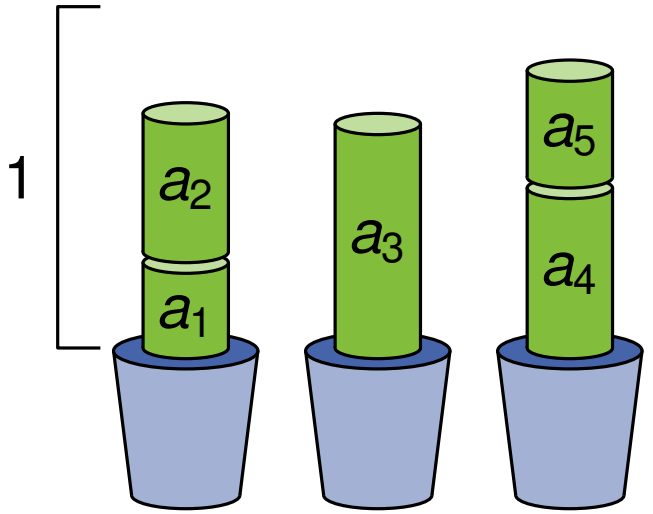
Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Bin-Packing

Strategie:
Füge Elemente nacheinander in den aktuellen Bin ein. Wenn ein Element nicht mehr passt, schließe den Bin ab und nimm einen neuen.

Beispiel:



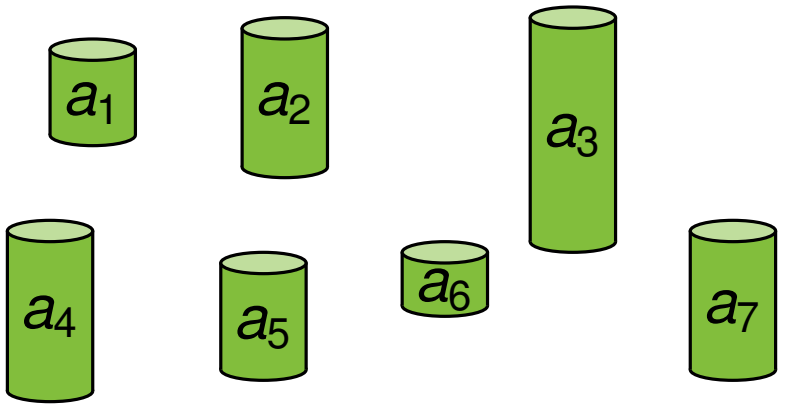


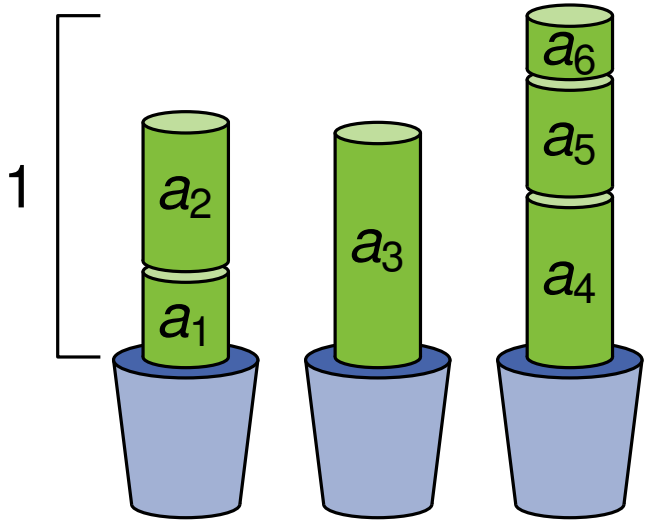
Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Bin-Packing

Strategie:
Füge Elemente nacheinander in den aktuellen Bin ein. Wenn ein Element nicht mehr passt, schließe den Bin ab und nimm einen neuen.

Beispiel:



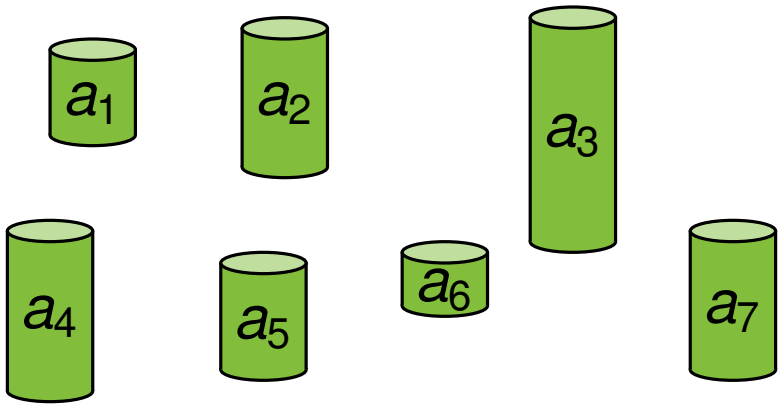


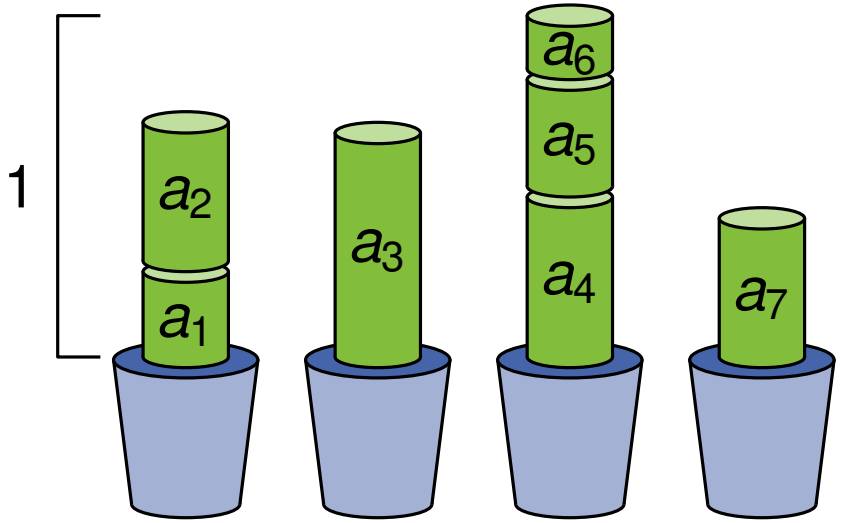
Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Bin-Packing

Strategie:
Füge Elemente nacheinander in den aktuellen Bin ein. Wenn ein Element nicht mehr passt, schließe den Bin ab und nimm einen neuen.

Beispiel:





Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

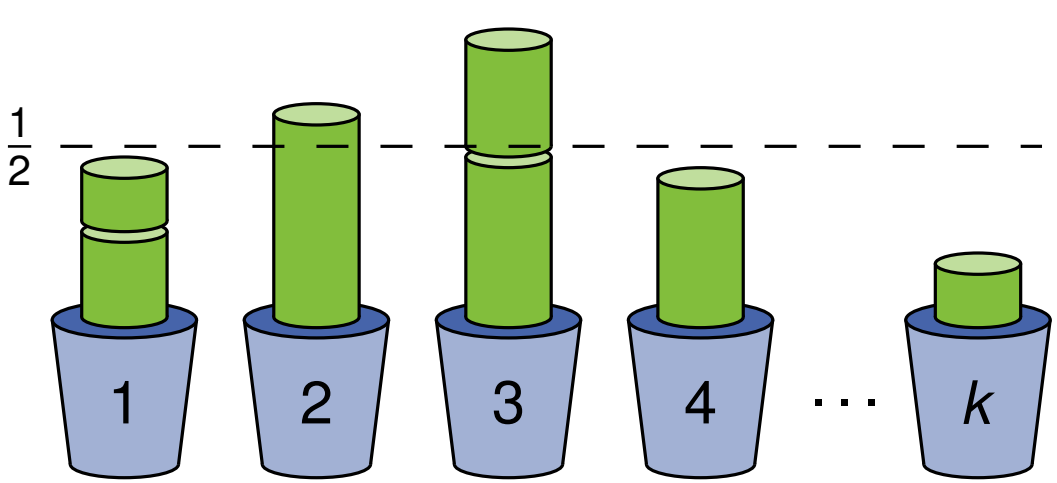
Next Fit – Approximation

Satz: Approximation **(Satz 7.5)**

NEXT FIT erfüllt $\mathcal{R}_{NF} \leq 2$.

Beweis:

- Sei $k = NF(I)$ die Anzahl an Bins, die NEXTFIT für die Instanz I benötigt.



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

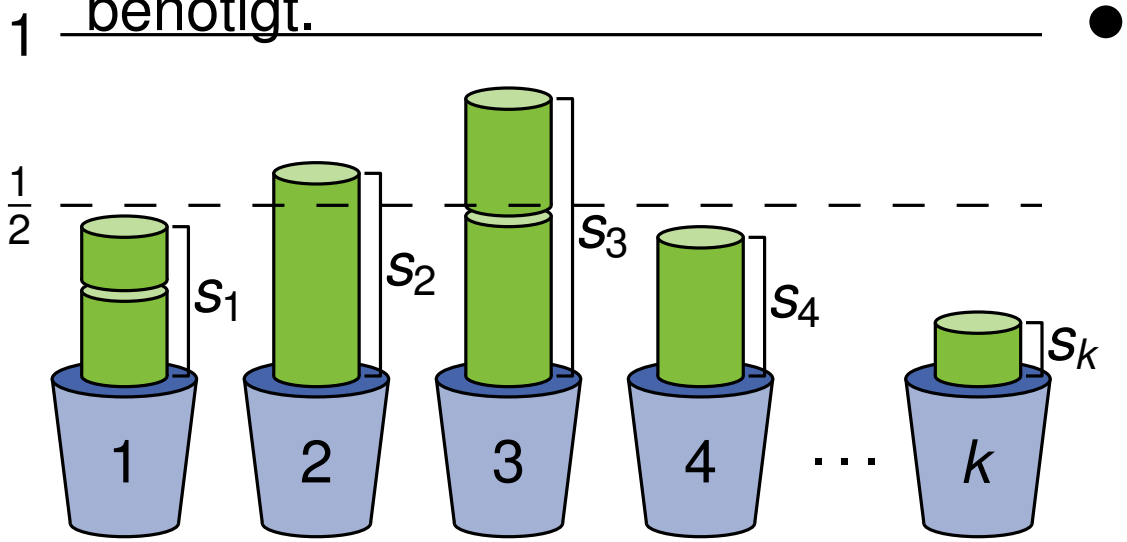
Next Fit – Approximation

Satz: Approximation **(Satz 7.5)**

NEXT FIT erfüllt $\mathcal{R}_{NF} \leq 2$.

Beweis:

- Sei $k = NF(I)$ die Anzahl an Bins, die NEXTFIT für die Instanz I benötigt.



- Sei s_i die Größe der Elemente in Bin i.

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

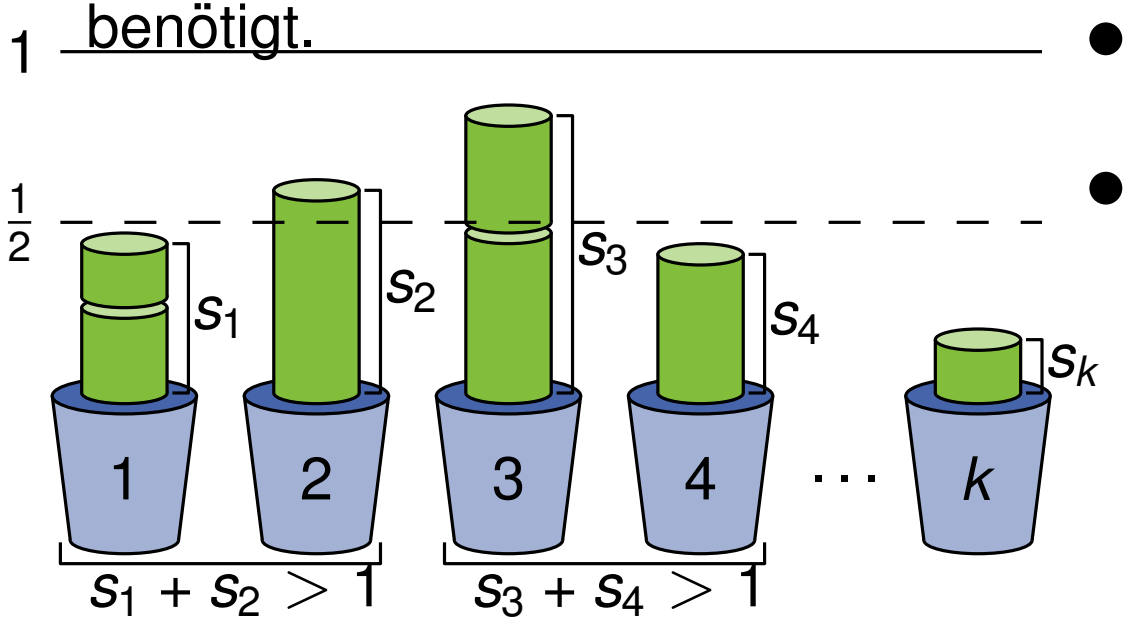
Next Fit – Approximation

Satz: Approximation **(Satz 7.5)**

NEXT FIT erfüllt $\mathcal{R}_{NF} \leq 2$.

Beweis:

- Sei $k = NF(I)$ die Anzahl an Bins, die NEXTFIT für die Instanz I benötigt.



- Sei s_i die Größe der Elemente in Bin i.
- Für zwei aufeinanderfolgende Bins gilt: $s_i + s_{i+1} > 1$

  (sonst hätten die Elemente in Bin $i + 1$ noch in Bin i gepasst)

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

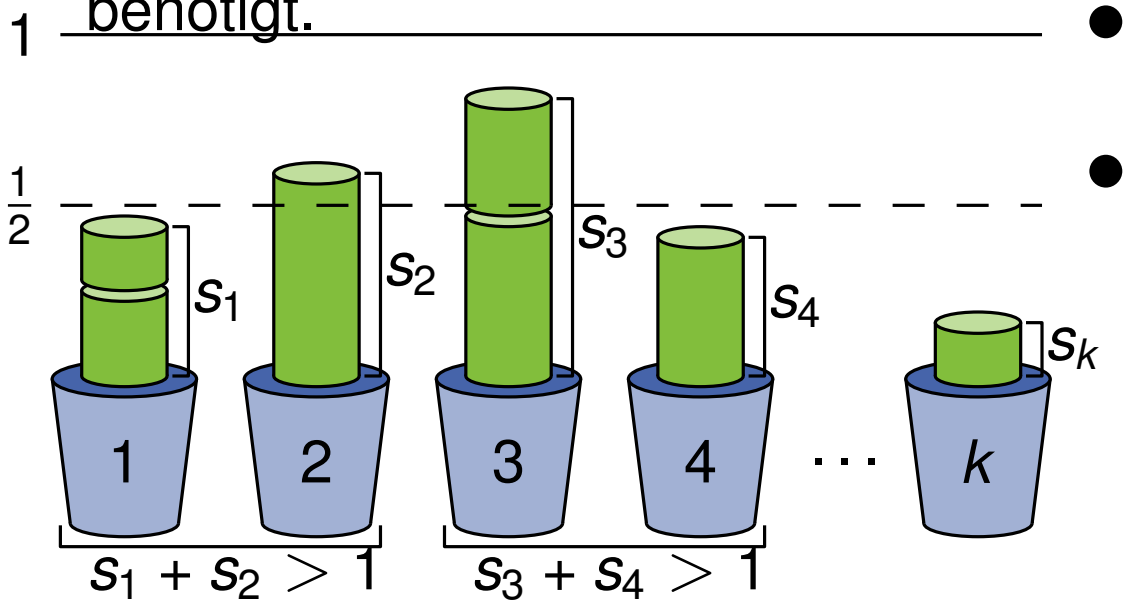
Next Fit – Approximation

Satz: Approximation **(Satz 7.5)**

NEXT FIT erfüllt $\mathcal{R}_{NF} \leq 2$.

Beweis:

- Sei $k = NF(I)$ die Anzahl an Bins, die NEXTFIT für die Instanz I benötigt.



- Sei s_i die Größe der Elemente in Bin i.
- Für zwei aufeinanderfolgende Bins gilt: $s_i + s_{i+1} > 1$

  (sonst hätten die Elemente in Bin $i + 1$ noch in Bin i gepasst)

$$\Rightarrow \sum_{i=1}^{k} s_i > \frac{k}{2} \text{ falls } k \text{ gerade bzw. } \sum_{i=1}^{k-1} s_i > \frac{k-1}{2} \text{ falls } k \text{ ungerade}$$

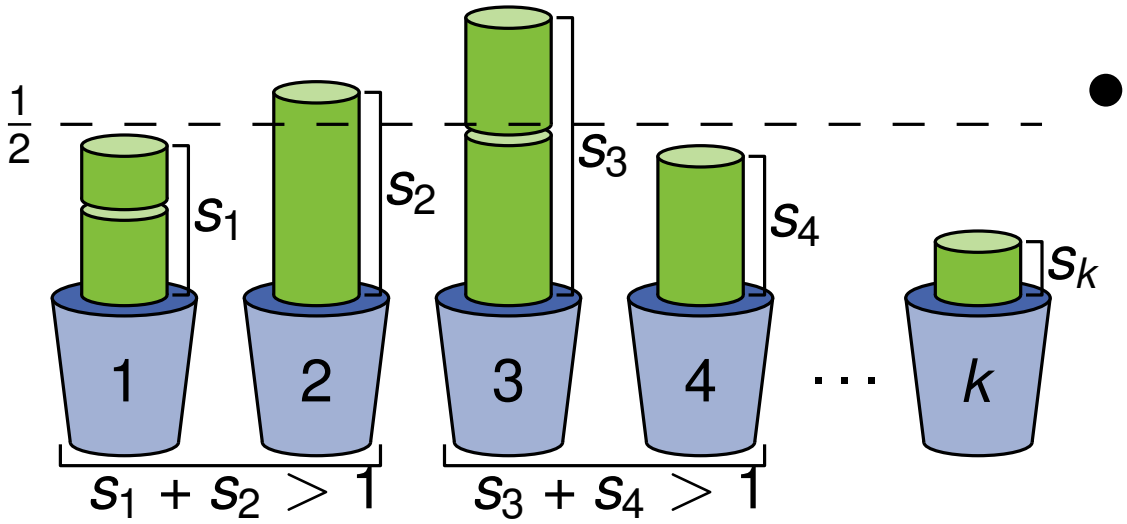
Next Fit – Approximation

Satz: Approximation **(Satz 7.5)**

NEXT FIT erfüllt $\mathcal{R}_{NF} \leq 2$.

Beweis:

- Sei $k = \text{NF}(I)$ die Anzahl an Bins, die NEXTFIT für die Instanz I benötigt.



- Sei s_i die Größe der Elemente in Bin i.
- Für zwei aufeinanderfolgende Bins gilt: $s_i + s_{i+1} > 1$

  (sonst hätten die Elemente in Bin $i + 1$ noch in Bin i gepasst)

$$\Rightarrow \sum_{i=1}^{k} s_i > \frac{k}{2} \text{ falls } k \text{ gerade bzw. } \sum_{i=1}^{k-1} s_i > \frac{k-1}{2} \text{ falls } k \text{ ungerade}$$

$$\Rightarrow \text{OPT}(I) > \frac{k-1}{2} \Rightarrow \text{NF}(I) = k < 2\text{OPT}(I) + 1 \Rightarrow \text{NF}(I) \leq 2\text{OPT}(I)$$

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Ganzzahlige Programmierung

Aufgabe

GANZZAHLIGEPROGRAMMIERUNG

$$\begin{array}{lll} \text{Minimiere}_{x} & c^T x & \} \text{ Zielfunktion} \\ \text{unter} & Ax \leq b, & \} \text{ Einschränkungen} \\ & x \geq 0, & \} \text{ Schranken} \\ & x \in \mathbb{Z}, & \end{array}$$

c, b sind Vektoren, A ist Matrix

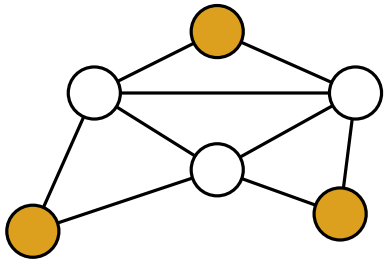
GANZZAHLIGEPROGRAMMIERUNG ist NP-schwer.

Problem UNABHÄNGIGE MENGE

Gegeben: Ungerichteter Graph $G = (V, E)$ und Zahl $k \in \mathbb{N}$.

Frage: Existiert eine unabhängige Knotenmenge $V' \subseteq V$, so dass $|V'| \geq k$ gilt?

Hinweis: $V' \subseteq V$ heißt *unabhängig*, falls für alle $u, v \in V'$ mit $u \neq v$ gilt $\{u, v\} \notin E$.



Aufgabe

Problem UNABHÄNGIGE MENGE
Gegeben: Ungerichteter Graph $G = (V, E)$ und Zahl $k \in \mathbb{N}$.
Gesucht: Möglichst große unabhängige Menge $V' \subseteq V$.
Hinweis: $V' \subseteq V$ heißt *unabhängig*, falls für alle $u, v \in V'$ mit $u \neq v$ gilt $\{u, v\} \notin E$.

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Aufgabe

Problem UNABHÄNGIGE MENGE
Gegeben: Ungerichteter Graph $G = (V, E)$ und Zahl $k \in \mathbb{N}$.
Gesucht: Möglichst große unabhängige Menge $V' \subseteq V$.
Hinweis: $V' \subseteq V$ heißt *unabhängig*, falls für alle $u, v \in V'$ mit $u \neq v$ gilt $\{u, v\} \notin E$.

Variablen: Für jeden Knoten $u \in V$ eine Variable x_u

Idee: $x_u = 1$ genau dann wenn x_u gehört zu gesuchten unabhängigen Menge.

Nebenbedingungen:

Für alle $\{u, v\} \in E$: $x_u + x_v \leq 1$

Für alle $u \in V$: $x_u \in \{0, 1\}$

Zielfunktion: $\sum_{u \in V} x_u$

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Aufgabe

Problem MAX2SAT:
Gegeben: Menge U von Variablen, Menge C von Klauseln über U, wobei jede Klausel genau **zwei** Literale enthält und eine Zahl $k \in \mathbb{N}$.
Gesucht: Wahrheitsbelegung, sodass möglichst viele Klauseln erfüllt werden.

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Aufgabe

Problem MAX2SAT:
Gegeben: Menge U von Variablen, Menge C von Klauseln über U, wobei jede Klausel genau **zwei** Literale enthält und eine Zahl $k \in \mathbb{N}$.
Gesucht: Wahrheitsbelegung, sodass möglichst viele Klauseln erfüllt werden.

Variablen: Für jede Variable v führe die Variablen x_v und $\overline{x}_v$ ein.
Für jede Klausel c führe die Variable x_c ein.

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Aufgabe

Problem MAX2SAT:
Gegeben: Menge U von Variablen, Menge C von Klauseln über U, wobei jede Klausel genau **zwei** Literale enthält und eine Zahl $k \in \mathbb{N}$.
Gesucht: Wahrheitsbelegung, sodass möglichst viele Klauseln erfüllt werden.

Variablen: Für jede Variable v führe die Variablen x_v und $\overline{x}_v$ ein.
Für jede Klausel c führe die Variable x_c ein.

Nebenbedingungen:

Für alle Variablen v: $x_v + \overline{x_v} = 1$ und $x_v \in \{0, 1\}$

Für jede Klausel c: $x_c \in \{0, 1\}$

$x_c \leq x_u + x_v$ falls $c = u \vee v$
$x_c \leq \overline{x_u} + x_v$ falls $c = \overline{u} \vee v$
$x_c \leq x_u + \overline{x_v}$ falls $c = u \vee \overline{v}$
$x_c \leq \overline{x_u} + \overline{x_v}$ falls $c = \overline{u} \vee \overline{v}$

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Aufgabe

Problem MAX2SAT:
Gegeben: Menge U von Variablen, Menge C von Klauseln über U, wobei jede Klausel genau **zwei** Literale enthält und eine Zahl $k \in \mathbb{N}$.
Gesucht: Wahrheitsbelegung, sodass möglichst viele Klauseln erfüllt werden.

Variablen: Für jede Variable v führe die Variablen x_v und $\overline{x}_v$ ein.
Für jede Klausel c führe die Variable x_c ein.

Nebenbedingungen:

Für alle Variablen v: $x_v + \overline{x_v} = 1$ und $x_v \in \{0, 1\}$

Für jede Klausel c: $x_c \in \{0, 1\}$

$$x_c \leq x_u + x_v \text{ falls } c = u \vee v$$
$$x_c \leq \overline{x_u} + x_v \text{ falls } c = \overline{u} \vee v$$
$$x_c \leq x_u + \overline{x_v} \text{ falls } c = u \vee \overline{v}$$
$$x_c \leq \overline{x_u} + \overline{x_v} \text{ falls } c = \overline{u} \vee \overline{v}$$

Zielfunktion: $\sum_{\text{Klausel } c} x_c$

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Aufgabe

Wie CLIQUE mithilfe von UNABHÄNGIGEMENGE lösen?

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner