

2020_t2_A

⚠ Disclaimer

Alles was hier drinnen steht kann Fehler enthalten! Falls dir etwas auffällt melde dich gerne auf Discord bei mir ([@xmozz](#))

A1: P und NP

Stoff: [9. Polynominalzeitreduktionen](#)

a)

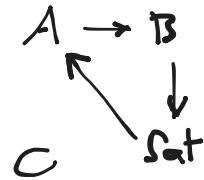
a) (12 Punkte) Seien A, B, C Ja/Nein-Probleme und n die Eingabegröße. Weiters seien A und B in NP. Nehmen Sie an, es gibt

- eine Reduktion von A nach B in Zeit $O(n^2)$,
- eine Reduktion von B nach SAT in Zeit $O(n \cdot \log n)$,
- eine Reduktion von SAT nach A in Zeit $O(n^3)$,
- eine Reduktion von C nach A in Zeit $O(n!)$.

- (i) Kreuzen Sie in den folgenden Tabellen jeweils die zutreffenden Felder an:

(je korrekter Zeile 1 Punkt, keine Minuspunkte)

A ist ...	Ja	Nein	Keine Aussage möglich
... NP-schwer	X		
... NP-vollständig	X		



B ist ...	Ja	Nein	Keine Aussage möglich
... NP-schwer	X		
... NP-vollständig	X		

C ist ...	Ja	Nein	Keine Aussage möglich
... NP-schwer			X
... NP-vollständig			X

- (ii) Für welche der Probleme A, B, C und SAT würde ein polynomieller Algorithmus zeigen, dass P=NP gilt?

A | B | SAT

- (iii) Nehmen Sie nun an, SAT kann für Eingaben der Größe n in Zeit $O(n)$ gelöst werden. Welche engste obere Schranke können Sie aus den gegebenen Informationen für die Worst-Case Laufzeit eines optimalen Algorithmus für A schließen?

$A \rightarrow B \rightarrow SAT$

$$n^2 + n \log n + k$$

$$\Rightarrow O(n^2 + (n^2 \log n^2) + (n^2 \log n^2)) = O(n^2 \log n^2)$$

$$\Rightarrow \underline{\underline{O(n^2 \cdot \log(n))}}$$

b)

- b) (8 Punkte) Im Folgenden sind zwei Probleme mit verschiedenen Zertifikaten gegeben. Welche Zertifikate sind (gemeinsam mit einem geeigneten Zertifizierer) geeignet, um zu zeigen, dass das gegebene Problem in NP ist? Kreuzen Sie Zutreffendes an. Nehmen Sie hierfür $P \neq NP$ an.

Es kann mehr als eine richtige Antwort geben.

(je Unteraufgabe: alles korrekt: 4 Punkte, ein Fehler: 2 Punkte, sonst / kein Kreuz: 0 Punkte)

- i) **Problem:** Gegeben sei ein Graph G mit n Knoten und m Kanten und eine natürliche Zahl k . Gibt es ein Independent Set von G mit k oder mehr Knoten?

Zertifikat:

- Ein Independent Set der Größe k .
- Ein leerer String.
- Ein Vertex Cover mit $n - k$ Knoten.
- Keines der Zertifikate ist geeignet.

- ii) **Problem:** Gegeben sei ein Graph G . Hat das *kleinste* Vertex Cover von G genau k Knoten?

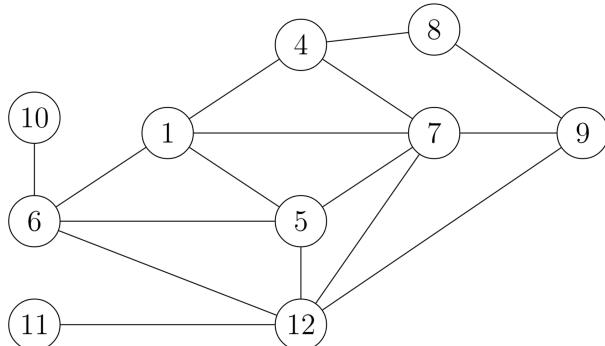
Zertifikat:

- Ein Vertex Cover mit weniger als k Knoten.
- Ein Vertex Cover mit genau k Knoten.
- Die Größe l des kleinsten Vertex Covers.
- Keines der Zertifikate ist geeignet.

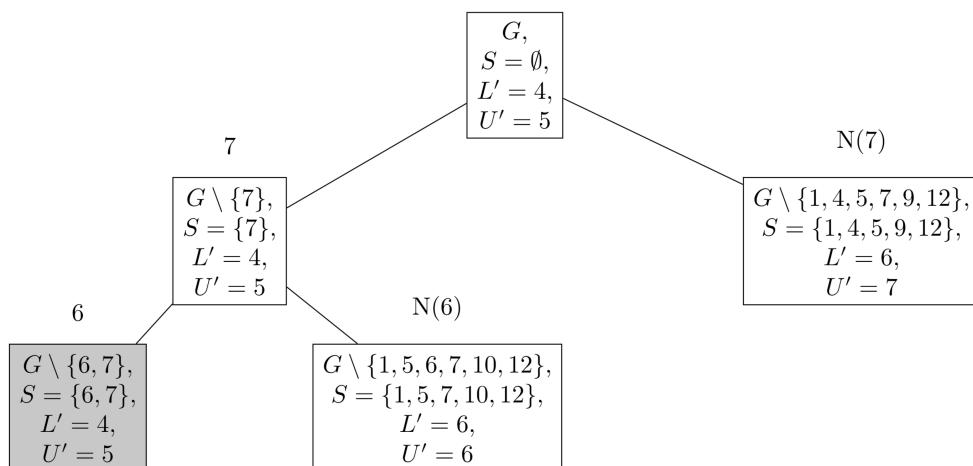
A2: Branch-and-Bound

Stoff: sem_2/AlgoDat/vo/2. Test/11. Branch and Bound

Auf dem folgenden Graphen soll mittels des Branch and Bound Algorithmus aus der Vorlesung ein minimales Vertex Cover gefunden werden.



Betrachten Sie die folgende Skizze eines partiellen Ablaufs des Branch and Bound Algorithmus. In der Skizze geben die ersten beiden Zeilen das aktuelle Teilproblem an. In der ersten Zeile wird der Graph des aktuellen Teilproblems angegeben, in der zweiten Zeile die aktuelle partielle Lösung S . Weiter steht L' für die lokale untere Schranke und U' für die lokale obere Schranke.



a)

- a) (3 Punkte) Wird bei dem Algorithmus eine globale obere Schranke oder eine globale untere Schranke verwaltet? Geben Sie den Wert der Schranke zum Zeitpunkt, wie auf der vorigen Seite abgebildet, an.

Ja, die Obere Schranke ist 5 und die untere 4

b)

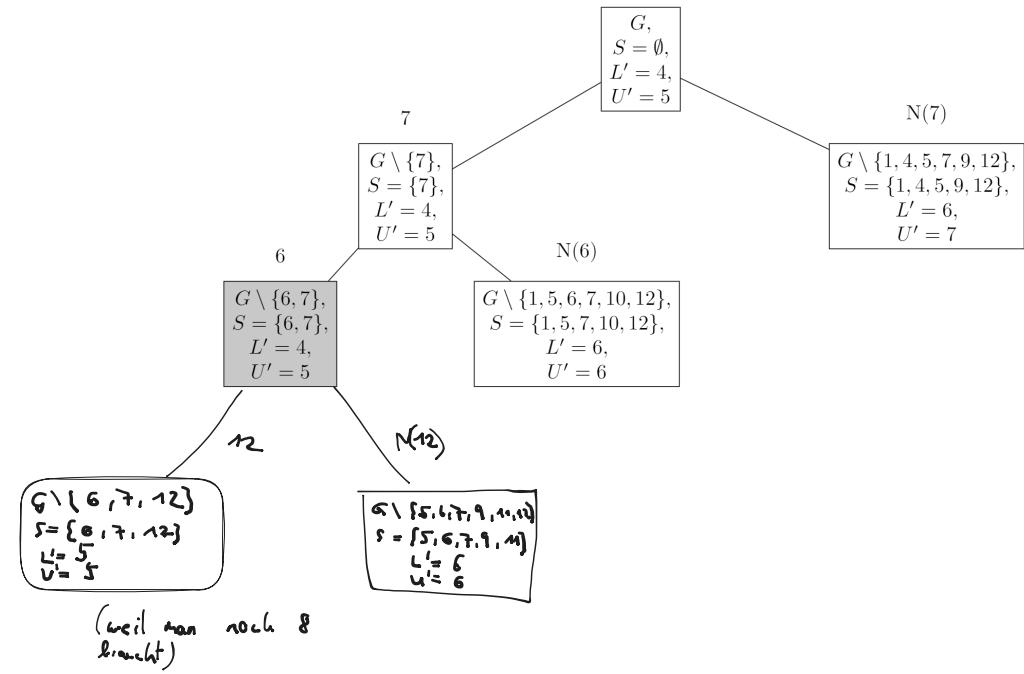
- b) (3 Punkte) Kreuzen Sie an, welche Blätter des Baums in der Skizze auf der vorigen Seite nicht mehr weiter aufgespalten werden müssen.

(alles korrekt: 3 Punkte, ein Fehler: 2 Punkte, sonst: 0 Punkte)

- 6 N(6) N(7)

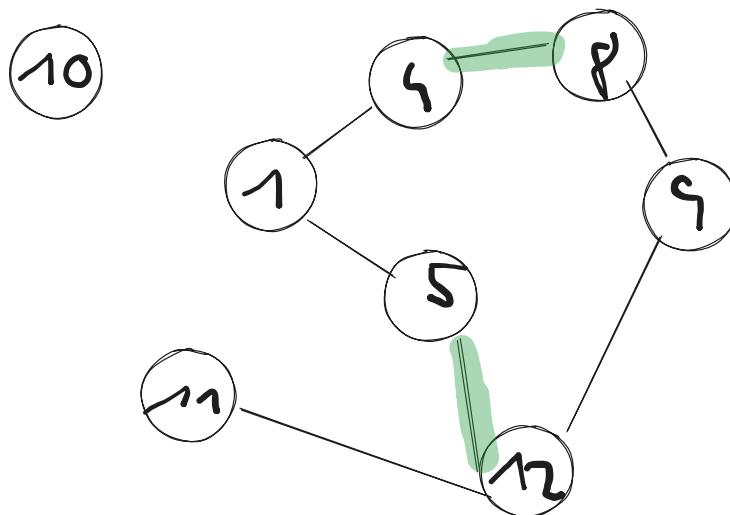
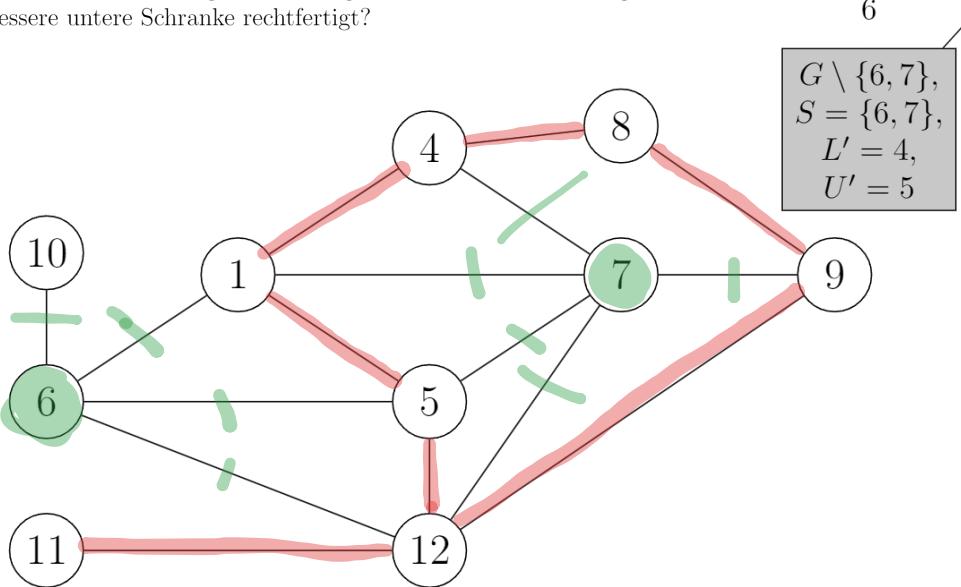
c)

- c) (7 Punkte) Vervollständigen Sie die Skizze auf der vorigen Seite, sodass sie einen vollständigen Ablauf des Branch and Bound Algorithmus darstellt.



d)

- d) (7 Punkte) Zeichnen Sie den Graphen, der im grau hinterlegten Knoten betrachtet wird. Zeichnen Sie ein maximales Matching in den Graphen ein, das die untere Schranke 4 rechtfertigt. Ist es möglich ein anderes Matching zu finden, das eine bessere untere Schranke rechtfertigt?



A3: NP-Vollständigkeit Spezialfälle

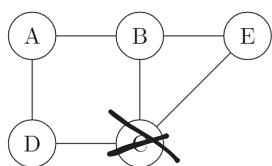
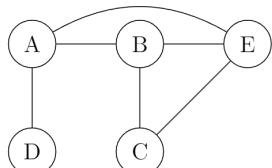
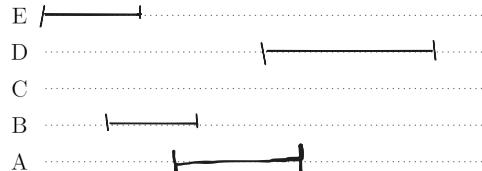
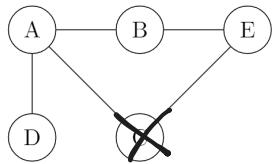
Stoff: 10. NP-Vollständigkeit Spezialfälle

a)

- a) (9 Punkte) Gegeben sind die folgenden vier Graphen. Wenn es sich bei einem Graphen um einen **Intervalgraphen** handelt, zeichnen Sie eine passende Intervallmenge auf die vorgegebenen Linien. Zum Beispiel:



Wenn ein Graph **kein Intervalgraph** ist, entfernen Sie so wenige Knoten wie möglich, damit die Intervalleigenschaft hergestellt wird und zeichnen Sie dann vom resultierenden Graphen die Intervallmenge. Streichen Sie die entfernten Knoten im Graphen eindeutig erkennbar durch.



b)

b) (5 Punkte) Kreuzen Sie an, ob folgende Aussagen wahr oder falsch sind.

(+1 Punkt für jede richtige, -1 Punkt für jede falsche und 0 Punkte für keine Antwort, keine negativen Punkte)

(i) Jeder Intervallgraph ist 3-färbbar.

Wahr Falsch

(ii) Es gibt eine polynomiale Reduktion von 3-COLOR auf 5-COLOR.

Wahr Falsch

(iii) Für Intervallgraphen liegt das Problem 3-COLOR in NP.

Wahr Falsch

(iv) Für Intervallgraphen ist das Problem 2-COLOR NP-vollständig.

Wahr Falsch

(v) Jeder Baum ist 3-färbbar.

Wahr Falsch

c)

c) (6 Punkte) Gegeben sind folgende Reduktionen von 2-COLOR nach 2-SAT. Bei jeder Reduktion wird aus einem Graphen $G = (V, E)$ eine 2-SAT Formel Φ generiert. Geben Sie für jede Reduktion an, ob diese korrekt ist. Begründung ist *nicht* notwendig.

Hinweis: Damit eine Reduktion korrekt ist, muss Φ erfüllbar sein genau dann wenn G 2-färbbar ist.

(+3 Punkte für jede richtige, -3 Punkte für jede falsche und 0 Punkte für keine Antwort, keine negativen Punkte)

(i) Für jede Kante $(v, w) \in E$ wird eine Klausel $(x_v \vee x_w)$ generiert. Für jedes Knotenpaar das nicht mit einer Kante verbunden ist wird eine Klausel $(\neg x_v \vee \neg x_w)$ generiert. Alle Klauseln konjugiert ergeben dann Φ .

Formal bedeutet das

$$\Phi = \bigwedge_{(v,w) \in E} (x_v \vee x_w) \wedge \bigwedge_{(v,w) \notin E} (\neg x_v \vee \neg x_w)$$

Korrekt Inkorrekt

(ii) Für jede Kante $(v, w) \in E$ werden zwei Klauseln generiert: $(\neg x_v \vee x_w)$ und $(x_v \vee \neg x_w)$. Alle Klauseln konjugiert ergeben dann Φ .

Formal bedeutet das

$$\Phi = \bigwedge_{(v,w) \in E} ((\neg x_v \vee x_w) \wedge (x_v \vee \neg x_w))$$

Korrekt Inkorrekt

(i)

Hier können auch beide richtig / die gleiche Farbe sein --> Nicht Richtig

(ii)

Hier wird sichergestellt, dass immer nur eins der beiden true ist --> Nachbarn haben verschiedene Farben.

A4: Dynamisches Programmieren

Stoff: 12. Dynamische Programmierung

a)

- a) (10 Punkte) Betrachten Sie das aus der Vorlesung bekannte Rucksackproblem. Gegeben sei eine Instanz mit einer Kapazität $G = 5$ und den folgenden fünf Gegenständen, die jeweils nur einmal vorhanden sind:

	Gegenstand				
	o_1	o_2	o_3	o_4	o_5
Wert w_i	7	4	4	5	6
Gewicht g_i	4	2	1	2	3

Vervollständigen Sie die untenstehende Tabelle M , sodass der Eintrag $M[i, g]$ in Zeile i und Spalte g gerade dem Wert $OPT(i, g)$ entspricht, der sich mit den ersten i Gegenständen o_1, \dots, o_i und einem Rucksack der Kapazität g erreichen lässt.

		Kapazität					
		0	1	2	3	4	5
Gegenstände	\emptyset	0	0	0	0	0	0
	$\{o_1\}$	0	○	○	○	7	7
	$\{o_1, o_2\}$	0	○	4	4	7	7
	$\{o_1, o_2, o_3\}$	0	4	4	4	7	13
	$\{o_1, o_2, o_3, o_4\}$	0	4	5	5	9	13
	$\{o_1, o_2, o_3, o_4, o_5\}$	0	4	5	6	10	14

- (i) Was ist der optimale Wert einer Lösung für einen Rucksack mit Kapazität $G = 4$? Wird der Gegenstand o_5 für diese optimale Lösung benötigt?

Wert:

10

Wird o_5 benötigt (ja/nein)?

ja

- (ii) Wie errechnet sich der Eintrag $M[i, g]$ unter der Annahme, dass $g_i \leq g$?

$$M[i, g] = \max \left\{ M[i-1, g], w_i + M[i, g - g_i] \right\}$$

b)

- b) (8 Punkte) Sei nun eine weitere Instanz des Rucksackproblems mit Kapazität $G = 7$ und den folgenden sechs Gegenständen gegeben.

	Gegenstand					
	o_1	o_2	o_3	o_4	o_5	o_6
Wert	1	4	7	3	3	8
Gewicht	2	1	4	1	2	3

Die Wertetabelle M nach Ausführung des Dynamischen Programms lautet wie folgt:

	Kapazität								
	0	1	2	3	4	5	6	7	
\emptyset	0	0	0	0	0	0	0	0	
$\{o_1\}$	0	0	1	1	1	1	1	1	
$\{o_1, o_2\}$	0	4	4	5	5	5	5	5	
$\{o_1, o_2, o_3\}$	0	4	4	5	7	11	11	12	
$\{o_1, o_2, o_3, o_4\}$	0	4	7	7	8	11	14	14	
$\{o_1, o_2, o_3, o_4, o_5\}$	0	4	7	7	10	11	14	14	
$\{o_1, o_2, o_3, o_4, o_5, o_6\}$	0	4	7	8	12	15	15	18	

- (i) Kreisen Sie in der Tabelle all jene Felder ein, die der Algorithmus **Find-Solution(M)** aus der Vorlesung bei der Berechnung der Lösungsmenge S ausliest und verwendet.
(ii) Geben Sie die Menge der Gegenstände in der Lösungsmenge S an.

$$S = \{ \quad O_3, O_4, O_5, O_6 \quad \} \quad \}$$

c)

- c) (2 Punkte) Sei eine Instanz des Rucksackproblems mit n Gegenständen und Kapazität G des Rucksacks gegeben. Kreuzen Sie *alle* gültigen Laufzeitschranken für den Algorithmus **Find-Solution(M)**, der die ausgefüllte Wertetabelle M als Eingabe erhält.

(2 Punkte, wenn alle Kreuze korrekt sind, 1 Punkt bei genau einem Fehler, ansonsten 0 Punkte)

- $O(n)$
- $O(G)$
- $O(\min\{n, G\})$
- $O(n \cdot G)$
- $O(n + G)$

A5: Approximationsalgorithmen

Stoff: 13. Approximation

a)

- a) (15 Punkte) Für einen gerichteten Graphen $G = (V, E)$ wird eine größtmögliche (bezüglich Kardinalität) Teilmenge $F \subseteq E$ von Kanten gesucht, die es erlaubt eine topologische Sortierung für den reduzierten Graphen $G' = (V, F)$ zu finden.

(i)

- (i) Geben Sie für dieses Problem einen polynomiellen Approximationsalgorithmus mit Gütegarantie $1/2$ an. Eine eindeutige Beschreibung in wenigen Wörtern reicht, Pseudocode ist nicht notwendig.

Hinweis: Betrachten Sie eine beliebige lineare Ordnung der Knotenmenge. Welche zwei Arten von Kanten können Sie unterscheiden?

Definiere Vorwärts und Rückwärts, Vorwärts wenn die mehrheit der Kanten in die Richtung gehen, Rückwärtskanten wenn sie in die andere Richtung zurück auf eine schon besuchte Kante gehen.

Wähle eine beliebige lineare Ordnung der Knoten. Behalte alle Kanten, die mit dieser Ordnung vorwärts gehen. Entferne die Rückwärtskanten. Die verbleibenden Kanten ergeben einen azyklischen Graphen.

Gütegarantie: Mindestens die Hälfte aller Kanten bleibt erhalten.

(ii)

- (ii) Argumentieren Sie, warum Ihr Algorithmus immer eine korrekte Lösung liefert und die geforderte Gütegarantie einhält.

Gültig, weil alle Vorwärtskanten alleine erzeugen keinen Zyklus.

Das funktioniert immer mit Gütegarantie $\frac{1}{2}$ da, falls mehr als die Hälfte der Kanten Rückwärts gehen wir einfach diese Rückwärtskanten als Vorwärtskanten definieren könnten und schon haben wir wieder mehr als die Hälfte der Kanten in unserer Lösung.

b)

b) (5 Punkte) Kreuzen Sie die richtigen Aussagen an:

(+1 Punkt für jede richtige, -1 Punkt für jede falsche und 0 Punkte für keine Antwort, keine negativen Punkte)

- Sei v^* der Wert einer optimalen Lösung eines Minimierungsproblems. Ein Approximationsalgorithmus mit Gütegarantie α liefert immer eine Lösung mit einem Wert $\leq v^* + \alpha$.

Wahr Falsch
- Ein Approximationsalgorithmus mit Gütegarantie 2 kann eine Lösung zurückliefern, deren Wert 2% über dem optimalen Lösungswert liegt.

Wahr Falsch
- Ein Approximationsalgorithmus mit Gütegarantie 1 liefert immer eine optimale Lösung.

Wahr Falsch
- Sei v^* der Wert einer optimalen Lösung eines Maximierungsproblems. Ein Approximationsalgorithmus mit Gütegarantie α liefert immer eine Lösung mit einem Wert $\geq v^* \cdot \alpha$.

Wahr Falsch
- Die aus der Vorlesung bekannte Spannbaumheuristik löst das *Euklidische Traveling Salesperson Problem* mit Gütegarantie 3.

Wahr Falsch

Appendix

