

Theoretische Grundlagen der Informatik

Übung am 25.11.2010

INSTITUT FÜR THEORETISCHE INFORMATIK



Problem 3-Hitting-Set

Gegeben sei

- eine Grundmenge $S = \{1, \dots, n\}$
- ein Mengensystem $\mathcal{C} \subseteq 2^S$.

Jede Menge $A \in \mathcal{C}$ enthalte maximal 3 Elemente, d.h. $|A| \leq 3$.

Ein Hitting Set für \mathcal{C} ist eine Teilmenge $X \subseteq S$, so dass für jedes $A \in \mathcal{C}$ gilt $A \cap X \neq \emptyset$.

Gesucht ist ein Hitting Set minimaler Kardinalität.

Beispiel:

$$S = \{1, 2, 3, 4, 5\}$$

$$\mathcal{C} = \{c_1 = \{1, 2, 4\}, c_2 = \{1, 3\}, c_3 = \{2, 3\}, c_4 = \{3, 5\}\}$$

Aufgabe 1

Algorithmus 1: 3-Hitting-Set-Approximation

Eingabe : $S = \{1, \dots, n\}$, $\mathcal{C} \subseteq 2^S$ mit $|A| \leq 3$ für alle $A \in \mathcal{C}$

Ausgabe : Hitting Set X für \mathcal{C}

$X \leftarrow \emptyset$;

solange $\mathcal{C} \neq \emptyset$ **tue**

$A \leftarrow$ wähle eine Menge $A \in \mathcal{C}$;
 $X \leftarrow X \cup A$;
 $\mathcal{C} \leftarrow \mathcal{C} \setminus \{A \in \mathcal{C} \mid A \cap X \neq \emptyset\}$;

Geben Sie die Menge X an, die Algorithmus 1 bei Eingabe der folgenden Instanz berechnet:

$$S = \{1, 2, 3, 4, 5\}$$

$$\mathcal{C} = \{c_1 = \{1, 2, 4\}, c_2 = \{1, 3\}, c_3 = \{2, 3\}, c_4 = \{3, 5\}\}$$

In Zeile 7 werde jeweils die Menge c_i mit dem kleinsten Index i unter allen Mengen in \mathcal{C} gewählt.

Aufgabe 1

Algorithmus 2: 3-Hitting-Set-Approximation

Eingabe : $S = \{1, \dots, n\}$, $\mathcal{C} \subseteq 2^S$ mit $|A| \leq 3$ für alle $A \in \mathcal{C}$

Ausgabe : Hitting Set X für \mathcal{C}

$X \leftarrow \emptyset$;

solange $\mathcal{C} \neq \emptyset$ **do**

$A \leftarrow$ wähle eine Menge $A \in \mathcal{C}$;
 $X \leftarrow X \cup A$;
 $\mathcal{C} \leftarrow \mathcal{C} \setminus \{A \in \mathcal{C} \mid A \cap X \neq \emptyset\}$;

$$S = \{1, 2, 3, 4, 5\}$$

$$\mathcal{C} = \{c_1 = \{1, 2, 4\}, c_2 = \{1, 3\}, c_3 = \{2, 3\}, c_4 = \{3, 5\}\}$$

$$X = \{\}$$

Aufgabe 1

Algorithmus 3: 3-Hitting-Set-Approximation

Eingabe : $S = \{1, \dots, n\}$, $\mathcal{C} \subseteq 2^S$ mit $|A| \leq 3$ für alle $A \in \mathcal{C}$

Ausgabe : Hitting Set X für \mathcal{C}

$X \leftarrow \emptyset$;

solange $\mathcal{C} \neq \emptyset$ **tue**

$A \leftarrow$ wähle eine Menge $A \in \mathcal{C}$;
 $X \leftarrow X \cup A$;
 $\mathcal{C} \leftarrow \mathcal{C} \setminus \{A \in \mathcal{C} \mid A \cap X \neq \emptyset\}$;

$$S = \{1, 2, 3, 4, 5\}$$

$$\mathcal{C} = \{\mathbf{c}_1 = \{1, 2, 4\}, \mathbf{c}_2 = \{1, 3\}, \mathbf{c}_3 = \{2, 3\}, \mathbf{c}_4 = \{3, 5\}\}$$

$$X = \{1, 2, 4\}$$

Aufgabe 1

Algorithmus 4: 3-Hitting-Set-Approximation

Eingabe : $S = \{1, \dots, n\}$, $\mathcal{C} \subseteq 2^S$ mit $|A| \leq 3$ für alle $A \in \mathcal{C}$

Ausgabe : Hitting Set X für \mathcal{C}

$X \leftarrow \emptyset$;

solange $\mathcal{C} \neq \emptyset$ **tue**

$A \leftarrow$ wähle eine Menge $A \in \mathcal{C}$;
 $X \leftarrow X \cup A$;
 $\mathcal{C} \leftarrow \mathcal{C} \setminus \{A \in \mathcal{C} \mid A \cap X \neq \emptyset\}$;

$$S = \{1, 2, 3, 4, 5\}$$

$$\mathcal{C} = \{\mathbf{c}_4 = \{3, 5\}\}$$

$$X = \{1, 2, 4\}$$

Aufgabe 1

Algorithmus 5: 3-Hitting-Set-Approximation

Eingabe : $S = \{1, \dots, n\}$, $\mathcal{C} \subseteq 2^S$ mit $|A| \leq 3$ für alle $A \in \mathcal{C}$

Ausgabe : Hitting Set X für \mathcal{C}

$X \leftarrow \emptyset$;

solange $\mathcal{C} \neq \emptyset$ **tue**

$A \leftarrow$ wähle eine Menge $A \in \mathcal{C}$;
 $X \leftarrow X \cup A$;
 $\mathcal{C} \leftarrow \mathcal{C} \setminus \{A \in \mathcal{C} \mid A \cap X \neq \emptyset\}$;

$$S = \{1, 2, 3, 4, 5\}$$

$$\mathcal{C} = \{\mathbf{c}_4 = \{3, 5\}\}$$

$$X = \{1, 2, \mathbf{3}, 4, 5\}$$

Aufgabe 1

Algorithmus 6: 3-Hitting-Set-Approximation

Eingabe : $S = \{1, \dots, n\}$, $\mathcal{C} \subseteq 2^S$ mit $|A| \leq 3$ für alle $A \in \mathcal{C}$

Ausgabe : Hitting Set X für \mathcal{C}

$X \leftarrow \emptyset$;

solange $\mathcal{C} \neq \emptyset$ **tue**

$A \leftarrow$ wähle eine Menge $A \in \mathcal{C}$;
 $X \leftarrow X \cup A$;
 $\mathcal{C} \leftarrow \mathcal{C} \setminus \{A \in \mathcal{C} \mid A \cap X \neq \emptyset\}$;

$$S = \{1, 2, 3, 4, 5\}$$

$$\mathcal{C} = \{\}$$

$$X = \{1, 2, 3, 4, 5\}$$

 $X \leftarrow \emptyset;$ **solange** $\mathcal{C} \neq \emptyset$ **tue**
$$\left[\begin{array}{l} A \leftarrow \text{wähle eine Menge } A \in \mathcal{C}; \\ X \leftarrow X \cup A; \\ \mathcal{C} \leftarrow \mathcal{C} \setminus \{A \in \mathcal{C} \mid A \cap X \neq \emptyset\}; \end{array} \right.$$

Zeigen Sie, dass Algorithmus 1 ein 3-Approximationsalgorithmus ist, d.h. dass

$$\mathcal{A}(I) \leq 3 \cdot \text{OPT}(I)$$

gilt. Hierbei sei

- I eine beliebige Instanz des Problems,
- $\mathcal{A}(I)$ die Kardinalität des von Algorithmus 1 zurückgegebenen Hitting Sets
- $\text{OPT}(I)$ die Kardinalität einer optimalen Lösung.

$X \leftarrow \emptyset;$

solange $\mathcal{C} \neq \emptyset$ **tue**

$A \leftarrow$ wähle eine Menge $A \in \mathcal{C}$;
 $X \leftarrow X \cup A$;
 $\mathcal{C} \leftarrow \mathcal{C} \setminus \{A \in \mathcal{C} \mid A \cap X \neq \emptyset\}$;

Vorüberlegung

- Sei X^* eine beliebige (aber feste) optimale Lösung von I
- Dann wird in Zeile 3 eine Menge A ausgewählt mit $A \cap (X^* \setminus X) \neq \emptyset$:
- X^* ist Hitting Set, also $X^* \cap A \neq \emptyset$.
- Sei $z \in X^* \cap A \neq \emptyset$.
- Annahme: $z \in X$.
- Dann wurde die Schleife mindestens einmal durchlaufen.
- Wegen $z \in X$ hätte aber A vorher aus \mathcal{C} entfernt werden müssen.
- Widerspruch.

$X \leftarrow \emptyset;$

solange $\mathcal{C} \neq \emptyset$ **tue**

$A \leftarrow$ wähle eine Menge $A \in \mathcal{C}$;
 $X \leftarrow X \cup A$;
 $\mathcal{C} \leftarrow \mathcal{C} \setminus \{A \in \mathcal{C} \mid A \cap X \neq \emptyset\}$;

- In jedem Durchlauf von Zeile 4 werden
 - wegen $|A| \leq 3$ höchstens drei Elemente zu X zugefügt.
 - mindestens ein Element aus X^* zugefügt.
- Also gilt $\mathcal{A}(I) = |X| \leq 3 \cdot |X^*| = 3 \cdot OPT(I)$.

Problem MAX2SAT

Gegeben: Menge U von Variablen
Menge C von Klauseln über U
wobei jede Klausel genau zwei Literale enthält
Zahl $K \in \mathbb{N}$

Frage: Existiert eine Wahrheitsbelegung, die mindestens K Klauseln erfüllt?

Zeige, dass es keinen polynomiellen Approximationsalgorithmus mit absoluter Gütegarantie für MAX2SAT gibt, falls $P \neq NP$.

Aufgabe 2

Schritt 1:

Wie heißt die zugehörige Optimierungsvariante von Problem MAX2SAT?

Problem MAX2SAT - Maximalwertproblem

Gegeben: Menge U von Variablen
Menge C von Klauseln über U
wobei jede Klausel genau zwei Literale enthält

Frage: Wieviele Klauseln können maximal gleichzeitig erfüllt werden?

Problem MAX2SAT - Maximalwertproblem

Gegeben: Menge U von Variablen
Menge C von Klauseln über U
wobei jede Klausel genau zwei Literale enthält

Frage: Wieviele Klauseln können maximal gleichzeitig erfüllt werden?

- Sei $I = (U, C)$ MAX2SAT-Instanz
- Sei $opt(I)$ die Lösung von MAX2SAT-Maximalwertproblem für I
- Annahme: \exists polynomiellen Algorithmus apx mit

$$|apx(I) - opt(I)| \leq \Delta$$

für ein $\Delta \in \mathbb{N}$ und alle Instanzen I gilt.

Problem MAX2SAT - Maximalwertproblem

Gegeben: Menge U von Variablen
Menge C von Klauseln über U
wobei jede Klausel genau zwei Literale enthält

Frage: Wieviele Klauseln können maximal gleichzeitig erfüllt werden?

- Mit apx könnte man MAX2SAT-Entscheidung in Polynomialzeit lösen
- MAX2SAT ist NP-vollständig: Widerspruch zu $P \neq NP$.
- Beweis dazu: Nächste Folie

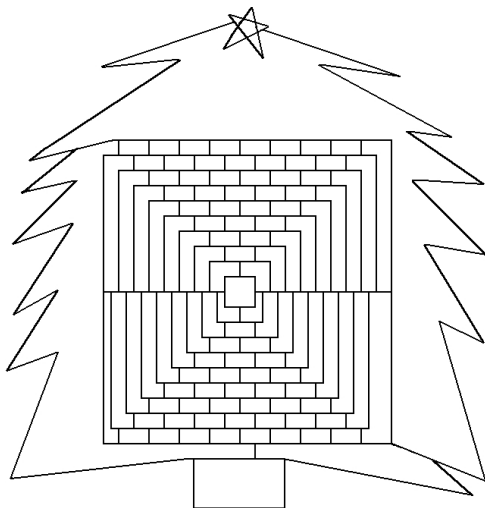
Aufgabe 2

- Sei $I = (V, C, K)$ Instanz des Entscheidungsproblems
- Konstruiere Instanz $I' = (V', C')$ des Maximalwertproblems mit
 - V' $2\Delta + 1$ mal so viele Variablen enthält wie V
 - C' ergibt in dem die Klauseln aus C $2\Delta + 1$ mal durch Substitution 'klont'
- Die Transformation ist polynomial.
- Es gilt:

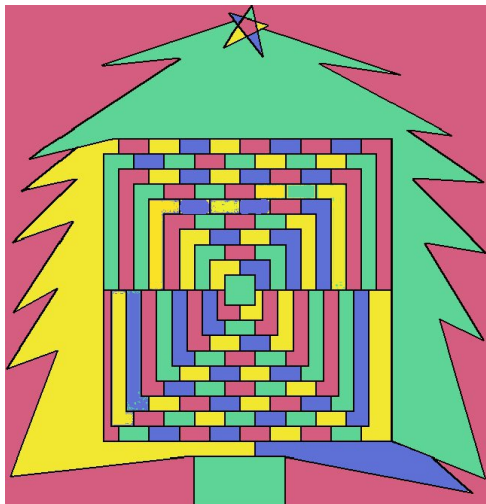
$$\begin{aligned} |apx(I') - opt(I')| &\leq \Delta \\ |apx(I') - (2\Delta + 1)opt(I)| &\leq \Delta \\ |apx(I') / (2\Delta + 1) - opt(I)| &\leq \frac{\Delta}{2\Delta + 1} < \frac{1}{2} \end{aligned}$$

- D.h man kann $opt(I)$ berechnen, in dem man $apx(I') / (2\Delta + 1)$ rundet.

Aufgabe 3



Aufgabe 3



Aufgabe 4 - Schokoladenmaschine

- Die **blauen** Kobolde wandeln **Schokolade in Schlangengift** und **Schlängengift in Schokolade** um.
- Die **roten** Kobolde erzeugen **nur Schlangengift**, wenn sie mit **beiden Händen Schlängengift** bekommen, ansonsten Schokolade.
- Die **gelben** Kobolde erzeugen **nur Schokolade**, wenn sie mit **beiden Händen Schokolade bekommen**, ansonsten Schlängengift.

Aufgabe: Welcher Trichter ist mit Schlängengift bzw Schokoladenkrümeln zu füttern, damit die Maschine Schokolade liefert.

Aufgabe 4 - Schokoladenmaschine

- Die **blauen** Kobolde wandeln **Schokolade in Schlangengift** und **Schlangengift in Schokolade** um.
- Die **roten** Kobolde erzeugen **nur Schlangengift**, wenn sie mit **beiden Händen Schlangengift** bekommen, ansonsten Schokolade.
- Die **gelben** Kobolde erzeugen **nur Schokolade**, wenn sie mit **beiden Händen Schokolade bekommen**, ansonsten Schlangengift.

Aufgabe: Welcher Trichter ist mit Schlangengift bzw Schokoladenkrümeln zu füttern, damit die Maschine Schokolade liefert.

Logische Entsprechung

Schokoladenmaschine

falsch

Schlangengift

wahr

Schokolade

nicht-Operator

blauer Kobold

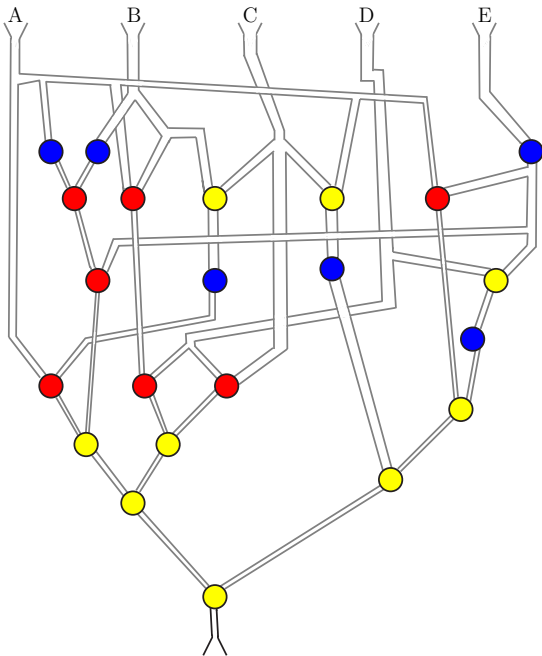
oder-Operator

roter Kobold

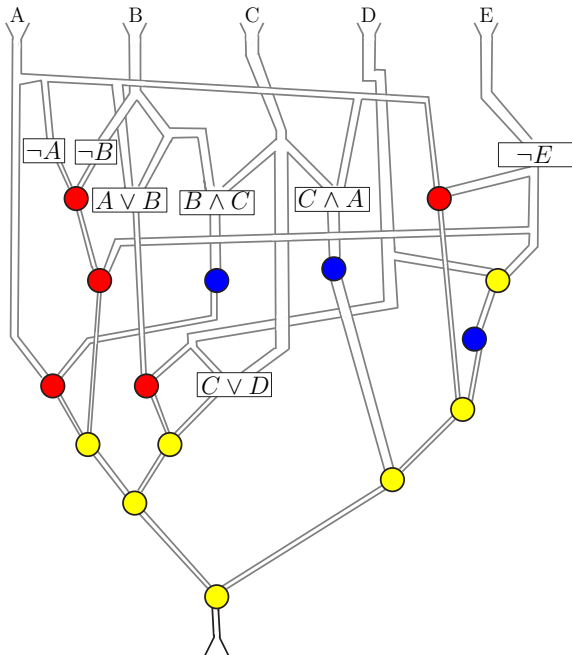
und-Operator

gelber Kobold

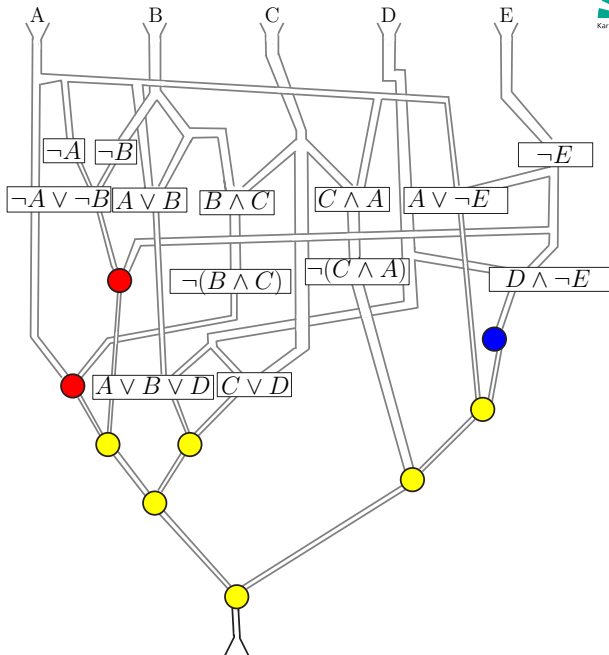
Aufgabe 4



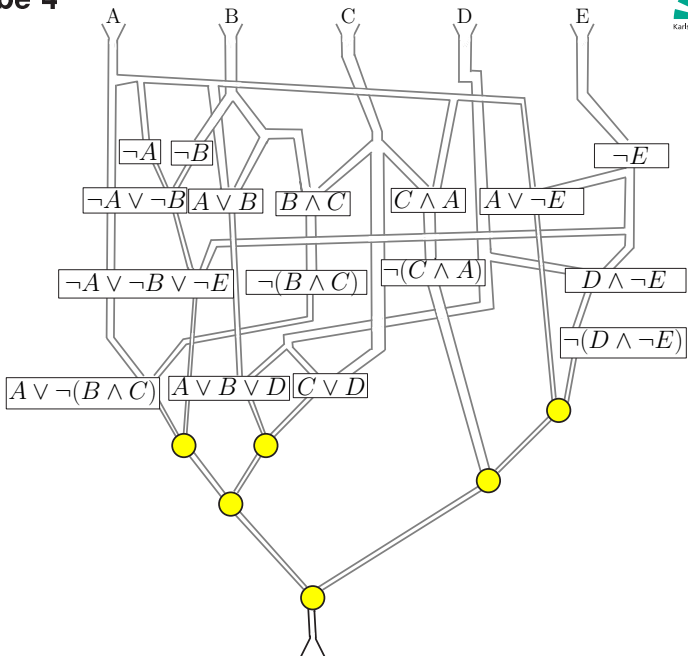
Aufgabe 4



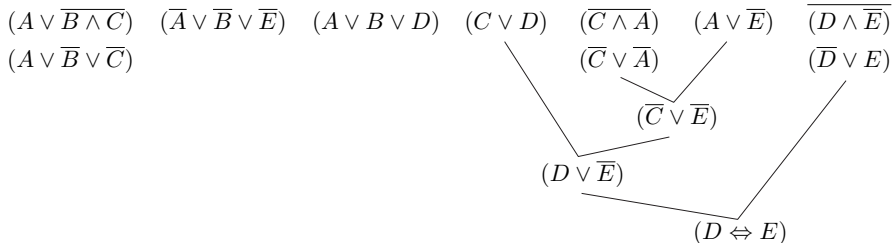
Aufgabe 4



Aufgabe 4

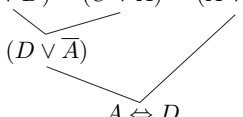


Aufgabe 4



$(A \vee \overline{B} \vee \overline{C})$ $(\overline{A} \vee \overline{B} \vee \overline{D})$ $(A \vee B \vee D)$ $(C \vee D)$ $(\overline{C} \vee \overline{A})$ $(A \vee \overline{D})$

Aufgabe 4

$$(A \vee \bar{B} \vee \bar{C}) \quad (\bar{A} \vee \bar{B} \vee \bar{D}) \quad (A \vee B \vee D) \quad (C \vee D) \quad (\bar{C} \vee \bar{A}) \quad (A \vee \bar{D})$$

$$(D \vee \bar{A})$$
$$A \Leftrightarrow D$$

$$(A \vee \bar{B} \vee \bar{C}) \quad (\bar{A} \vee \bar{B} \vee \bar{A}) \quad (A \vee B) \quad (C \vee A) \quad (\bar{C} \vee \bar{A})$$

Aufgabe 4

$$(A \vee \bar{B} \vee \bar{C}) \quad (\bar{A} \vee \bar{B} \vee \bar{A}) \quad (A \vee B) \quad (C \vee A) \quad (\bar{C} \vee \bar{A})$$
$$\swarrow \quad \searrow$$
$$A \Leftrightarrow \bar{C}$$

$$(\bar{B} \vee \bar{C}) \quad (C \vee \bar{B}) \quad (\bar{C} \vee B)$$

Aufgabe 4

$$\begin{array}{ccc} (\overline{B} \vee \overline{C}) & (C \vee \overline{B}) & (\overline{C} \vee B) \\ & \swarrow \quad \searrow & \\ & B \Leftrightarrow C & \end{array}$$

\overline{C}

Aufgabe 4

$$A \Leftrightarrow \bar{C}$$

A wahr

$$B \Leftrightarrow C$$

B falsch

$$\bar{C}$$

C falsch

$$A \Leftrightarrow D$$

D wahr

$$D \Leftrightarrow E$$

E wahr

Aufgabe 5

Wir betrachten das BIN PACKING-Problem:

Gegeben k ‚Behälter‘ mit ‚Fassungsvermögen‘ $b_1, \dots, b_k \in \mathbb{N}$ und n ‚Objekte‘ mit natürlichen ‚Gewichten‘ $w_1, \dots, w_n \leq B$. Minimiere die Anzahl ‚benötigter Behälter‘ um alle Objekte zu verpacken (d. h. finde ein möglichst kleines k und eine Abbildung $f : \{1, \dots, n\} \rightarrow \{1, \dots, k\}$, sodass für alle $j \in \{1, \dots, k\}$ gilt: $\sum_{f(i)=j} w_i \leq B$). Der Wert einer Lösung $f : \{1, \dots, n\} \rightarrow \{1, \dots, k\}$ sei k .

Aufgabe: Geben Sie einen relativen Approximationsalgorithmus \mathcal{A} für BIN PACKING mit $\mathcal{R}_{\mathcal{A}}^{\infty} \leq 2$ an (und beweisen Sie die Gütegarantie!).

Aufgabe 5

Wir betrachten das BIN PACKING-Problem:

Gegeben k ‚Behälter‘ mit ‚Fassungsvermögen‘ $b_1, \dots, b_k \in \mathbb{N}$ und n ‚Objekte‘ mit natürlichen ‚Gewichten‘ $w_1, \dots, w_n \leq B$. Minimiere die Anzahl ‚benötigter Behälter‘ um alle Objekte zu verpacken (d. h. finde ein möglichst kleines k und eine Abbildung $f : \{1, \dots, n\} \rightarrow \{1, \dots, k\}$, sodass für alle $j \in \{1, \dots, k\}$ gilt: $\sum_{f(i)=j} w_i \leq B$). Der Wert einer Lösung $f : \{1, \dots, n\} \rightarrow \{1, \dots, k\}$ sei k .

Aufgabe: Geben Sie einen relativen Approximationsalgorithmus \mathcal{A} für BIN PACKING mit $\mathcal{R}_{\mathcal{A}}^{\infty} \leq 2$ an (und beweisen Sie die Gütegarantie!).

- Die Aufgabe war verwirrend, weil die Werte b_i nicht gebraucht werden und k vorher festgelegt ist.
- Die Aufgabe wird deswegen als Bonusaufgabe gewertet, deren Punkte nicht in die Berechnung der 50% Schranke eingeht.
- Außerdem ist jede „sinnvolle“ Möglichkeit die Aufgabenstellung zu interpretieren erlaubt.

Aufgabe 5

Gegeben:

- beliebig viele Behälter mit Fassungsvermögen B
- n Objekte mit natürlichen Gewichten $0 < w_1, \dots, w_n < B$.

Aufgabe:

- Minimiere die Anzahl ‚benötigter Behälter‘ um alle Objekte zu verpacken.
- **Formal:** Finde ein möglichst kleines k und eine Abbildung

$$f : \{1, \dots, n\} \rightarrow \{1, \dots, k\} ,$$

sodass für alle $j \in \{1, \dots, k\}$ gilt:

$$\sum_{f(i)=j} w_i \leq B$$

- 15 ■ Der Wert einer Lösung $f : \{1, \dots, n\} \rightarrow \{1, \dots, k\}$ sei k .

Aufgabe 5

Aufgabe: Geben Sie einen relativen Approximationsalgorithmus \mathcal{A} für BIN PACKING mit $\mathcal{R}_{\mathcal{A}}^{\infty} \leq 2$ an und beweisen Sie die Gütegarantie.

Aufgabe 5

Aufgabe: Geben Sie einen relativen Approximationsalgorithmus \mathcal{A} für BIN PACKING mit $\mathcal{R}_{\mathcal{A}}^{\infty} \leq 2$ an und beweisen Sie die Gütegarantie.

Die **First-Fit** Heuristic hat die gewünschte Approximationsgüte:

- Bearbeite die Objekte in beliebiger Reihenfolge $1, \dots, n$.
- Starte mit n leeren Behältern b_1, \dots, b_n .
- Für jedes Objekt o_i in Reihenfolge $1, \dots, n$.
 - Finde den Behälter b_j mit kleinstem j in dem Objekt o_i Platz hat
 - Stecke o_i in b_j .
- Gib die Menge aller nichtleeren Behälter zurück.

Aufgabe 5

Aufgabe: Geben Sie einen relativen Approximationsalgorithmus \mathcal{A} für BIN PACKING mit $\mathcal{R}_{\mathcal{A}}^{\infty} \leq 2$ an und beweisen Sie die Gütegarantie.

- Sei $S = \sum_{i=1}^n w_i$.
- Jede gültige Lösung benötigt mindestens $\lceil S/B \rceil$ Behälter.
- Dies gilt also auch für jede optimale Lösung.
- Die First-Fit-Lösung erzeugt höchstens einen Behälter, der zu weniger als der Hälfte gefüllt ist.

Aufgabe 5

Aufgabe: Geben Sie einen relativen Approximationsalgorithmus \mathcal{A} für BIN PACKING mit $\mathcal{R}_{\mathcal{A}}^{\infty} \leq 2$ an und beweisen Sie die Gütegarantie.

Es gilt: Eine First-Fit-Lösung benötigt nie mehr als $\lceil 2S/B \rceil$ Behälter.

- Sei $S = \sum_{i=1}^n w_i$.
- Sei $B_j = \sum_{i \in B_j} w_i$ die Summe der Gewichte in Behälter j für eine First-Fit-Lösung.

$$\sum_{j=1}^k B_j = S$$

$$\sum_{j=1}^{k-1} B/2 + B_k \leq S$$

$$\sum_{j=1}^{k-1} 1 \leq 2(S - B_k) / B$$

$$k - 1 \leq 2(S - B_k) / B$$

$$k \leq \lceil 2S/B \rceil$$

Aufgabe 5

Aufgabe: Geben Sie einen relativen Approximationsalgorithmus \mathcal{A} für BIN PACKING mit $\mathcal{R}_{\mathcal{A}}^{\infty} \leq 2$ an und beweisen Sie die Gütegarantie.

Zusammenfassung:

- Jede optimale Lösung k^* ist mindestens $\lceil S/B \rceil$.
- Die First-Fit-Lösung k' ist höchstens $\lceil 2S/B \rceil$.
- Damit ist

$$\frac{k'}{k^*} \leq \frac{\lceil 2S/B \rceil}{\lceil S/B \rceil} \leq 2.$$

Aufgabe 6

Ein **Vertex Cover** eines ungerichteten, einfachen Graphen $G = (V, E)$ ist eine Teilmenge $C \subseteq V$ mit der Eigenschaft, dass für jede Kante $\{u, v\} \in E$ mindestens einer der beiden Knoten u, v in C enthalten ist.

Problem Vertex-Cover - Optimierungsversion

Gegeben ein Graph $G = (V, E)$, finde ein Vertex-Cover C für G von minimaler Kardinalität.

Aufgabe 6

Algorithmus 7: Vertex-Cover-Approx-1(G)

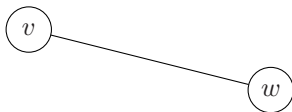
```
 $C \leftarrow \emptyset; \quad E' \leftarrow E[G];$   
solange  $E' \neq \emptyset$  tue  
     $e \leftarrow \{u, v\}$  beliebige Kante aus  $E'$ ;  
     $C \leftarrow C \cup \{u, v\}$ ;  
    entferne alle Kanten inzident zu  $u, v$  aus  $E'$ ;  
return  $C$ ;
```

Aufgabe 6

Algorithmus 8: Vertex-Cover-Approx-1(G)

```
 $C \leftarrow \emptyset; \quad E' \leftarrow E[G];$   
solange  $E' \neq \emptyset$  tue  
     $e \leftarrow \{u, v\}$  beliebige Kante aus  $E'$ ;  
     $C \leftarrow C \cup \{u, v\}$ ;  
    entferne alle Kanten inzident zu  $u, v$  aus  $E'$ ;  
return  $C$ ;
```

- Zeigen Sie, dass Algorithmus 7 keine bessere Gütegarantie als 2 geben kann.



Algorithmus 9: Vertex-Cover-Approx-1(G)

```
 $C \leftarrow \emptyset; \quad E' \leftarrow E[G];$   
solange  $E' \neq \emptyset$  tue  
     $e \leftarrow \{u, v\}$  beliebige Kante aus  $E'$ ;  
     $C \leftarrow C \cup \{u, v\}$ ;  
    entferne alle Kanten inzident zu  $u, v$  aus  $E'$ ;  
return  $C$ ;
```

- Zeigen Sie, dass Algorithmus Vertex-Cover-Approx-1(G) ein 2-Approximationsalgorithmus ist.

Aufgabe 6

Algorithmus 10: Vertex-Cover-Approx-1(G)

```
 $C \leftarrow \emptyset; \quad E' \leftarrow E[G];$   
solange  $E' \neq \emptyset$  tue  
     $e \leftarrow \{u, v\}$  beliebige Kante aus  $E'$ ;  
     $C \leftarrow C \cup \{u, v\}$ ;  
    entferne alle Kanten inzident zu  $u, v$  aus  $E'$ ;  
return  $C$ ;
```

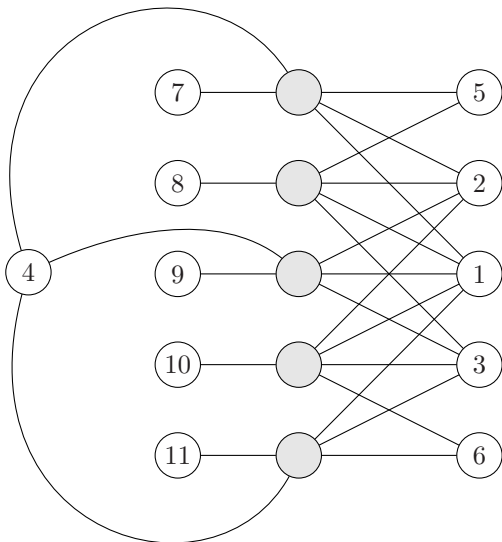
- Gegeben sei eine Instanz $G = (V, E)$
- Sei V^* ein beliebiges optimales VC von G
- Sei V' eine VC berechnet durch Vertex-Cover-Approx-1(G).
- Für jedes $v^* \in V^*$ wird höchstens einmal in Zeile 4 eine adjazente Kante ausgewählt.
- Jede Kante, die in Zeile 4 ausgewählt wird ist adjazent zu mindestens einem Knoten in V^* .
- Pro Schleifendurchlauf werden höchstens 2 Knoten zu V' zugefügt.
- Damit ist $|V'| \leq 2|V^*|$.

Algorithmus 11: Vertex-Cover-Approx-2(G)

```
 $C \leftarrow \emptyset;$   
 $E' \leftarrow E[G];$   
solange  $E' \neq \emptyset$  tue  
   $v \leftarrow$  Knoten mit maximalem Grad in  $G' = (V \setminus C, E');$   
   $C \leftarrow C \cup \{v\};$   
  entferne alle Kanten inzident zu  $v$  aus  $E';$   
return  $C;$ 
```

- Zeigen Sie mit Hilfe eines Gegenbeispiels, dass Algorithmus Vertex-Cover-Approx-2(G) kein 2-Approximationsalgorithmus ist.

Aufgabe 6



graue Knoten:
optimales
Vertex-Cover

Nummerierung:
Reihenfolge des
Algorithmus