Schranken für Mengenüberdeckungsprobleme

Diplomarbeit

vorgelegt von

Markus Thiemann

geboren am 30. September 1982 in Georgsmarienhütte

Januar 2009

Angefertigt im
Institut für Numerische und Angewandte
Mathematik
der
Georg-August Universität Göttingen

Inhaltsverzeichnis

Einleitung **v**

1 Mengenüberdeckungsprobleme **1**
1.1 Formulierung des Problems 2
1.2 Reduktionslemma für (SCP) 5
1.3 Mengenüberdeckungsprobleme mit C1P 6

2 Untere Schranken für Mengenüberdeckungsprobleme **10**
2.1 Relaxationen von (SCP) mit C1P-Matrizen 11
2.1.1 Das Problem (SCPl) 12
2.1.2 Das Problem (S$\mathring{\text{C}}$Pl) 13
2.1.3 Güte der von (SCPl) und (S$\mathring{\text{C}}$Pl) erzeugten unteren Schranken 15
2.2 Duale Schranke 20
2.3 Hierarchie für die Güte der unteren Schranken 22

3 Obere Schranken für Mengenüberdeckungsprobleme **25**
3.1 Blockauswahl und das Problem (SCPu(ι)) 26
3.2 Blockauswahl „best-cover“ 28
3.2.1 Güte von Algorithmus 1 für ungewichtete (SCP) 31
3.3 Blockauswahl basierend auf Greedy-Heuristiken 36
3.3.1 Greedy-Heuristik 37
3.3.2 Greedy-Blockauswahl 39
3.3.3 Güte von Algorithmus 3 41
3.3.4 Andere Gewichtungsfunktionen in Algorithmus 3 44
3.3.5 Vergleich von „best-cover“- und Greedy-Blockauswahl 45

3.4 Blockauswahl für beliebige zulässige Lösungen von (SCP) 48
3.5 Erweiterung einer Relaxationslösung durch Blockauswahl 51
3.5.1 Vergleich von Algorithmus 6 und Algorithmus 7 54

4 Numerische Ergebnisse **59**
4.1 Vergleich von Algorithmus 2 und Algorithmus 3 60
4.1.1 Vergleich von Algorithmus 2 und Algorithmus 4 für das ungewichtete (SCP) 65
4.2 Untersuchung von (SCPl) und (S$\mathring{\text{C}}$Pl) 67
4.3 Vergleich verschiedener oberer und unterer Schranken für gewichtete (SCP) 69

5 Fazit **73**
5.1 Untere Schranken 73
5.1.1 Duale Schranke 73
5.1.2 (SCPl) 74
5.1.3 (S$\mathring{\text{C}}$Pl) 74
5.2 Obere Schranken 74
5.2.1 Greedy-Heuristik 75
5.2.2 Algorithmus 3 und Algorithmus 4 75
5.2.3 Algorithmus 5 76
5.2.4 Algorithmus 6 77
5.2.5 Algorithmus 7 77
5.2.6 Algorithmus 1 78

Literaturverzeichnis **79**

Algorithmenverzeichnis **81**

Erklärung **82**

Einleitung

Viele Probleme aus der kombinatorischen Optimierung und ihrer praktischen Anwendung lassen sich durch Mengenüberdeckungsprobleme als ganzzahlige Programme modellieren. Die entstehenden ganzzahligen Probleme können im Allgemeinen nicht mehr exakt gelöst werden, was im Laufe der letzten 40 Jahre zur Entwicklung zahlreicher Verfahren zur Bestimmung von Näherungslösungen sowie oberen und unteren Schranken für Mengenüberdeckungsprobleme geführt hat.

Die Beobachtung, dass die Struktur der Koeffizientenmatrix eines Mengenüberdeckungsproblems in manchen Fällen auf natürliche Weise aus dem zu modellierenden Problem hervorgeht, führte uns zu der Idee, diese spezielle Gestalt der Matrix für die Berechnung von oberen und unteren Schranken des Mengenüberdeckungsproblems auszunutzen. Das heißt zum einen, dass wir bekannte Verfahren erweitern und ändern werden, um die Gestalt der Koeffizientenmatrix des Problems zur effizienteren Berechnung von Schranken zu nutzen. Zum anderen entwickeln wir durch eine direkte Analyse der speziellen Struktur neue Verfahren zur Berechnung von Schranken für diese Probleme.

Konkret behandeln wir in dieser Arbeit Mengenüberdeckungsprobleme, bei denen in der Koeffizientenmatrix des ganzzahligen Programms in vielen Zeilen jeweils nur wenige Blöcke von aufeinanderfolgenden Einsen auftreten. Eine solche Struktur entsteht z. B. bei der Modellierung des Problems, neue Bahnhöfe in einem gegebenen Schienennetz zu platzieren („stop location problem“ , siehe [24]).

Die Koeffizientenmatrix mit der beschriebenen Struktur hat annähernd die *consecutive ones property* (C1P). Hat die Matrix eines Mengenüberdeckungsproblems C1P, so sind effiziente Lösungsmethoden für das Problem bekannt. Wir werden daher die Berechnung von Schranken für ein gegebenes Mengenüberdeckungsproblem auf die Lösung eines Problems mit *consecutive ones property* zurückführen.

Kapitel 1

Mengenüberdeckungsprobleme

Das Mengenüberdeckungsproblem (englisch: *set covering problem*) ist eines der wichtigsten und am besten studierten Probleme der kombinatorischen Optimierung. Der Grund hierfür liegt darin, dass das Problem sehr allgemein ist und viele andere kombinatorische Optimierungsprobleme durch Mengenüberdeckungsprobleme modelliert werden können. Zu nennen sind hier z. B. Mengenpackungsprobleme, Kantenüberdeckung und Knotenpackung auf Graphen sowie Rucksackprobleme.

Zusätzlich können viele praktische Probleme als Mengenüberdeckungsprobleme modelliert werden, z. B. Personalumlaufplanung im Flug- und Busverkehr [18, 22], Verkehrsplanung [24], Standortplanung von Krankenhäusern und Feuerwehren [27, 30] und industrielle Anwendungen [28, 29]. Weitere Anwendungen sowie einen Literaturüberblick zum Thema Mengenüberdeckungsprobleme sind in der kommentierten Bibliographie von Ceria et al. [8] zu finden.

Im ersten Teil dieses Kapitels werden wir das Mengenüberdeckungsproblem formulieren, über die Lösbarkeit des Problems diskutieren und einen Überblick über Lösungsansätze in der Literatur geben. Im zweiten Abschnitt wird ein Reduktionslemma für das Problem vorgestellt. Im letzten Teil dieses Kapitels beschäftigen wir uns mit Mengenüberdeckungsproblemen mit *consecutive ones property*.

1.1 Formulierung des Problems

Zunächst werden wir das Mengenüberdeckungsproblem formal definieren.

Es sei $M = \{1, ..., m\}$ eine endliche, nichtleere Menge und es sei $\mathcal{S} = \{S_j\}_{j \in N}$ für $N = \{1, ..., n\}$ eine gegebene Familie von Teilmengen von M, also $S_j \subseteq M$ $\forall j \in N$. Wir sagen, dass eine Teilmenge $F \subseteq N$ die Menge M *überdeckt*, falls $\bigcup_{j \in F} S_j = M$ und nennen F dann eine zulässige *Überdeckung* von M. Seien nun zusätzlich Kosten $c_j \in \mathbb{R}$ für alle Teilmengen S_j gegeben. Die Aufgabe des *Mengenüberdeckungsproblems* besteht darin, eine kostenminimale Überdeckung von M zu finden.

Das Mengenüberdeckungsproblem kann als ganzzahliges Programm beschrieben werden. Sei dazu $A \in \mathbb{B}^{m \times n}$ eine 0-1-Matrix, dessen Elemente (a_{ij}) für $i \in M$, $j \in N$ definiert sind durch:

$$a_{ij} := \begin{cases} 1, & \text{falls } i \in S_j \\ 0, & \text{sonst.} \end{cases}$$

Außerdem definieren wir noch eine Entscheidungsvariable $x \in \mathbb{B}^n$, die angibt, ob eine Teilmenge S_j für die Überdeckung F ausgewählt wurde oder nicht. Für alle $j \in N$ sei

$$x_j := \begin{cases} 1, & \text{falls } j \in F \\ 0, & \text{sonst.} \end{cases}$$

Des Weiteren seien Kosten $c_j \in \mathbb{R}$ für alle Mengen S_j, $j \in N$ gegeben. Dann kann das Mengenüberdeckungsproblem auf folgende Weise als ganzzahliges Programm definiert werden:

$$\begin{aligned} \text{minimiere} \quad & z := \sum_{j \in N} c_j x_j & \\ \text{so dass} \quad & \sum_{j \in N} a_{ij} x_j \geq 1 & \forall i \in M \\ & x_j \in \{0, 1\} & \forall j \in N. \end{aligned}$$

Für das Mengenüberdeckungsproblem können wir o. B. d. A. annehmen, dass $c_j > 0 \, \forall j \in N$. Denn falls es eine Menge S_k mit Kosten $c_k < 0$ gibt, so wird diese Menge in jeder kostenminimalen Überdeckung vorkommen. Falls $c_k = 0$, so kann die Menge S_k in jede kostenminimale Überdeckung aufgenommen werden, ohne deren Kosten zu vergrößern. In beiden Fällen können wir daher $x_k = 1$ setzen und das Problem

$$\begin{aligned} &\text{minimiere} && \sum_{j \in N \setminus \{k\}} c_j x_j && \\ &\text{so dass} && \sum_{j \in N \setminus \{k\}} a_{ij} x_j \geq 1 && \forall i \in M \setminus S_k \\ & && x_j \in \{0,1\} && \forall j \in N \setminus \{k\} \end{aligned}$$

lösen.

Nun folgt die Definition des Mengenüberdeckungsproblems, mit der wir im Folgenden arbeiten werden. Dabei sei $\underline{s}_l$ der Vektor mit Länge l, in dem jeder Eintrag s ist.

Definition 1.1.1. *Es seien $A \in \{0,1\}^{m \times n}$, $c \in \mathbb{R}_+^n$ und $\underline{1}_m \in \mathbb{N}^m$. Das* Mengenüberdeckungsproblem *ist gegeben durch:*

$$\begin{aligned} & && \textit{minimiere} && z = c^t x && (1.1) \\ & \textbf{\textit{(SCP)}} && \textit{so dass} && Ax \geq \underline{1}_m && (1.2) \\ & && && x \in \mathbb{B}^n && (1.3) \end{aligned}$$

Die Matrix A nennen wir auch Überdeckungsmatrix *des Mengenüberdeckungsproblems.*

Bemerkung 1.1.2. *In der Überdeckungsmatrix A des Mengenüberdeckungsproblems entsprechen die Zeilen den zu überdeckenden Elementen $\{1, ..., m\}$ und die Spalten den Teilmengen S_j, $j \in \{1, ..., n\}$, von M. Eine optimale Lösung von (SCP) besteht also aus einer kostenminimalen Teilmenge der Spalten, so dass alle Zeilen von A durch diese Spalten überdeckt werden.*

Notation 1.1.3. *Falls die Kosten für alle Spalten gleich sind, so sprechen wir von einem* ungewichteten (SCP) *oder* ungewichteten Mengenüberdeckungspro-

blem *und schreiben* $c = \underline{1}_n$. *Andernfalls nennen wir das Problem* gewichtetes Mengenüberdeckungsproblem *oder* gewichtetes (SCP).

Mengenüberdeckungsprobleme gehören zu der Klasse der NP-schweren Probleme:

Satz 1.1.4. *Das ungewichtete Mengenüberdeckungsproblem ist NP-schwer.*

Beweis. Ein Beweis kann in [14] gefunden werden. □

Bemerkung 1.1.5. *Da das ungewichtete (SCP) und damit insbesondere das gewichtete Mengenüberdeckungsproblem nach Satz 1.1.4 NP-schwer ist, können wir nicht erwarten, dass es einen polynomiellen Algorithmus zur Lösung von (SCP) gibt.*

Zur Berechnung optimaler Lösungen von (SCP) werden daher oft Branch & Bound-Techniken genutzt (siehe z. B. [1, 3, 6]). Falls große Instanzen für ein Mengenüberdeckungsproblem gegeben sind, so führen bekannte Verfahren im Allgemeinen nicht mehr in annehmbarer Zeit zu einer Optimallösung des Problems. In diesem Fall können Näherungslösungen durch heuristische Methoden, wie z. B. Lagrange-Heuristiken [4, 9] oder Greedy-Heuristiken [10, 13], gefunden werden. Ein Überblick weiterer Methoden, Näherungs- und Optimallösungen von Mengenüberdeckungsproblemen zu berechnen, findet sich in [7] und [8].

Es folgen die Definitionen von Kandidatenmenge einer Zeile und Überdeckung einer Spalte.

Definition 1.1.6. *Es sei* $A \in \mathbb{B}^{m \times n}$, $M = \{1, ..., m\}$, $N = \{1, ..., n\}$.

1. *Die Menge*
$$cover(j) := \{i \in M : a_{ij} = 1\}$$
heißt Überdeckung *oder* Menge der überdeckten Zeilen *von Spalte* $j \in N$.

2. *Die Menge*
$$cand(i) := \{j \in N : a_{ij} = 1\}$$
heißt Kandidatenmenge *von Zeile* $i \in M$. *Die Elemente der Menge* $cand(i)$ *nennen wir auch die* Kandidaten *der Zeile* i.

Notation 1.1.7. *Die Anzahl der Zeilen $|cover(j)|$, die von Spalte j überdeckt werden, nennen wir die* Überdeckungszahl *von j.*

Bemerkung 1.1.8.

- *Die Bedingung (1.2) aus Definition 1.1.1 kann mit Definition 1.1.6 auch geschrieben werden als:*

$$\sum_{j \in cand(i)} x_j \geq 1 \qquad \forall i \in M \tag{1.4}$$

- *Die Überdeckung $cover(j)$ einer Spalte j entspricht der Menge S_j aus der Familie $\mathcal{S}$ der Teilmengen von M.*

1.2 Reduktionslemma für (SCP)

Die Überdeckungsmatrix A eines Mengenüberdeckungsproblems besitzt bei praktischen Anwendungen im Allgemeinen sehr viele Zeilen und Spalten. Es ist daher wichtig, das Problem vor der Anwendung von Algorithmen so weit wie möglich zu reduzieren, um eine kürzere Laufzeit der Algorithmen zu erreichen. Das heißt, redundante Bedingungen des Problems sollen gefunden und effizient entfernt werden. Jede optimale Lösung des reduzierten Problems soll dann auch eine Optimallösung des ursprünglichen Problems sein. Das folgende Reduktionslemma für Mengenüberdeckungsprobleme gibt Regeln für Zeilen- und Spaltenreduktionen der Überdeckungsmatrix A an. Die Regeln stammen von Toregas und ReVelle [26].

Lemma 1.2.1. ***(Reduktionslemma für (SCP))***
Gegeben sei ein (SCP) mit Überdeckungsmatrix $A \in \mathbb{B}^{m \times n}$ und Kosten $c \in \mathbb{R}_+$. Es sei $N = \{1, ..., n\}$ die Menge der Spalten und $M = \{1, ..., m\}$ die Menge der Zeilen von A. Dann gilt:

1. *(SCP) hat eine Lösung $\Leftrightarrow \forall i \in M : \ cand(i) \neq \varnothing$.*
2. *Wenn es ein $i \in M$ und ein $j \in N$ gibt mit $cand(i) = \{j\} \Rightarrow x_j = 1$ in jeder zulässigen Lösung x von (SCP). Die Spalte j und alle Zeilen $i \in cover(j)$ können aus A entfernt werden.*
3. *Falls $cand(i) \subseteq cand(k)$ für $i, k \in M \Rightarrow$ entferne Zeile k aus A.*

4. Falls cover$(k) \subseteq$ cover(l) für $k, l \in N$ und $c_k \geq c_l \Rightarrow$ es gibt eine optimale Lösung x von (SCP) mit $x_k = 0$. Die Spalte k kann aus A entfernt werden.

Beweis. Ein Beweis findet sich in [26].

□

Bemerkung 1.2.2. *Nach Lemma 1.2.1 ist ein Mengenüberdeckungsproblem mit Überdeckungsmatrix A genau dann lösbar, wenn es in A keine Nullzeile gibt. Wir werden im Folgenden für alle betrachteten (SCP) implizit vorraussetzen, dass die Überdeckungsmatrix keine Nullzeile hat.*

Bemerkung 1.2.3. *Die Regeln aus Lemma 1.2.1 müssen mit Bedacht implementiert werden, da eine gradlinige Programmierung bei großen Probleminstanzen zu einem hohen Zeitaufwand führen kann. Andererseits führt eine Anwendung der Reduktionsregeln bei großen Instanzen zu einem erheblich kleineren Problem. Heuristiken und andere Näherungsverfahren liefern für das reduzierte Problem im Allgemeinen bessere Lösungen als für das ursprüngliche Problem.*

Bei ungewichteten Mengenüberdeckungsproblemen empfiehlt sich die Anwendung der Reduktionsregeln besonders, denn die Voraussetzung $c_k \geq c_l$ für $k, l \in N$ aus Regel 4 ist auf Grund der Kostengleichheit aller Spalten immer erfüllt.

1.3 Mengenüberdeckungsprobleme mit C1P

In diesem Abschnitt betrachten wir Mengenüberdeckungsprobleme, bei denen die Überdeckungsmatrix die *consecutive ones property* hat. Für solche Mengenüberdeckungsprobleme kann mit polynomiellen Algorithmen eine Optimallösung gefunden werden.

Definition 1.3.1. *(siehe z. B. [23]) Es sei $A \in \mathbb{B}^{m \times n}$. A hat die* consecutives ones property (C1P)*, falls es eine Permutation der Spalten von A gibt, so dass für alle Zeilen $k \in \{1, ..., m\}$ von A und alle $j_1, j_2 \in \{1, ..., n\}$ gilt:*

$$a_{kj_1} = 1,\ a_{kj_2} = 1 \Rightarrow a_{kj} = 1 \text{ für alle } j_1 \leq j \leq j_2.$$

Wir sagen auch, A hat *C1P oder A ist eine* C1P-Matrix.

Falls eine Matrix C1P hat, kann die Permutation der Spalten, so dass die Einsen aufeinanderfolgend auftreten, mit dem Algorithmus von Meidanis und Telles [20] gefunden werden. Dieser Algorithmus hat eine Laufzeit von $\mathcal{O}(mn)$ für $A \in \mathbb{R}^{m \times n}$. Ohne Einschränkung der Allgemeinheit nehmen wir daher an, dass die Spalten einer Matrix mit C1P bereits geordnet sind, d. h. die Einsen jeder Zeile treten aufeinanderfolgend auf. Diese Eigenschaft wird auch *starke C1P* genannt. Wir arbeiten im Folgenden also, falls wir Matrizen mit C1P betrachten, mit der starken C1P.

Notation 1.3.2.

- *Wir sagen, dass ein Mengenüberdeckungsproblem C1P hat, falls seine Überdeckungsmatrix A C1P hat. Wir sprechen in diesem Fall auch von Mengenüberdeckungsproblemen* mit *C1P.*
- *Für eine Matrix $A \in \mathbb{B}^{m \times n}$ (die nicht notwendig C1P hat) sagen wir, dass die Zeile $\bar{k} \in \{1, ..., m\}$ C1P hat, falls die Einsen dieser Zeile aufeinanderfolgend auftreten, d. h. für alle $j_1, j_2 \in \{1, ..., n\}$ gilt:*

$$a_{\bar{k}j_1} = 1,\ a_{\bar{k}j_2} = 1 \text{ und } j_1 \leq j_2 \Rightarrow a_{\bar{k}j} = 1 \text{ für alle } j_1 \leq j \leq j_2.$$

Jede Matrix mit C1P hat die folgende Eigenschaft:

Lemma 1.3.3. *Eine Matrix A mit C1P ist total unimodular.*

Beweis. Ein Beweis dieses Lemmas kann in [21] gefunden werden. □

Bemerkung 1.3.4. *Die LP-Relaxation eines ganzzahligen Programms mit total unimodularer Koeffizientenmatrix A hat nur ganzzahlige Extrempunkte (siehe [21]). In einem Problem (SCP) mit C1P kann daher die Bedingung (1.3) durch $x \in \mathbb{R}^n$, $x \geq 0$ ersetzt werden und jedes Verfahren zur Lösung linearer Programme wird eine boolesche Lösung des Problems finden. Insbesondere kann solch ein Mengenüberdeckungsproblem in polynomialer Zeit gelöst werden.*

In [24] und [17] werden spezielle, besonders effiziente Algorithmen zur Lösung von Mengenüberdeckungsproblemen mit C1P vorgestellt. Der Algorithmus von Hochbaum [17] basiert auf der Idee, ein gegebenes (SCP) mit C1P in ein Longest Path Problem *in einem azyklischen Graphen zu transformieren. Das transformierte Problem kann durch dynamische Programmierung in linearer Zeit gelöst werden. Insgesamt ist die Laufzeit des in [17] vorgestellten Algorithmus zur Lösung*

eines Mengenüberdeckungsproblems mit C1P mit $m \times n$-Überdeckungsmatrix durch $\mathcal{O}(m+n)$ gegeben und damit nur linear abhängig von der Größe der Koeffizientenmatrix des Problems.

Diese effiziente Lösungsmöglichkeit für Mengenüberdeckungsprobleme mit C1P ist für uns ein Ansatzpunkt zur Erzeugung unterer und oberer Schranken für (SCP) in den folgenden Kapiteln. Die Überdeckungsmatrix A wird dabei so abgeändert, dass sie C1P hat und durch die Lösung des abgeänderten Problems untere bzw. obere Schranken entstehen.

Nehmen wir nun an, eine Matrix A ist gegeben, die nicht C1P hat. Es gibt also Zeilen in A, die nicht C1P haben. In diesen Zeilen steht somit mehr als nur ein Block von aufeinanderfolgenden Einsen. Um mit diesen Blöcken arbeiten zu können, definieren wir zunächst die Anzahl der Blöcke einer Zeile, sowie Blockanfang und Blockende eines bestimmten Blocks.

Definition 1.3.5. *(siehe [23]) Es sei $A \in \mathbb{B}^{m \times n}$, $M = \{1, ..., m\}, N = \{1, ..., n\}$.*

1. *Für die Zeile $k \in M$ von A sei bl_k die* Anzahl *der Blöcke von aufeinanderfolgenden Einsen.*

2. *Für den i-ten Block von Zeile $k \in M$ ($i \in \{1, ..., bl_k\}$) sei*

   - *$f_{k,i}$ die Spalte der ersten Eins von Block i.*
   - *$l_{k,i}$ die Spalte der letzten Eins von Block i.*

3. *Die* Länge *von Block i der Zeile $k \in M$ ($i \in \{1, ..., bl_k\}$) ist gegeben durch $l_{k,i} - f_{k,i} + 1$.*

Das bedeutet also für eine Matrix $A \in \mathbb{B}^{m \times n}$, dass

$$a_{kj} = \begin{cases} 1, & \text{falls es ein } i \in \{1, ..., bl_k\} \text{ gibt, so dass } f_{k,i} \leq j \leq l_{k,i} \\ 0, & \text{sonst.} \end{cases}$$

Notation 1.3.6. *Die Blöcke von aufeinanderfolgenden Einsen nennen wir im Folgenden auch einfach* Blöcke*.*

Die Schranken, die in den folgenden Kapiteln entwickelt werden, können zwar für beliebige Mengenüberdeckungsprobleme berechnet werden, eignen sich aber besonders gut für solche mit Überdeckungsmatrizen, die „fast“ C1P haben. D.h. die Anzahl der Blöcke der Zeilen der Matrix soll klein gegenüber der Größe der Matrix sein. Daher definieren wir:

Definition 1.3.7. *[23] Sei $A \in \mathbb{B}^{m \times n}$. A hat* fast C1P*, falls $\sum_{k=1}^{m} bl_k \ll mn$.*

Notation 1.3.8. *Anstatt von Matrizen mit fast C1P sprechen wir im Folgenden auch von Matrizen mit* Blockstruktur*. Dabei sollte sich der Leser Matrizen mit nur wenigen, aber langen Blöcken von aufeinanderfolgenden Einsen vorstellen.*

In vielen praktischen Anwendungen von Mengenüberdeckungsproblemen treten relativ dünn besetzte Überdeckungsmatrizen auf, die viele Zeilen mit C1P haben, falls die Spalten der Matrix auf eine bestimmte, oft durch die Anwendung selbst motivierte Art und Weise permutiert werden. Ein Beispiel ist das Problem, neue Bahnhöfe in einem gegebenen Schienennetz zu platzieren („stop location problem“), wie es in [24] und [23] vorgestellt wird.

In vielen weiteren Anwendungen treten Überdeckungsmatrizen mit fast C1P auf. Dies motiviert die Entwicklung von Verfahren zur Berechnung oberer und unterer Schranken für (SCP), die eine solche spezielle Struktur der Überdeckungsmatrix nutzen.

Kapitel 2

Untere Schranken für Mengenüberdeckungsprobleme

Untere Schranken für (SCP) werden im Allgemeinen durch die Betrachtung einer Relaxation des ganzzahligen Problems berechnet. Das heißt, (SCP) wird durch ein Optimierungsproblem mit größerem zulässigen Bereich und/oder veränderter Zielfunktion ersetzt, welches leichter zu lösen ist und dessen Lösung eine untere Schranke für (SCP) liefert. Die zwei Standardrelaxationen für Mengenüberdeckungsprobleme sind zum einen die LP-Relaxation, bei der die Forderung der Ganzzahligkeit der Lösung weggelassen wird. Da die exakte Lösung der LP-Relaxation mit Lösungsverfahren für lineare Programme typischerweise ebenfalls sehr zeitaufwändig ist (siehe [7]), stellt die Lagrange-Relaxation, wie sie z. B. in [4, 2, 9] vorgestellt wird, eine wichtige Alternative zur LP-Relaxation dar.

Wir wollen in diesem Kapitel Ideen von Ruf und Schöbel [23] zur Berechnung unterer Schranken von (SCP) aufgreifen und erweitern. Wir wollen dabei die Nebenbedingungen des Problems relaxieren, d. h. die Ganzzahligkeitsbedingung für die Variablen soll erhalten bleiben, die Überdeckungsmatrix A aber in der Art abgeändert werden, dass eine C1P-Matrix entsteht und das entsprechende Mengenüberdeckungsproblem effizient zu lösen ist. Basierend auf dieser Idee stellen wir zunächst in den Abschnitten 2.1.1 und 2.1.2 zwei Verfahren zur Berechnung unterer Schranken eines Mengenüberdeckungsproblems vor. In Teil 2.2 werden wir die

duale Schranke von (SCP) präsentieren. Schließlich vergleichen wir in Abschnitt 2.3 die Güte der vorgestellten unteren Schranken miteinander.

Die in den Abschnitten 2.1, 2.1.1 und 2.1.2 vorgestellte Theorie zur Berechnung unterer Schranken für (SCP) wurde ursprünglich für Matrizen, die fast C1P sind, entwickelt. Für diese Probleme werden im Allgemeinen auch bessere Ergebnisse als für solche mit beliebigen Überdeckungsmatrizen erzielt. Da die Theorie aber unabhängig von der Struktur der Matrix ist, werden wir im Folgenden beliebige Mengenüberdeckungsprobleme betrachten. Der Leser sollte aber im Hinterkopf behalten, dass eine Blockstruktur der betrachteten Matrizen sinnvoll für die Anwendung der Verfahren ist.

2.1 Relaxationen von (SCP) mit C1P-Matrizen

Wir wollen die Nebenbedingungen eines gegebenen (SCP) derart relaxieren, dass wir ein Mengenüberdeckungsproblem mit C1P zu lösen haben. Dazu werden wir die Zeilen der Überdeckungsmatrix A so abändern, dass jeweils genau ein Block von aufeinanderfolgenden Einsen pro Zeile vorkommt. Da es in A im Allgemeinen schon Zeilen mit genau einem Block geben kann, teilen wir die Matrix dahingehend auf, dass wir die Zeilen mit C1P und die Zeilen, die mehrere Blöcke haben, separat betrachten können. Da die Zeilen der Matrix permutiert werden können, ohne dass sich das Problem (SCP) verändert, schreiben wir nun für $A \in \mathbb{B}^{m \times n}$:

$$A = \begin{pmatrix} A^{bad} \\ A^{C1P} \end{pmatrix}. \tag{2.1}$$

Dabei sei $A^{bad} \in \mathbb{B}^{p \times n}$, $p \in \{1, ..., m-1\}$ und $A^{C1P} \in \mathbb{B}^{(m-p) \times n}$ und es gelte:

$$\begin{aligned} bl_k &> 1 \qquad \forall k \in \{1, ..., p\}, \\ bl_k &= 1 \qquad \forall k \in \{p+1, ..., m\}. \end{aligned}$$

Falls $bl_k = 1 \, \forall k \in \{1, ..., m\}$, so schreiben wir $A = A^{C1P}$ und setzen $p = 0$. Falls $bl_k > 1 \; \forall k \in \{1, ..., m\}$ schreiben wir $A = A^{bad}$ und setzen $p = m$.

Das Problem (SCP) kann mit dieser Aufteilung von A äquivalent geschrieben werden als:

$$\begin{array}{lll} & \text{minimiere} & z = c^t x \\ \textbf{(SCP')} & \text{so dass} & A^{bad} x \geq \underline{1}_p \qquad (2.2) \\ & & A^{C1P} x \geq \underline{1}_{(m-p)} \qquad (2.3) \\ & & x \in \mathbb{B}^n. \end{array}$$

Nach Konstruktion von (SCP') gilt das folgende Lemma:

Lemma 2.1.1. *[23] (SCP) und (SCP') sind äquivalent.*

Die Nebenbedingung $Ax \geq \underline{1}_m$ von (SCP) ist nun aufgeteilt in die „leichter" zu erfüllende Nebenbedingung (2.3) und die Nebenbedingung (2.2), welche im Allgemeinen zu Schwierigkeiten bei der Lösung des Problems führt.

2.1.1 Das Problem (SCPl)

Betrachten wir das Problem (SCP') und gehen davon aus, dass die Matrix A^{C1P} existiert (also $p < m$), so erhalten wir eine Relaxation von (SCP'), indem wir die Nebenbedingung (2.2) weglassen. D.h., wir vernachlässigen alle Zeilen der Matrix A, die seine C1P-Eigenschaft zerstören. Falls die Matrix A^{C1P} existiert, erhalten wir das folgende Mengenüberdeckungsproblem mit C1P.

$$\begin{array}{lll} & \text{minimiere} & c^t x \\ \textbf{(SCPl)} & \text{so dass} & A^{C1P} x \geq \underline{1}_{m-p} \\ & & x \in \mathbb{B}^n. \end{array}$$

Lemma 2.1.2. *(siehe [23]) Jede Optimallösung von (SCPl) liefert eine untere Schranke für (SCP').*

Beweis. Da die Zielfunktionen der beiden Programme identisch sind und (SCPl) nur einen Teil der Nebenbedingungen von (SCP') enthält, ist (SCPl) eine Relaxa-

tion von (SCP') und damit ist nach [21] der Zielfunktionswert jeder Optimallösung von (SCPl) eine untere Schranke für (SCP'). □

Bemerkung 2.1.3. *Da die Überdeckungsmatrix A^{C1P} C1P hat, können wir das Problem (SCPl) nach Bemerkung 1.3.4 effizient lösen und somit eine untere Schranke von (SCP') bzw. (SCP) bestimmen.*

Falls die Matrix A nur wenige Zeilen mit C1P hat, ist der zulässige Bereich von (SCPl) viel größer als der zulässige Bereich von (SCP) und der Wert der unteren Schranke kann stark vom optimalen Zielfunktionswert von (SCP) abweichen. Falls es in A überhaupt keine C1P-Zeile gibt, hat das Problem (SCPl) keine Nebenbedingungen und führt zu der trivialen unteren Schranke von 0*. In diesem Fall könnten als Ausweg die Spalten von A derart permutiert werden, dass zumindest eine Zeile mit C1P auftritt und somit A^{C1P} aufgestellt werden kann. Die Lösung von (SCPl) liefert dann eine nicht triviale untere Schranke für (SCP).*

Den Zusammenhang des Wertes der von (SCPl) erzeugten unteren Schranke von (SCP) und dem Anteil der Zeilen mit C1P in der Überdeckungsmatrix des Problems werden wir in Kapitel 4 numerisch untersuchen.

2.1.2 Das Problem (S$\mathring{\text{C}}$Pl)

Um eine bessere untere Schranke als durch Lösen von (SCPl) zu erhalten, wollen wir eine Relaxation von (SCP) aufstellen, deren zulässiger Bereich in dem zulässigen Bereich von (SCPl) enthalten ist. Dazu behalten wir die Bedingung $A^{C1P} \geq \underline{1}_{(m-p)}$ bei, fügen aber weitere Nebenbedingungen hinzu. Um diese zu bestimmen, relaxieren wir die Bedingung (2.2), indem wir die Zeilen von A^{bad} „konvexifizieren“, wie wir es im Folgenden beschreiben.

Definition 2.1.4. *Es sei $A \in \mathbb{B}^{m\times n}$. Weiterhin sei bl_k die Anzahl der Blöcke von Zeile k für alle $k \in \{1, ..., m\}$ und $f_{k,i}$ bzw. $l_{k,i}$ die Spalte der ersten bzw. letzten Eins des i-ten Blocks der Zeile k von A. Dann definieren wir die* konvexifizierte *Matrix $\mathring{A} \in \mathbb{B}^{m\times n}$ durch:*

$$\mathring{a}_{kj} = \begin{cases} 1, & \textit{falls } f_{k,1} \leq j \leq l_{k,bl_k} \\ 0, & \textit{sonst.} \end{cases}$$

Hierbei ist $\mathring{a}_{kj}$ der Eintrag von $\mathring{A}$ in der k-ten Zeile und j-ten Spalte.

In der konvexifizierten Matrix $\mathring{A}$ werden also alle Nullen, die zwischen dem ersten und letzten Block einer Zeile von A liegen, mit Einsen aufgefüllt. Jede Zeile von $\mathring{A}$ besitzt genau einen Block von aufeinanderfolgenden Einsen.

Beispiel 2.1.5. *Sei*

$$A = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix}.$$

Dann ist die konvexifizierte Matrix $\mathring{A}$ gegeben durch:

$$\mathring{A} = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix}.$$

Bemerkung 2.1.6. *Schreiben wir*

$$A = \begin{pmatrix} A^{bad} \\ A^{C1P} \end{pmatrix}$$

mit $A^{bad} \in \mathbb{B}^{p \times n}$, $A^{C1P} \in \mathbb{B}^{(m-p) \times n}$, so treten die Zeilen von A^{C1P} in der konvexifizierten Matrix $\mathring{A}$ unverändert auf, da sie jeweils genau einen Block besitzen.

Um nun eine untere Schranke für (SCP) zu erhalten, lösen wir das Mengenüberdeckungsproblem, das als Überdeckungsmatrix die konvexifizierte Matrix $\mathring{A}$ hat:

$$\begin{array}{lll} & \text{minimiere} & c^t x \\ (\mathbf{S\mathring{C}Pl}) & \text{so dass} & \mathring{A}x \geq \underline{1}_m \\ & & x \in \mathbb{B}^n. \end{array}$$

Lemma 2.1.7. *Jede Optimallösung von ($S\mathring{C}Pl$) liefert eine untere Schranke für (SCP'). Diese untere Schranke ist nicht schlechter als die untere Schranke, die durch eine Optimallösung von (SCPl) erhalten wird.*

Beweis. Für die zulässigen Bereiche von (SCP'), (S$\mathring{\text{C}}$Pl) und (SCPl) gilt:

$$\begin{aligned}\left\{x \in \mathbb{B}^n : \begin{pmatrix} A^{bad} \\ A^{C1P} \end{pmatrix} x \geq \underline{1}_m\right\} &\subseteq \{x \in \mathbb{B}^n : \mathring{A}x \geq \underline{1}_m\} \\ &\subseteq \{x \in \mathbb{B}^n : A^{C1P}x \geq \underline{1}_{(m-p)}\}.\end{aligned}$$

Da die Zielfunktionen der Probleme identisch sind und die zulässigen Bereiche einander enthalten, gilt nach [21], dass (S$\mathring{\text{C}}$Pl) eine Relaxation von (SCP') und (SCPl) eine Relaxation von (S$\mathring{\text{C}}$Pl) ist. Daraus folgt nach [21], dass die Optimallösung von (S$\mathring{\text{C}}$Pl) eine untere Schranke für (SCP') und die Optimallösung von (SCPl) eine untere Schranke für (S$\mathring{\text{C}}$Pl) liefert. Damit ist die untere Schranke, die durch (S$\mathring{\text{C}}$Pl) erzeugt wird, mindestens so gut wie die durch (SCPl) erzeugte. □

Bemerkung 2.1.8. *Da die Koeffizientenmatrix von (S$\mathring{\text{C}}$Pl) C1P hat, können wir das Problem nach Bemerkung 1.3.4 effizient lösen. Da die durch (S$\mathring{\text{C}}$Pl) erzeugte untere Schranke mindestens so gut ist wie eine untere Schranke, die durch Lösen von (SCPl) gefunden werden kann, sollten wir (S$\mathring{\text{C}}$Pl) bei der Berechnung unterer Schranken von (SCP) bevorzugen.*

Wir bemerken zusätzlich, dass die Überdeckungsmatrix A^{C1P} von (SCPl) $p \leq m$ Zeilen hat. Da die Laufzeit zur Lösung eines Mengenüberdeckungsproblems mit C1P abhängig von der Anzahl der Zeilen der Überdeckungsmatrix ist, kann (SCPl) schneller gelöst werden als (S$\mathring{\text{C}}$Pl).

2.1.3 Güte der von (SCPl) und (S$\mathring{\text{C}}$Pl) erzeugten unteren Schranken

Zunächst bemerken wir, wie wir die Güte eines Verfahrens zur Berechnung einer oberen bzw. unteren Schranke bewerten wollen.

Notation 2.1.9. *Für die Güte eines Verfahrens zur Berechnung einer unteren bzw. oberen Schranke z eines (SCP) mit optimalem Zielfunktionswert $z^\star$ sind drei verschiedene Kriterien von Bedeutung:*

1. *Der* absolute Fehler $|z - z^\star|$ *des Verfahrens.*

2. *Der* Approximationsfaktor $\frac{z}{z^\star}$ *des Verfahrens.*

3. *Der* relative Fehler $\varepsilon_r := \frac{|z - z^\star|}{z^\star}$ *des Verfahrens.*

Dabei kann der relative Fehler direkt aus dem Approximationsfaktor berechnet werden und umgekehrt. Wir werden bei den Güteabschätzungen für die vorgestellten Verfahren Abschätzungen für den Approximationsfaktor beweisen. Den relativen Fehler des jeweiligen Verfahrens geben wir zusätzlich in einer Bemerkung an.

Je kleiner der absolute und der relative Fehler eines Verfahrens zur Berechnung einer Schranke von (SCP) sind (bzw. je näher der Approximationsfaktor bei 1 liegt), umso besser ist die Güte des Verfahrens.

Das folgende Lemma gibt Aufschluss über den absoluten Fehler und den Approximationsfaktor der von (SCPl) erzeugten unteren Schranke von (SCP).

Lemma 2.1.10. *Sei* $A = \begin{pmatrix} A^{bad} \\ A^{C1P} \end{pmatrix} \in \mathbb{B}^{m \times n}$ *mit* $A^{bad} \in \mathbb{B}^{p \times n}$ $(1 \leq p \leq m)$ *und* $c \in \mathbb{R}^n_+$.

Es sei $c_p :=$ *Summe der* p *größten Kostenwerte und* $c_{min} := \min_{j \in \{1,\ldots,n\}}\{c_j\}$ *der kleinste Kostenwert. Sei* $x^\star$ *eine Optimallösung von (SCP) mit Zielfunktionswert* $z^\star = c^t x^\star$ *und* x_l *eine Optimallösung von (SCPl) mit Zielfunktionswert* $z_l = c^t x_l$. *Dann gilt:*

1. $$|z^\star - z_l| \leq c_p. \tag{2.4}$$

2. *Für* $p < m$ *ist*
$$\frac{z_l}{z^\star} \geq \frac{c_{min}}{c_{min} + c_p}. \tag{2.5}$$

Beweis. 1. Sei x_l eine Optimallösung von (SCPl). Falls $p < m$ ist, existiert A^{C1P} und als zulässige Lösung von (SCPl) überdeckt x_l alle Zeilen dieser Matrix. Falls $p = m$ ist, so hat (SCPl) keine Nebenbedingung und x_l ist der Nullvektor. In beiden Fällen werden die p Zeilen von A^{bad} im Allgemeinen nicht von x_l überdeckt. Wählen wir für jede dieser p Zeilen eine Spalte, die diese Zeile überdeckt, so haben wir zusammen mit x_l eine Spaltenauswahl getroffen, die alle Zeilen von A überdeckt. Die zusätzlich gewählten Spalten

haben Kosten von maximal c_p. Da die so konstruierte Spaltenauswahl zulässig für (SCP) ist, gilt:

$$z_l + c_p \geq z^\star, \tag{2.6}$$

also

$$|z^\star - z_l| = z^\star - z_l \leq c_p.$$

2. Mit Abschätzung (2.6) für den Zielfunktionswert z_l einer Lösung x_l von (SCPl) und der Beobachtung, dass $z_l \geq c_{min}$ ist (da $p < m$ ist, existiert A^{C1P} und x_l kann nicht der Nullvektor sein), folgt:

$$\begin{aligned} \frac{z^\star}{z_l} &\leq \frac{z_l + c_p}{z_l} = 1 + \frac{c_p}{z_l} \\ &\leq 1 + \frac{c_p}{c_{min}} = \frac{c_{min} + c_p}{c_{min}}. \end{aligned}$$

Da beide Seiten der Ungleichung positiv sind, gilt für den Kehrwert

$$\frac{z_l}{z^\star} \geq \frac{c_{min}}{c_{min} + c_p}$$

und die Abschätzung ist bewiesen.

□

Bemerkung 2.1.11. *Für den relativen Fehler ε_r der Lösung z_l von (SCPl) gilt nach Lemma 2.1.10:*

$$\varepsilon_r = \frac{|z^\star - z_l|}{z^\star} = \frac{z^\star - z_l}{z^\star} = 1 - \frac{z_l}{z^\star} \leq 1 - \frac{c_{min}}{c_{min} + c_p} = \frac{c_p}{c_{min} + c_p}.$$

Hierbei ist p die Anzahl der Zeilen von A^{bad} mit $1 \leq p < m$.

Bemerkung 2.1.12. *Falls ein ungewichtetes Mengenüberdeckungsproblem vorliegt, so gilt nach Lemma 2.1.10 mit $c = \underline{1}_n$:*

1. $|c^t x^\star - c^t x_l| \leq p$.
2. $\frac{c^t x_l}{c^t x^\star} \geq \frac{1}{1+p}$.

3. $\varepsilon_r = \dfrac{|c^t x^\star - c^t x_l|}{c^t x^\star} \leq \dfrac{p}{1+p}$.

Hierbei ist p die Anzahl der Zeilen von A^{bad} mit $1 \leq p < m$. Die erste Aussage gilt auch für $p = m$.

Da die Optimallösung von (S$\mathring{\text{C}}$Pl) immer einen mindestens so großen Zielfunktionswert wie eine Lösung von (SCPl) hat, übertragen sich die Güteabschätzungen aus Lemma 2.1.10 auf die von (S$\mathring{\text{C}}$Pl) erzeugte untere Schranke von (SCP).

Korollar 2.1.13. *Sei* $A = \begin{pmatrix} A^{bad} \\ A^{C1P} \end{pmatrix} \in \mathbb{B}^{m \times n}$ *mit* $A^{bad} \in \mathbb{B}^{p \times n}$ *($1 \leq p \leq m$) und* $c \in \mathbb{R}^n_+$.

Es sei $c_p :=$ Summe der p größten Kostenwerte und $c_{min} := \min_{j \in \{1,\ldots,n\}}\{c_j\}$ der kleinste Kostenwert. Sei $x^\star$ eine Optimallösung von (SCP) mit Zielfunktionswert $z^\star = c^t x^\star$ und $\mathring{x}$ eine Optimallösung von (S$\mathring{\text{C}}$Pl) mit Zielfunktionswert $\mathring{z} = c^t \mathring{x}$. Dann gilt:

1.
$$|z^\star - \mathring{z}| \leq c_p. \tag{2.7}$$

2.
$$\frac{\mathring{z}}{z^\star} \geq \frac{c_{min}}{c_{min} + c_p}. \tag{2.8}$$

Beweis. Für die optimalen Zielfunktionswerte z_l von (SCPl) und $\mathring{z}$ von (S$\mathring{\text{C}}$Pl) gilt nach Lemma 2.1.7:

$$z_l \leq \mathring{z}.$$

Daher folgt (2.7) für $1 \leq p \leq m$ und (2.8) für $1 \leq p < m$ direkt aus Lemma 2.1.10.

Da die Matrix $\mathring{A}$ für $p = m$ existiert, gilt auch in diesem Fall $\mathring{z} \geq c_{min}$ und wie im Beweis von Lemma 2.1.10 gilt

$$\frac{z^\star}{\mathring{z}} \leq \frac{c_{min} + c_p}{c_{min}}.$$

Der Kehrwert liefert (2.8) für $p = m$. □

Bemerkung 2.1.14. *Für den relativen Fehler ε_r der Lösung $\mathring{z}$ von (SCPl) gilt nach Korollar 2.1.13:*

$$\varepsilon_r = \frac{|z^\star - \mathring{z}|}{z^\star} = \frac{z^\star - \mathring{z}}{z^\star} = 1 - \frac{\mathring{z}}{z^\star} \leq 1 - \frac{c_{min}}{c_{min} + c_p} = \frac{c_p}{c_{min} + c_p}.$$

Hierbei ist p die Anzahl der Zeilen von A^{bad} mit $1 \leq p \leq m$.

Bemerkung 2.1.15. *Falls ein ungewichtetes Mengenüberdeckungsproblem vorliegt, so gilt nach Korollar 2.1.13 mit $c = \underline{1}_n$:*

1. $|c^t x^\star - c^t \mathring{x}| \leq p$.
2. $\frac{c^t \mathring{x}}{c^t x^\star} \geq \frac{1}{1+p}$.
3. $\varepsilon_r = \frac{|c^t x^\star - c^t \mathring{x}|}{c^t x^\star} \leq \frac{p}{1+p}$.

Hierbei ist p die Anzahl der Zeilen von A^{bad} mit $1 \leq p \leq m$.

Bei Problemen mit großer Matrix A^{bad} oder hohen maximalen Kosten geben die in Lemma 2.1.10 und Korollar 2.1.13 berechneten Güteabschätzungen keine hilfreichen Beschränkungen für die aus (SCPl) und (S$\mathring{\text{C}}$Pl) berechneten unteren Schranken an, da dann der Faktor c_p aus den Abschätzungen (2.4) und (2.5) bzw. (2.7) und (2.8) sehr groß ist. In der Praxis zeigt sich aber, dass die Ideen, die Matrix A^{C1P} oder das konvexifizierte A für die Berechnung einer unteren Schranke für (SCP) zu nutzen, durchaus sinnvoll sein können und die Abweichungen von der optimalen Lösung bei weitem nicht so groß sind, wie sie durch Lemma 2.1.10 und Korollar 2.1.13 angegeben werden. Allerdings zeigt das folgende Beispiel, dass die Abschätzungen scharf sind.

Beispiel 2.1.16. *Sei*

$$A = \begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix},$$

$$A^{C1P} = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}$$

und

$$\mathring{A} = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}.$$

Die Anzahl der Zeilen von A^{bad} ist $p = 1$. Mit den Kosten $c = (1, \frac{1}{2}, 1)^t$ gilt für die Summe der p größten Kostenwerte $c_p = 1$.

Optimale Lösungen von (SCP), (SCPl) und ($S\mathring{C}Pl$) sind durch $x^\star = (1, 1, 0)$, $x_l = (0, 1, 0)$ und $\mathring{x} = (0, 1, 0)$ mit Zielfunktionswerten $z^\star = \frac{3}{2}$, $z_l = \frac{1}{2}$ und $\mathring{z} = \frac{1}{2}$ gegeben. Demnach gilt $|z^\star - z_l| = |z^\star - \mathring{z}| = 1 = c_p$ und $\frac{z_l}{z^\star} = \frac{\mathring{z}}{z^\star} = \frac{1}{3} = \frac{c_{min}}{c_{min}+c_p}$.

Die beiden unteren Schranken z_l und $\mathring{z}$ sind also identisch. Die in Lemma 2.1.10 und Korollar 2.1.13 angegebenen Abschätzungen werden in diesem Beispiel erreicht.

2.2 Duale Schranke

In diesem Abschnitt wird eine weitere Möglichkeit vorgestellt, eine untere Schranke für ein Mengenüberdeckungsproblem effizient zu berechnen. Wir werden die Dualitätstheorie aus der linearen Optimierung anwenden, um duale Schranken für (SCP) zu berechnen. Wir betrachten hierzu das Duale der LP-Relaxation von (SCP), gegeben durch

$$\begin{array}{lll} & \text{maximiere} & \underline{1}_m^t y \\ (\textbf{Dual-SCP}) & \text{so dass} & A^T y \leq c \\ & & y \geq 0. \end{array}$$

Das folgende Lemma stellt sicher, dass die Lösung von (Dual-SCP) tatsächlich eine untere Schranke von (SCP) liefert.

Lemma 2.2.1. *[21] Es sei $y^\star$ eine Optimallösung von (Dual-SCP). Dann ist*

$$f^l := \underline{1}_m^t y^\star$$

eine untere Schranke für (SCP).

Beweis. Nach dem schwachen Dualitätssatz für lineare Programme (siehe [16]) liefert der Ausdruck $\underline{1}_m^t y^\star$ eine untere Schranke für die LP-Relaxation von (SCP) und damit nach [21] auch für das Problem (SCP) selbst. □

Notation 2.2.2. *Wir nennen die untere Schranke f^l, die aus der Lösung von (Dual-SCP) entsteht, auch die* duale Schranke *von (SCP).*

Da es sich bei (Dual-SCP) um ein lineares Programm handelt, kann die duale Schranke für (SCP) mit bekannten Verfahren zur Lösung linearer Programme berechnet werden. Insbesondere ist dies in polynomieller Zeit möglich. Allerdings kann die Berechnung einer Optimallösung von (Dual-SCP) für große Probleme sehr aufwändig sein, weshalb in der Praxis auch für (Dual-SCP) Näherungslösungen gesucht werden. Dies wird z. B. in [2] durch duale Heuristiken und in [23] speziell für Mengenüberdeckungsprobleme mit fast C1P vorgestellt.

Die Lösung von (Dual-SCP) liefert eine Schranke für den Zielfunktionswert von (SCP), gibt aber keine Hinweise darauf, welche Spalten von A in einer Optimallösung von (SCP) gewählt werden sollten. Diese Erkenntnis macht sie unbrauchbar für den in Abschnitt 3.5 vorgestellten Algorithmus für die Berechnung einer oberen Schranke für (SCP), für welchen die Lösungen von (SCPl) und (S$\mathring{\text{C}}$Pl) verwendet werden können.

In Abschnitt 2.3 werden wir zeigen, dass die duale Schranke nie schlechter ist als die unteren Schranken, die wir durch Lösen von (SCPl) und (S$\mathring{\text{C}}$Pl) erhalten. Daraus folgt zum einen, dass die Abschätzungen aus Lemma 2.1.13 auch für die duale Schranke gelten. Andererseits belegen Tests mit zufälligen Matrizen (Kapitel 4), dass die duale Schranke sogar erheblich bessere Ergebnisse als (S$\mathring{\text{C}}$Pl) liefert.

Wir werden nun für das (SCP) aus Beispiel 2.1.16 die duale Schranke berechnen.

Beispiel 2.2.3. *Sei ein Mengenüberdeckungsproblem (SCP) mit Überdeckungsmatrix*

$$A = \begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}$$

und Kosten $c = (1, \frac{1}{2}, 1)^t$ gegeben. Dann ist (Dual-SCP) gegeben durch

$$
\begin{array}{lll}
\textit{maximiere} & y_1 + y_2 + y_3 & \\
\textit{s.d.} & & \\
& y_1 + y_2 & \leq 1 \\
& y_2 + y_3 & \leq \frac{1}{2} \\
& y_1 + y_3 & \leq 1 \\
& y_i \geq 0 \ , \ i = 1,2,3 &
\end{array}
$$

mit optimaler Lösung $y^\star = (1,0,0)$*. Damit ist* $f^l = 1$ *die duale Schranke für (SCP). Insbesondere gilt für den optimalen Zielfunktionswert* $\mathring{z} = \frac{1}{2}$ *von (SCPl):* $\mathring{z} < f_l$*.*

2.3 Hierarchie für die Güte der unteren Schranken

Die Beispiele 2.1.16 und 2.2.3 zeigen, dass die duale Schranke echt besser sein kann als die untere Schranke $\mathring{z}$, die aus (SCPl) berechnet wird. Im folgenden Lemma beweisen wir, dass die duale Schranke in jedem Fall mindestens so gut ist wie $\mathring{z}$.

Lemma 2.3.1. *Gegeben sei ein (SCP) mit Überdeckungsmatrix* $A \in \mathbb{B}^{m \times n}$ *und Kosten* $c \in \mathbb{R}^n_+$*. Weiterhin sei* $\mathring{z}$ *der Zielfunktionswert einer Optimallösung von (SCPl) und* f^l *die duale Schranke von (SCP). Dann gilt:*

$$\mathring{z} \leq f^l.$$

Beweis. Wir betrachten das zur LP-Relaxation von (SCPl) duale Programm

$$\text{(Dual-SCPl)} \qquad \max\{\underline{1}_m^t y : \mathring{A}^t y \leq c, y \geq 0\}$$

und das Programm

$$\text{(Dual-SCP)} \qquad \max\{\underline{1}_m^t y : A^t y \leq c, y \geq 0\}.$$

Für die zulässigen Bereiche der beiden Programme gilt:

$$\{y \in \mathbb{R}^m : \mathring{A}^t y \leq c, y \geq 0\} \subseteq \{y \in \mathbb{R}^m : A^t y \leq c, y \geq 0\}.$$

Denn sei y zulässig für (Dual-S$\mathring{\text{C}}$Pl). Für die k-te Zeile $(A^t)_k$ $(k \in \{1, ..., n\})$ von A^t gilt dann nach Konstruktion von $\mathring{A}$:

$$(A^t)_k y = \sum_{j=1}^{m} a_{jk} y_k \leq \sum_{j=1}^{m} \mathring{a}_{jk} y_k \leq c_k.$$

Hierbei sind a_{jk} bzw. $\mathring{a}_{jk}$ die Einträge von A bzw. $\mathring{A}$ in Zeile j und Spalte k.

Da die Zielfunktionen von (Dual-S$\mathring{\text{C}}$Pl) und (Dual-SCP) identisch sind und die zulässigen Bereiche einander enthalten, ist (Dual-SCP) eine Relaxation von (Dual-S$\mathring{\text{C}}$Pl) und nach [21] gilt für Optimallösungen $\mathring{y}$ von (Dual-S$\mathring{\text{C}}$Pl) und $y^\star$ von (Dual-SCP):

$$\underline{1}_m^t \mathring{y} \leq \underline{1}_m^t y^\star.$$

Da wir voraussetzen, dass A keine Nullzeile enthält, gibt es insbesondere auch in $\mathring{A}$ keine Nullzeile und es existiert eine endliche Optimallösung von (Dual-S$\mathring{\text{C}}$Pl). Außerdem sind die optimalen Zielfunktionswerte von (S$\mathring{\text{C}}$Pl) und seiner LP-Relaxation gleich, da (S$\mathring{\text{C}}$Pl) ein C1P-Mengenüberdeckungsproblem ist (siehe Bemerkung 1.3.4).

Nach dem starken Dualitätssatz für linearen Programme (siehe [16]) gilt somit für eine Optimallösung $\mathring{x}$ von (S$\mathring{\text{C}}$Pl):

$$c^t \mathring{x} = \underline{1}_m^t \mathring{y}.$$

Insgesamt folgt

$$\mathring{z} = c^t \mathring{x} = \underline{1}_m^t \mathring{y} \leq \underline{1}_m^t y^\star = f^l$$

und die Abschätzung ist bewiesen. □

Bemerkung 2.3.2. *Für den optimalen Zielfunktionswert $z^\star$, die duale Schranke f^l, die durch (S$\mathring{C}$Pl) berechnete untere Schranke $\mathring{z}$ und die von (SCPl) erzeugte Schranke z_l ergibt sich nach den Lemmata 2.1.7 und 2.3.1 die folgende Güte-Hierarchie:*

$$z_l \leq \mathring{z} \leq f^l \leq z^\star.$$

Die Berechnung einer unteren Schranke durch die Lösung von (Dual-SCP) liefert also das beste Ergebnis. Allerdings ist diese Alternative im Allgemeinen aufwändiger als die Berechnung von Lösungen der beiden anderen Probleme, da diese nach Bemerkung 1.3.4 in linearer Zeit lösbar sind. Außerdem können die Lösungen von (SCPl) und (S$\mathring{C}$Pl) für die Berechnung einer oberen Schranke (siehe Abschnitt 3.5) weiterverwendet werden, dies ist für (Dual-SCP) nicht möglich.

Kapitel 3

Obere Schranken für Mengenüberdeckungsprobleme

Obere Schranken für (SCP) werden im Allgemeinen dadurch gewonnen, dass eine zulässige Lösung des Problems berechnet wird, dessen Zielfunktionswert dann automatisch eine obere Schranke für den optimalen Zielfunktionswert von (SCP) ist. So können obere Schranken für Mengenüberdeckungsprobleme z. B. durch die Greedy-Heuristik [10], Varianten der Greedy-Heuristik [2, 13, 15], Innere Punkte Methoden [19], Genetische Algorithmen [5] oder Verschärfung der Nebenbedingungen [23] berechnet werden.

In diesem Kapitel wollen wir Ansätze von Ruf und Schöbel [23] aufgreifen und erweitern. Die grundlegende Idee eines ersten Ansatzes besteht darin, für jede Zeile der Überdeckungsmatrix A von (SCP) nur einen bestimmten Block aufeinanderfolgender Einsen zu wählen, und die Einsen der anderen Blöcke der Zeile durch Nullen zu ersetzen. Die Matrix $\widetilde{A}$, die durch dieses Vorgehen entsteht, hat C1P und das Mengenüberdeckungsproblem mit Überdeckungsmatrix $\widetilde{A}$ kann nach Bemerkung 1.3.4 effizient gelöst werden. Diese Lösung ist dann zulässig für (SCP) und erzeugt daher eine obere Schranke des Problems.

In Abschnitt 3.1 werden wir zunächst die allgemeine Konstruktion der Matrix $\widetilde{A}$ vornehmen und beweisen, dass durch die Lösung des entsprechenden Mengenüberdeckungsproblems tatsächlich eine obere Schranke von (SCP) berechnet wird. In den darauf folgenden Abschnitten werden wir der Frage nachgehen, wie

die Blockauswahl zu treffen ist, um möglichst gute obere Schranken zu erzeugen. Insbesondere stellen wir in Abschnitt 3.2 einen Algorithmus zur Berechnung oberer Schranken für ungewichtete Mengenüberdeckungsprobleme vor und geben eine Güteabschätzung für diesen Algorithmus an. Der in Abschnitt 3.3 vorgestellte Algorithmus für beliebige (SCP) nutzt die bekannte Greedy-Heuristik zur Auswahl der Blöcke von $\widetilde{A}$. Für die so erzeugte obere Schranke geben wir in Abschnitt 3.3.3 eine Gütegarantie an.

In Abschnitt 3.4 werden wir die vorangegangenen Ideen zur Blockauswahl verallgemeinern, indem wir eine Vorschrift zur Auswahl der Blöcke für eine beliebige zulässige Lösung x von (SCP) angeben. Der darauf aufbauende Algorithmus berechnet eine zulässige Lösung $\widetilde{x}$ von (SCP), dessen Zielfunktionswert höchstens so groß wie der Zielfunktionswert von x ist.

In Abschnitt 3.5 greifen wir eine zweite Idee von Ruf und Schöbel [23] auf. Hierbei werden wir zunächst eine Lösung $\mathring{x}$ von einem der Probleme (SCPl) oder (S$\mathring{\text{C}}$Pl) aus Kapitel 2 berechnen. Aus dieser Lösung konstruieren wir ein neues Mengenüberdeckungsproblem, welches nur einen Teil der Nebenbedingungen von (SCP) besitzt. Für dieses Problem berechnen wir mit dem in Abschnitt 3.3 vorgestellten Algorithmus eine obere Schranke $\widetilde{x}$. Eine Kombination von $\mathring{x}$ und $\widetilde{x}$ liefert dann eine zulässige Lösung und damit eine obere Schranke des Problems (SCP).

3.1 Blockauswahl und das Problem (SCPu(ɬ))

Gegeben sei die Überdeckungsmatrix $A \in \mathbb{B}^{m\times n}$ eines Mengenüberdeckungsproblems. Um eine Matrix $\widetilde{A}$ zu konstruieren, die genau einen der Blöcke aufeinanderfolgender Einsen pro Zeile von A besitzt, benötigen wir zunächst eine Vorschrift dafür, welcher Block für eine bestimmte Zeile gewählt werden soll. Dazu definieren wir eine Abbildung, die genau dies erledigt.

Definition 3.1.1. *[23] Es sei $A \in \mathbb{B}^{m\times n}$. Es sei $ɬ : \{1, ..., m\} \to \mathbb{N}^m$ eine Abbildung, die einen Block $i = ɬ(k)$ für jede Zeile $k \in \{1, ..., m\}$ auswählt. Die Abbildung $ɬ$ heißt zulässig für A, falls $1 \leq ɬ(k) \leq bl_k$ für alle $k = 1, ..., m$.*
Eine für A zulässige Abbildung $ɬ$ nennen wir Blockabbildung *von A.*

Nun definieren wir die Matrix $\widetilde{A}^{\mathfrak{l}} \in \mathbb{B}^{m \times n}$, welche in jeder Zeile genau den durch die zulässige Abbildung $\mathfrak{l}$ bestimmten Block von A besitzt.

Definition 3.1.2. *Es sei $A \in \mathbb{B}^{m \times n}$ und es sei $\mathfrak{l}$ eine Blockabbildung von A. Dann definieren wir die zu $\mathfrak{l}$ korrespondierende* Blockmatrix $\widetilde{A}^{\mathfrak{l}}$ *durch:*

$$\widetilde{a}^{\mathfrak{l}}_{kj} = \begin{cases} 1 & \text{falls } f_{k,\mathfrak{l}(k)} \leq j \leq l_{k,\mathfrak{l}(k)} \\ 0 & \text{sonst.} \end{cases}$$

Hierbei ist $\widetilde{a}^{\mathfrak{l}}_{kj}$ der Eintrag von $\widetilde{A}^{\mathfrak{l}}$ in der k-ten Zeile und j-ten Spalte.

Wenn aus dem Zusammenhang hervorgeht, um welche Blockabbildung es sich handelt, schreiben wir auch nur $\widetilde{A}$ statt $\widetilde{A}^{\mathfrak{l}}$.

Es sei nun eine zulässige Blockabbildung $\mathfrak{l}$ von A gegeben. Wir betrachten das folgende Mengenüberdeckungsproblem mit Überdeckungsmatrix $\widetilde{A}^{\mathfrak{l}}$:

$$\begin{aligned} & & \text{minimiere} \quad & c^t x & (3.1) \\ & \textbf{(SCPu(ɬ))} & \text{so dass} \quad & \widetilde{A}^{\mathfrak{l}} x \geq \underline{1}_m & (3.2) \\ & & & x \in \mathbb{B}^n & (3.3) \end{aligned}$$

Da die Matrix $\widetilde{A}^{\mathfrak{l}}$ nach Konstruktion genau einen Block von aufeinanderfolgenden Einsen pro Zeile hat, handelt es sich um eine C1P-Matrix. Das Mengenüberdeckungsproblem (SCPu(ɬ)) kann also nach Bemerkung 1.3.4 in $\mathcal{O}(m+n)$ gelöst werden. Da das folgende Lemma zeigt, dass Lösungen von (SCPu(ɬ)) obere Schranken von (SCP) sind, haben wir also, falls eine Blockabbildung bekannt ist, eine sehr effiziente Methode zur Berechnung von oberen Schranken für Mengenüberdeckungsprobleme gefunden.

Lemma 3.1.3. *[23] Es sei $x^\star$ eine Optimallösung von (SCP) mit Überdeckungsmatrix $A \in \mathbb{B}^{m \times n}$ und Kosten $c \in \mathbb{R}^n_+$. Sei weiterhin $\mathfrak{l}$ eine Blockabbildung von A. Dann ist jede zulässige Lösung x von (SCPu(ɬ)) zulässig für (SCP) und es gilt: $c^t x \geq c^t x^\star$.*

Beweis. Es sei $x = (x_1, ..., x_n) \in \mathbb{B}^n$ eine zulässige Lösung von (SCPu($\mathfrak{l}$)). Betrachte die Zeile $k \in \{1, ..., m\}$. Nach Konstruktion von $\widetilde{A}^{\mathfrak{l}}$ gilt $a_{kj} \geq \widetilde{a}^{\mathfrak{l}}_{kj}$ für alle $k \in \{1, ...m\}$ und $j \in \{1, ..., n\}$. Mit Bedingung (3.2) folgt:

$$\sum_{j=1}^{n} a_{kj} x_j \geq \sum_{j=1}^{n} \widetilde{a}^{\mathfrak{l}}_{kj} x_j \geq 1.$$

Hierbei sind a_{kj} bzw. $\widetilde{a}^{\mathfrak{l}}_{kj}$ die Einträge von A bzw. $\widetilde{A}^{\mathfrak{l}}$ in der k-ten Zeile und j-ten Spalte.

Also erfüllt x die Bedingung $Ax \geq \underline{1}_m$ und ist damit zulässig für (SCP). Wie für jede zulässige Lösung von (SCP) gilt für den Zielfunktionswert $c^t x \geq c^t x^\star$. □

Nachdem nun die theoretische Grundlage für die Berechnung von oberen Schranken für Mengenüberdeckungsprobleme durch Konstruktion einer C1P-Matrix $\widetilde{A}^{\mathfrak{l}}$ geschaffen wurde, schließt sich die Frage an, wie die Blöcke durch die Abbildung $\mathfrak{l}$ gewählt werden sollten, um möglichst gute, d. h. kleine obere Schranken zu erzeugen.

Die Grundidee liegt nun darin, für jede Zeile von A einen Kandidaten (d. h. einen Eins-Eintrag in der Zeile) mit möglichst „guten“ Eigenschaften zu finden und den Block zu wählen, der diesen Kandidaten enthält. Ein „guter“ Kandidat lässt sich durch Betrachtung des Verhältnisses von Kosten und Nutzen charakterisieren. Das heißt, die Kosten für die entsprechende Spalte sollen klein sein, dabei sollen aber durch diese Spalte möglichst viele Zeilen überdeckt werden.

Kosten bzw. Nutzen können auch separat betrachtet werden, wie es in [23] vorgeschlagen wird und auch von uns in Abschnitt 3.2 gemacht wird. Bei einem gegebenen (SCP) ist es aber oft schwer, zu entscheiden, welche der beiden Strategien zu besseren Ergebnissen führt. Daher betrachten wir in Abschnitt 3.3 das Verhältnis von Kosten und Nutzen für die Wahl des Kandidaten, was sich in den meisten Fällen als die beste Strategie erweist.

3.2 Blockauswahl „best-cover“

In diesem Abschnitt wollen wir einen Algorithmus zur Bestimmung einer oberen Schranke eines ungewichteten Mengenüberdeckungsproblems vorstellen. Da in

diesem Fall die Kosten für alle Spalten der Überdeckungsmatrix gleich sind, kann alleine der Nutzen eines Kandidaten für die Auswahl eines Blocks entscheidend sein. Konkret verfolgen wir die Strategie, für jede Zeile den Kandidaten zu finden, der die meisten Zeilen insgesamt überdeckt. Wir wählen für diese Zeilen dann den einen Block aus, der diesen Kandidaten enthält, und lösen das Problem (SCPu($\mathfrak{l}$)).

Wir formulieren nun zunächst den Algorithmus „best-cover“-Blockauswahl zur Bestimmung einer oberen Schranke von (SCP). Wir ändern dabei den Algorithmus von Ruf und Schöbel [23] leicht ab, indem wir in Schritt 1 des Algorithmus eine zusätzliche Bedingung an die Auswahl einer Spalte stellen.

Algorithmus 1 „best-cover“-Blockauswahl

Input: Überdeckungsmatrix $A \in \mathbb{B}^{m \times n}$ von (SCP).

Output: Zulässige Lösung x von (SCP).

1: **For** $k = 1, \ldots, m$:
Definiere $\mathfrak{l}(k) = i$ **if** $|cover(j)| = \max\limits_{j' \in cand(k)} \{|cover(j')|\}$ und $f_{k,i} \leq j \leq l_{k,i}$.
(Falls $\exists j_1, j_2 : j_1 < j_2, |cover(j_1)| = |cover(j_2)| = \max\limits_{j' \in cand(k)} \{|cover(j')|\}$*, wähle* j_1*).*

2: Löse (SCPu($\mathfrak{l}$)), sei x diese Lösung.

Output: x.

Bemerkung 3.2.1. *In Algorithmus 1 wird für jede Zeile der Überdeckungsmatrix genau ein Block ausgewählt und daraufhin das Problem (SCPu($\mathfrak{l}$)) aufgestellt und gelöst. Nach Lemma 3.1.3 berechnet der Algorithmus daher für beliebige Mengenüberdeckungsprobleme eine obere Schranke.*

Für gewichtete (SCP) führt der Algorithmus im Allgemeinen allerdings selten zu guten Ergebnissen, da die Blockauswahl unabhängig von den Kosten getroffen wird. Es können also Blöcke gewählt werden, dessen Kandidaten zwar große Überdeckungszahlen, dabei aber auch sehr hohe Kosten haben. Lösungen von (SCPu($\mathfrak{l}$)) kommen dann zwar mit wenigen zu wählenden Spalten aus, diese sind aber sehr teuer, was insgesamt zu einer großen, also schlechten oberen Schranke führt.

Definition 3.2.2. *Sei* $A \in \mathbb{B}^{m \times n}, N = \{1, ..., n\}$. *Die* maximale Überdeckungszahl *d von A wird definiert durch:* $d := \max\limits_{j \in N} |cover(j)|$.

Die Konstruktion der Matrix $\widetilde{A}^{\mathfrak{l}}$ in Algorithmus 1 ist so angelegt, dass $\widetilde{A}^{\mathfrak{l}}$ und A die gleiche maximale Überdeckungszahl haben.

Lemma 3.2.3. *Gegeben sei eine Matrix $A \in \mathbb{B}^{m\times n}$ mit maximaler Überdeckungszahl d. Dann hat die mit Algorithmus 1 aus A konstruierte Matrix $\widetilde{A}^{\mathfrak{l}}$ mindestens eine Spalte mit Überdeckungszahl d.*

Beweis. Falls es in A genau eine Spalte j mit Überdeckungszahl d gibt, so wird in Algorithmus 1 für jede Zeile aus cover(j) der Block gewählt, der diese Spalte enthält. Somit hat Spalte j von $\widetilde{A}^{\mathfrak{l}}$ die Überdeckungszahl d.

Falls es mehr als eine Spalte mit Überdeckungszahl d in A gibt, so sei j_1 diejenige unter diesen Spalten mit dem kleinsten Spaltenindex. Durch die Forderung in Schritt 1 von Algorithmus 1, dass bei mehreren potentiellen Kandidaten derjenige mit kleinerem Spaltenindex gewählt wird, ist gewährleistet, dass für alle Zeilen aus cover(j_1) der Block gewählt wird, welcher die Spalte j_1 enthält. Somit hat die Spalte j_1 von $\widetilde{A}^{\mathfrak{l}}$ die Überdeckungszahl d. □

Beispiel 3.2.4. *Gegeben sei ein Mengenüberdeckungsproblem (SCP) mit Überdeckungsmatrix*

$$A = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}$$

und Kosten $c = (1,1,1,1,1,1,1)^t$.

Eine Optimallösung von (SCP) ist durch $x^\star = (1,0,0,0,0,0,1)$ mit Zielfunktionswert $z^\star = 2$ gegeben.

Wenden wir nun die Vorschrift zur Konstruktion der Blockmatrix $\widetilde{A}$ aus Algorithmus 1 auf A an, so ergibt sich zunächst die Matrix

$$\widetilde{A} = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}.$$

Eine Optimallösung des Problems (SCPu(ı)) $\min\{c^T x : \widetilde{A}x \geq \underline{1}_6, x \in \mathbb{B}^7\}$ *ist dann gegeben durch* $\widetilde{x} = (1,1,0,1,0,1,0)$ *mit Zielfunktionswert* $\widetilde{z} = c^t\widetilde{x} = 4$. *Somit haben wir mit Algorithmus 1 die obere Schranke* $\widetilde{z} = 4$ *für (SCP) berechnet.*

3.2.1 Güte von Algorithmus 1 für ungewichtete (SCP)

Beispiel 3.2.4 gibt bereits Hinweise auf die Güte der oberen Schranke, die durch Algorithmus 1 berechnet wird. Das folgende Lemma, das die Beobachtungen aus Lemma 3.2.3 nutzt, gibt für ungewichtete Mengenüberdeckungsprobleme eine Gütegarantie des Algorithmus an.

Lemma 3.2.5. *Es sei ein ungewichtetes (SCP) mit Überdeckungsmatrix* $A \in \mathbb{B}^{m\times n}$ *und maximaler Überdeckungszahl* d *gegeben. Sei* $x^\star$ *die Optimallösung des Problems und es sei* $\widetilde{x}$ *eine durch Algorithmus 1 berechnete zulässige Lösung von (SCP). Dann gilt für die Zielfunktionswerte* $z^\star = \underline{1}_n^t x^\star$ *und* $\widetilde{z} = \underline{1}_n^t \widetilde{x}$*:*

1. $$|\widetilde{z} - z^\star| \leq m + 1 - d - \left\lceil \frac{m}{d} \right\rceil.$$

   Hierbei sei $\left\lceil \frac{m}{d} \right\rceil$ *die kleinste natürliche Zahl* $\geq \frac{m}{d}$.

2. *Für* $m \geq 5$ *gilt:*
   $$\frac{\widetilde{z}}{z^\star} \leq \frac{m}{3}.$$

Beweis. 1. Es sei (SCPu(ı)) das in Algorithmus 1 konstruierte Mengenüberdeckungsproblem mit Überdeckungsmatrix $\widetilde{A}$, dessen Lösung eine obere Schranke $\widetilde{z}$ für (SCP) liefert.

Die Idee des Beweises besteht darin, den Zielfunktionswert $\widetilde{z}$ der Lösung von Algorithmus 1 nach oben, und den optimalen Zielfunktionswert $z^\star$ nach unten abzuschätzen.

Zunächst betrachten wir die obere Schranke:

In der in Algorithmus 1 konstruierten Matrix $\widetilde{A}$ gibt es nach Lemma 3.2.3 mindestens eine Spalte mit Überdeckungszahl d. Um eine zulässige Lösung $\widetilde{x}$ von (SCPu(ł)) zu konstruieren, wählen wir zunächst diese Spalte. Für jede weitere der $m-d$ Zeilen von $\widetilde{A}$, die noch nicht überdeckt sind, wählen wir je eine Spalte, die diese Zeile überdeckt (dies ist möglich, da wir voraussetzen, dass A und somit auch $\widetilde{A}$ keine Nullzeile enthält). Durch die Wahl dieser insgesamt maximal $m-d+1$ Spalten (falls Spalten mehrfach für verschiedene Zeilen gewählt wurden, sind es weniger Spalten), erhalten wir eine zulässige Lösung und damit eine obere Schranke des Problems (SCPu(ł)). In einer optimalen Lösung von (SCPu(ł)) werden also höchstens $m-d+1$ Spalten gewählt und für den optimalen Zielfunktionswert des Problems gilt:

$$\widetilde{z} \leq m - d + 1.$$

Nun wollen wir die Anzahl der gewählten Spalten in einer Optimallösung von (SCP) nach unten abschätzen:

In A gibt es keine Spalte, die mehr als d Zeilen überdeckt. Für jede zulässige Lösung x von (SCP) müssen also mindestens $\left\lceil \frac{m}{d} \right\rceil$ verschiedene Spalten gewählt werden, damit alle Zeilen von A durch x überdeckt sind. Das gilt insbesondere auch für eine Optimallösung $x^\star$. Somit ist

$$z^\star \geq \left\lceil \frac{m}{d} \right\rceil .$$

Beachten wir noch, dass $\widetilde{z} \geq z^\star$, da $\widetilde{z}$ eine obere Schranke von (SCP) ist, dann erhalten wir:

$$|\widetilde{z} - z^\star| = \widetilde{z} - z^\star \leq m - d + 1 - z^\star \leq m - d + 1 - \left\lceil \frac{m}{d} \right\rceil .$$

2. Es sei $m \geq 5$. Wie in Teil 1. des Beweises gilt $\widetilde{z} \leq m - d + 1$ und $z^\star \geq \lceil \frac{m}{d} \rceil$ und damit
$$\frac{\widetilde{z}}{z^\star} \leq \frac{m - d + 1}{\lceil \frac{m}{d} \rceil}.$$
Wir betrachten nun alle verschiedenen Fälle die auftreten können, jeweils abhängig von der Größe der maximalen Überdeckungszahl d von A. Dabei ist zu beachten, dass d eine natürliche Zahl mit $1 \leq d \leq m$ ist.

- $m = d$: Es gilt
$$\frac{\widetilde{z}}{z^\star} \leq \frac{m - d + 1}{\lceil \frac{m}{d} \rceil} = \frac{1}{1} = 1 \leq \frac{m}{3}.$$

- $m > d \geq \frac{m}{3} + 1$: Es gilt:
$$\lceil \frac{m}{d} \rceil \geq 2 \text{ (da } d < m).$$
$$m - d + 1 \leq m - (\frac{m}{3} + 1) + 1 = \frac{2}{3}m \text{ (da } d \geq \frac{m}{3} + 1).$$
$$\Rightarrow \frac{\widetilde{z}}{z^\star} \leq \frac{m - d + 1}{\lceil \frac{m}{d} \rceil} \leq \frac{\frac{2}{3}m}{2} = \frac{m}{3}.$$

- $\frac{m}{3} + 1 > d > \frac{m}{3}$: Dieser Fall ist nur für $m \not\equiv 0 \pmod 3$ zu betrachten, denn falls $m \equiv 0 \pmod 3$, so ist $\frac{m}{3} \in \mathbb{N}$ und es gibt kein ganzzahliges $d \in (\frac{m}{3}, \frac{m}{3} + 1)$.
Zunächst betrachten wir den Spezialfall $m = 5$. Aus $\frac{5}{3} < d < \frac{5}{3} + 1$ folgt dann $d = 2$. Somit gilt
$$\frac{\widetilde{z}}{z^\star} \leq \frac{m - d + 1}{\lceil \frac{m}{d} \rceil} = \frac{4}{3} \leq \frac{5}{3} = \frac{m}{3}$$
und die Behauptung ist für $m = 5$ gezeigt. Der Fall $m = 6$ ist nicht zu betrachten, da $6 \equiv 0 \pmod 3$.
Sei nun $m \geq 7$ beliebig. Dann folgt:
$$\lceil \frac{m}{d} \rceil \geq \frac{m}{d} > \frac{m}{\frac{m}{3} + 1} = \frac{3m}{m + 3} > \frac{3m}{m + \frac{1}{2}m} = \frac{3m}{\frac{3}{2}m} = 2.$$

Da $\lceil \frac{m}{d} \rceil \in \mathbb{N}$ gilt somit $\lceil \frac{m}{d} \rceil \geq 3$. Außerdem folgt $1 + m - d \leq m$ wegen $d \geq 1$. Zusammen erhalten wir die behauptete Abschätzung:

$$\frac{\widetilde{z}}{z^\star} \leq \frac{m - d + 1}{\lceil \frac{m}{d} \rceil} \leq \frac{m}{3}.$$

- $\frac{m}{3} \geq d \geq 1$: Es gilt:

$$\lceil \frac{m}{d} \rceil \geq \frac{m}{d} \geq 3 \text{ (da } d \leq \frac{m}{3}).$$
$$m - d + 1 \leq m \text{ (da } d \geq 1).$$
$$\Rightarrow \frac{\widetilde{z}}{z^\star} \leq \frac{m - d + 1}{\lceil \frac{m}{d} \rceil} \leq \frac{m}{3}.$$

Da $1 \leq d \leq m$ und $d \in \mathbb{N}$ gilt, sind alle Fälle betrachtet worden und die Behauptung ist bewiesen.

□

Bemerkung 3.2.6. *Die zweite Abschätzung in Lemma 3.2.5 gilt nur für $m \geq 5$. Für $m \leq 3$ findet Algorithmus 1 immer eine Optimallösung, denn aus den möglichen maximalen Überdeckungszahlen $d \in \{1, 2, 3\}$ folgt für die Zielfunktionswerte:*

- *d=1* $\Rightarrow \widetilde{z} = z^\star = 3$
- *d=2* $\Rightarrow \widetilde{z} = z^\star = 2$
- *d=3* $\Rightarrow \widetilde{z} = z^\star = 1.$

Also gilt $\frac{\widetilde{z}}{z^\star} = 1$.
Falls $m = 4$ und $d = 2$ ist, so gilt die Abschätzung für den Approximationsfaktor im Allgemeinen nicht, denn für

$$A = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix}$$

ist die von Algorithmus 1 konstruierte Matrix

$$\widetilde{A} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix}.$$

Eine optimale Lösung des ungewichteten (SCP) mit Überdeckungsmatrix A ist gegeben durch $x = (1, 0, 0, 1)$, eine zulässige Lösung, die durch Algorithmus 1 berechnet wird, ist $\widetilde{x} = (1, 1, 1, 0)$. Somit gilt $\frac{\widetilde{z}}{z^\star} = \frac{3}{2}$. Da aber $\frac{3}{2} > \frac{4}{3} = \frac{m}{3}$, gilt die Abschätzung $\frac{\widetilde{z}}{z^\star} \leq \frac{m}{3}$ für $m = 4$ nicht.

Bemerkung 3.2.7. *Für den* relativen Fehler ε_r *von Algorithmus 1 gilt für $m \geq 5$:*

$$\varepsilon_r = \frac{|\widetilde{z} - z^\star|}{z^\star} = \frac{\widetilde{z} - z^\star}{z^\star} = \frac{\widetilde{z}}{z^\star} - 1 \leq \frac{m}{3} - 1 = \frac{m-3}{3}.$$

Die in Lemma 3.2.5 angegebenen Abschätzungen geben uns sowohl ein Maß für den absoluten, als auch für den relativen Fehler einer oberen Schranke, die von Algorithmus 1 für ein ungewichtetes Mengenüberdeckungsproblem berechnet wird.

Der relative Fehler hängt dabei linear von der Anzahl der Zeilen m der Überdeckungsmatrix A ab. Dies ist im Vergleich mit anderen Verfahren, bei denen der relative Fehler logarithmisch von m abhängt (siehe Abschnitt 3.3.3), ein schwaches Ergebnis für die Gütegarantie. Nichtsdestotrotz zeigt sich in der Praxis, dass der Algorithmus die angegebene Güteabschätzung weit unterschreitet und seine Anwendung durchaus sinnvoll sein kann.

Allerdings sind die angegebenen Abschätzungen scharf, wie das folgende Beispiel zeigt.

Beispiel 3.2.8. *Wir betrachten noch einmal das Mengenüberdeckungsproblem aus Beispiel 3.2.4. Die von Algorithmus 1 berechnete obere Schranke hat den Zielfunk-*

tionswert $\widetilde{z} = 4$. Der optimale Zielfunktionswert ist $z^\star = 2$. Die Anzahl der Zeilen von A ist $m = 6$, die maximale Überdeckungszahl $d = 3$. Somit gilt

$$\begin{aligned} |\widetilde{z} - z^\star| &= 4 - 2 = 2 = m + 1 - d - \left\lceil \frac{m}{d} \right\rceil, \\ \frac{\widetilde{z}}{z^\star} &= \frac{4}{2} = \frac{m}{3} \quad \textit{und} \\ \varepsilon_r &= \frac{|\widetilde{z} - z^\star|}{z^\star} = \frac{2}{2} = \frac{m-3}{3}. \end{aligned}$$

Absoluter und relativer Fehler, sowie der Approximationsfaktor von Algorithmus 1 erreichen in diesem Beispiel also genau die in Lemma 3.2.5 bewiesenen Güteabschätzungen.

3.3 Blockauswahl basierend auf Greedy-Heuristiken

In diesem Abschnitt wollen wir einen Algorithmus zur Bestimmung einer oberen Schranke von (SCP) entwickeln, der auf der bekannten Greedy-Heuristik für Mengenüberdeckungsprobleme (siehe [10]) basiert. Konkret soll die Blockauswahl zum Aufstellen eines Problems (SCPu($\mathfrak{l}$)) nach Vorschrift der Greedy-Heuristik geschehen. Dadurch wird die Lösung der Greedy-Heuristik für (SCPu($\mathfrak{l}$)) zulässig sein. Die aus der Optimallösung des Problems (SCPu($\mathfrak{l}$)) entstehende obere Schranke von (SCP) wird dann mindestens so gut sein, wie die durch die Greedy-Heuristik berechnete Schranke. Bekannte Güteabschätzungen für die Greedy-Heuristik können wir somit auf den von uns entwickelten Algorithmus übertragen.

Zunächst werden wir in Abschnitt 3.3.1 die Greedy-Heuristik in Erinnerung rufen, um dann in Abschnitt 3.3.2 unseren Algorithmus zu präsentieren. Darauf folgend werden wir in 3.3.3 eine Güteabschätzung für den Algorithmus angegeben. Abschnitt 3.3.4 gibt Varianten des Algorithmus basierend auf bekannten Varianten der Greedy-Heuristik an. Abschließend werden wir den Algorithmus für den Fall des ungewichteten Mengenüberdeckungsproblems mit dem in 3.2 entwickelten Algorithmus 1 vergleichen.

3.3.1 Greedy-Heuristik

Wir wollen im nächsten Abschnitt einen Algorithmus beschreiben, der die bekannte Greedy-Heuristik und dessen Güteabschätzungen nutzt. Zunächst geben wir daher die Greedy-Heuristik an, wie sie in [21] beschrieben wird. Die Idee ist, in jedem Schritt der Heuristik diejenige Spalte zu wählen, die das beste Verhältnis von Kosten und Überdeckungszahl hat. Die von der gewählten Spalte überdeckten Zeilen werden in den weiteren Schritten nicht mehr beachtet. Dieses Verfahren wird fortgesetzt, bis alle Zeilen überdeckt werden. Die gewählten Spalten erzeugen dann eine zulässige Lösung von (SCP).

Algorithmus 2 Greedy-Heuristik

Input: Kostenvektor $c \in \mathbb{R}^n_+$, Überdeckungsmatrix $A \in \mathbb{B}^{m\times n}$, $M^1 = \{1, ..., m\}$, $N^1 = \{1, ..., n\}$, $t = 1$.

Output: Zulässige Lösung x_G von (SCP).

1: Wähle $j^t \in N^t$ zu $\min\limits_{j\in N^t}\{\frac{c_j}{|\text{cover}(j)\cap M^t|}\}$.
(Falls $\exists j_1, j_2 \in N^t : j_1 < j_2$, $\frac{c_{j_1}}{|\text{cover}(j_1)\cap M^t|} = \frac{c_{j_2}}{|\text{cover}(j_2)\cap M^t|} = \min\limits_{j\in N^t}\{\frac{c_j}{|\text{cover}(j)\cap M^t|}\}$, wähle j_1).

2: $M^{t+1} = M^t \backslash \text{cover}(j^t)$, $N^{t+1} = N^t \backslash \{j^t\}$.

3: **If** $M^{t+1} \neq \emptyset : t = t + 1$. **Go to** 1.

4: Die Greedy-Lösung ist gegeben durch: $x_{G_j} = \begin{cases} 0, \text{ falls } j \in N^{t+1} \\ 1, \text{ sonst.} \end{cases}$

Output: x_G.

Der Algorithmus stoppt, wenn alle Zeilen $M = \{1, ..., m\}$ von A durch die ausgewählten Spalten überdeckt werden. Daher ist die Lösung x_G von Algorithmus 2 zulässig für (SCP). Der Zusatz in Schritt 1 des Algorithmus garantiert die Eindeutigkeit der Lösung.

Für Algorithmus 2 gilt die folgende Gütegarantie:

Lemma 3.3.1. *Es sei $z^\star$ der optimale Zielfunktionswert eines Mengenüberdeckungsproblems (SCP) mit Überdeckungsmatrix $A \in \mathbb{B}^{m\times n}$ und Kosten $c \in \mathbb{R}^n_+$. Sei x_G die von Algorithmus 2 berechnete zulässige Lösung von (SCP) mit Ziel-*

funktionswert $z_G := c^t x_G$. *Außerdem sei* $H(n) = \sum_{k=1}^{n} \frac{1}{k}$ *für* $n \in \mathbb{N}$ *und* d *die maximale Überdeckungszahl von* A. *Dann gilt:*

$$\frac{z_G}{z^\star} \leq H(d) \leq \log(d) + 1. \tag{3.4}$$

Beweis. Nach [21] gilt für den Approximationsfaktor der Greedy-Heuristik und damit für z_G die folgende Abschätzung:

$$\frac{z_G}{z^\star} \leq H(d).$$

Außerdem gilt nach [11]: $H(d) = \sum_{k=1}^{d} \frac{1}{k} \leq \log(d) + 1$. Damit ist das Lemma bewiesen. □

Bemerkung 3.3.2. *Für den* relativen Fehler ε_r *von Algorithmus 2 gilt:*

$$\varepsilon_r = \frac{|z_G - z^\star|}{z^\star} = \frac{z_G - z^\star}{z^\star} \leq H(d) - 1 \leq \log(d).$$

3.3.2 Greedy-Blockauswahl

Wir wollen nun die Idee der Greedy-Heuristik für die Auswahl der Blöcke der Überdeckungsmatrix $\widetilde{A}$ von (SCPu($\mathfrak{l}$)) nutzen. Das heißt konkret, wir wenden die Greedy-Heuristik auf (SCP) an, schieben aber in jedem Schritt der Heuristik noch eine Vorschrift zur Blockauswahl ein. So wird sukzessive für alle Zeilen genau ein Block ausgewählt. Durch die Lösung des Problems (SCPu($\mathfrak{l}$)) mit der entstandenen Blockmatrix $\widetilde{A}$ erhalten wir eine zulässige Lösung von (SCP).

Zunächst geben wir den Algorithmus an:

Algorithmus 3 Greedy-Blockauswahl

Input: Kostenvektor $c \in \mathbb{R}^n_+$, Überdeckungsmatrix $A \in \mathbb{B}^{m\times n}$, $M^1 = \{1, ..., m\}$, $N^1 = \{1, ..., n\}$, $t = 1$.

Output: Zulässige Lösung $\widetilde{x}$ von (SCP).

1: Wähle $j^t \in N^t$ zu $\min\limits_{j\in N^t}\{\frac{c_j}{|\text{cover}(j)\cap M^t|}\}$.
(Falls $\exists j_1, j_2 \in N^t : j_1 < j_2$, $\frac{c_{j_1}}{|\text{cover}(j_1)\cap M^t|} = \frac{c_{j_2}}{|\text{cover}(j_2)\cap M^t|} = \min\limits_{j\in N^t}\{\frac{c_j}{|\text{cover}(j)\cap M^t|}\}$, wähle j_1.)

2: **For** $k \in \text{cover}(j^t) \cap M^t$ **do**
definiere $\mathfrak{l}(k) = i$, **if** $f_{k,i} \le j^t \le l_{k,i}$.

3: $M^{t+1} = M^t \backslash \text{cover}(j^t)$, $N^{t+1} = N^t \backslash \{j^t\}$.

4: **If** $M^{t+1} \neq \emptyset : t = t + 1$. **Go to** 1.

5: Löse (SCPu($\mathfrak{l}$)), sei $\widetilde{x}$ diese Lösung.

Output: $\widetilde{x}$.

Da die in Algorithmus 3 definierte Abbildung $\mathfrak{l}$ jeder Zeile von A genau ein Block zuordnet, ist sie eine Blockabbildung der Matrix A und somit ist die Lösung $\widetilde{x}$ des entstehenden Problems (SCPu($\mathfrak{l}$)) nach Lemma 3.1.3 zulässig für (SCP).

Algorithmus 3 hat die gleiche Idee wie Algorithmus 2, nämlich in jedem Schritt die Spalte zu wählen, die die meisten noch nicht überdeckten Zeilen pro Einheitskosten überdeckt. Algorithmus 3 belässt es aber nicht bei diesen Spalten, sondern konstruiert hieraus zunächst die Blockmatrix $\widetilde{A}$, welche in jeder Zeile genau den einen Block aufeinanderfolgender Einsen hat, in dem die von der Greedy-Heuristik ausgewählte Spalte enthalten ist. Auf Grund der Eindeutigkeitsforderung in Schritt 1 des Algorithmus werden bei dem Durchlauf von Algorithmus 3 exakt

die selben Spalten gewählt, wie in Algorithmus 2. Da die Blöcke der Matrix $\widetilde{A}$ genau aus diesen Spalten konstruiert werden, ist die Lösung x_G von Algorithmus 2 zulässig für das Problem (SCPu(ł)) mit Überdeckungsmatrix $\widetilde{A}$ aus Algorithmus 3. Das heißt also, dass $z_G = c^t x_G$ eine obere Schranke für (SCPu(ł)) ist.

Da der Zielfunktionswert $\widetilde{z} := c^t \widetilde{x}$ der Lösung $\widetilde{x}$ aus Algorithmus 3 höchstens so groß ist wie z_G, ist auch der Wert der oberen Schranke, die aus Algorithmus 3 für (SCP) entsteht, nie schlechter als die obere Schranke für (SCP), die durch die Greedy-Heuristik berechnet wird. Algorithmus 3 ist daher eine Verbesserung des Algorithmus 2.

Für ein (SCP) mit Überdeckungsmatrix $A \in \mathbb{B}^{m \times n}$ hat die Auswahl der Blöcke in Schritt 2 des Algorithmus insgesamt eine zusätzliche Laufzeit von $\mathcal{O}(mn)$ und die Lösung des Problems (SCPu(ł)) in Schritt 5 nach Bemerkung 1.3.4 eine Laufzeit von $\mathcal{O}(m+n)$. Algorithmus 3 hat demnach nur einen wenig größeren Aufwand als die Greedy-Heuristik und stellt somit wegen seines nie schlechteren Ergebnisses eine sinnvolle Erweiterung der Greedy-Heuristik dar.

Bisher haben wir nur theoretisch festgestellt, dass die Lösung von Algorithmus 3 nie schlechter als die Lösung von Algorithmus 2 ist. Das folgende Beispiel zeigt, dass es tatsächlich zu einer echten Verbesserung durch Algorithmus 3 kommen kann.

In Kapitel 4 werden wir außerdem testen, für welche Art von Problemen Algorithmus 3 im Allgemeinen echt bessere obere Schranken als die Greedy-Heuristik liefert und wie groß diese Verbesserung ist.

Beispiel 3.3.3. *Gegeben sei ein Mengenüberdeckungsproblem (SCP) mit Überdeckungsmatrix*

$$A = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \end{pmatrix}$$

und Kosten $c = (4, 5, 3, 5, 1, 1)^t$.

Zunächst wenden wir Algorithmus 2 auf A und c an:

1. $M^1 = \{1, 2, 3, 4, 5\}, N^1 = \{1, 2, 3, 4, 5, 6\}$.

2. $j^1 = 5, M^2 = \{1,2,3,4\}, N^2 = \{1,2,3,4,6\}$.

3. $j^2 = 6, M^3 = \{2,3,4\}, N^3 = \{1,2,3,4\}$.

4. $j^3 = 3, M^4 = \{2\}, N^4 = \{1,2,4\}$.

5. $j^4 = 1, M^5 = \emptyset, N^5 = \{2,4\}$.

$\Rightarrow x_G = (1,0,1,0,1,1)$, $z_G = c^t x_G = 9$.

Durch Algorithmus 3 wird die folgende Blockmatrix $\widetilde{A}$ *konstruiert:*

$$\widetilde{A} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \end{pmatrix}.$$

Eine Optimallösung des Problems $(SCPu(\mathfrak{l}))$ $\min\{c^t x : \widetilde{A}x \geq \underline{1}_5, x \in \mathbb{B}^6\}$ *und damit eine Lösung von Algorithmus 3 ist gegeben durch* $\widetilde{x} = (0,1,0,0,1,1)$ *mit Zielfunktionswert* $\widetilde{z} = c^t\widetilde{x} = 7$.

Wir haben durch die zwei Algorithmen also jeweils eine zulässige Lösung von (SCP) $\min\{c^t x : Ax \geq \underline{1}_5, x \in \mathbb{B}^6\}$ *konstruiert. Dabei ist* x_G *insbesondere auch für (SCPu(*$\mathfrak{l}$*)) zulässig.*

Die obere Schranke, die durch Algorithmus 3 berechnet wurde, ist echt besser als die heuristische Lösung durch Algorithmus 2. Tatsächlich handelt es sich sogar um eine Optimallösung von (SCP).

Vergleichen wir die beiden Lösungen miteinander, so bekommen wir eine relative Verbesserung $\frac{|z_G - \widetilde{z}|}{z_G} = \frac{2}{9} \approx 0,22$ *von Algorithmus 3 gegenüber der Greedy-Heuristik. Das heißt, die beiden Lösungen weichen um ca. 22 % voneinander ab.*

3.3.3 Güte von Algorithmus 3

In diesem Abschnitt wollen wir eine Gütegarantie für die von Algorithmus 3 berechnete obere Schranke eines Mengenüberdeckungsproblems angeben. Diese basiert auf der in Lemma 3.3.1 angegebenen Gütegarantie für die Greedy-Heuristik.

Da Algorithmus 3 eine mindestens so gute obere Schranke für (SCP) liefert wie die Greedy-Heuristik, gilt dessen Güteabschätzung auch für Algorithmus 3.

Lemma 3.3.4. *Es sei $z^\star$ der optimale Zielfunktionswert eines Mengenüberdeckungsproblems (SCP) mit Überdeckungsmatrix $A \in \mathbb{B}^{m\times n}$ und Kosten $c \in \mathbb{R}^n_+$. Sei $\widetilde{x}$ die von Algorithmus 3 berechnete zulässige Lösung von (SCP) mit Zielfunktionswert $\widetilde{z} = c^t\widetilde{x}$. Außerdem sei $H(n) = \sum_{k=1}^{n} \frac{1}{k}$ für $n \in \mathbb{N}$ und d die maximale Überdeckungszahl von A. Dann gilt:*

$$\frac{\widetilde{z}}{z^\star} \leq H(d) \leq \log(d) + 1. \tag{3.5}$$

Beweis. Die Lösung x_G von Algorithmus 2 ist nach Konstruktion der Matrix $\widetilde{A}$ aus Algorithmus 3 zulässig für das Problem (SCPu(ł)). Da $\widetilde{x}$ eine Optimallösung dieses Minimierungsproblems ist, gilt

$$c^t\widetilde{x} \leq c^t x_G.$$

Da außerdem $\widetilde{z} \geq 0$ ist, folgt mit der Abschätzung der Güte von $z_G = c^t x_G$ aus Lemma 3.3.1:

$$\frac{\widetilde{z}}{z^\star} \leq \frac{z_G}{z^\star} \leq H(d) \leq \log(d) + 1.$$

□

Bemerkung 3.3.5. *Für den* relativen Fehler ε_r *von Algorithmus 3 gilt:*

$$\varepsilon_r = \frac{|\widetilde{z} - z^\star|}{z^\star} = \frac{\widetilde{z} - z^\star}{z^\star} \leq H(d) - 1 \leq \log(d).$$

Der größte relative Fehler von Algorithmus 3 wird nach Lemma 3.3.4 nur logarithmisch mit der Größe der maximalen Überdeckungszahl d eines (SCP) größer. Da d kleiner oder gleich der Anzahl der Zeilen der Überdeckungsmatrix A des Problems ist, wird der relative Fehler insbesondere auch nur logarithmisch mit der Anzahl der Zeilen von A größer. Außerdem ist festzuhalten, dass die Güteabschätzung für Algorithmus 3 weder von der Anzahl der Spalten von A, noch von den gegebenen Kosten von (SCP) abhängig ist.

Die Abhängigkeit des relativen Fehlers einzig vom Logarithmus der maximalen Überdeckungszahl ist eine sehr starke Gütegarantie für Algorithmus 3. Nichtsdestotrotz ist diese Angabe scharf, wie das folgende Beispiel zeigt.

Beispiel 3.3.6. *Wir geben ein Beispiel an, bei dem $\frac{\widetilde{z}}{z^\star}$ beliebig nah an $H(d)$ angenähert werden kann. Sei dazu $\frac{5}{6} > \varepsilon > 0$ und seien*

$$A = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix}$$

die Überdeckungsmatrix und $c = (1, \frac{1}{2}, \frac{1}{3}, 1+\varepsilon)$ die Kosten eines Mengenüberdeckungsproblems (SCP). Die maximale Überdeckungszahl von A ist $d = 3$.
Die Optimallösung von (SCP) ist $x^\star = (0,0,0,1)$ mit Zielfunktionswert $z^\star = 1+\varepsilon$.
In Algorithmus 3 wird die folgende Matrix $\widetilde{A}$ konstruiert:

$$\widetilde{A} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix}.$$

Eine Lösung von (SCPu(ι)) und damit das Ergebnis von Algorithmus 3 ist gegeben durch $\widetilde{x} = (1,1,1,0)$ mit Zielfunktionswert $\widetilde{z} = 1 + \frac{1}{2} + \frac{1}{3} = H(d)$.
Somit gilt $\frac{\widetilde{z}}{z^\star} = \frac{H(d)}{1+\varepsilon}$. Für beliebig kleine ε nähert sich dieser Ausdruck beliebig an $H(d)$ an.

Bemerkung 3.3.7. *Slavík [25] gibt für die Greedy-Heuristik für (SCP) einen Approximationsfaktor von $(1 - o(1)) \ln m$ für $m \geq 2$ (hierbei ist o das Landau-Symbol) statt des von uns angegebenen Faktors von $H(d)$ an. Dieser hängt zwar von der gesamten Anzahl der Zeilen m und nicht von der im Allgemeinen kleineren maximalen Überdeckungszahl d ab, kann aber kleiner als $H(d)$ sein, da $\ln m < H(m) \, \forall m \geq 2$ (siehe [11]).*

Außerdem zeigt Feige [12] unter der Annahme $P \neq NP$, dass es keinen Algorithmus mit polynomieller Laufzeit gibt, der einen Approximationsfaktor kleiner als $(1 - o(1)) \ln m$ für (SCP) hat. Zusammen folgt also, dass es, falls $P \neq NP$, keinen Algorithmus mit einem besseren Approximationsfaktor gibt, als die Greedy-Heuristik. Da die in Lemma 3.3.4 angegebene Güteabschätzung für Algorithmus 3

ausschließlich von der Güte der Greedy-Heuristik abhängt, gilt diese Aussage insbesondere auch für Algorithmus 3. Wir haben also, falls $P \neq NP$, einen Algorithmus mit dem besten möglichen Approximationsfaktor für (SCP) unter allen polynomiellen Algorithmen angegeben.

3.3.4 Andere Gewichtungsfunktionen in Algorithmus 3

Die in Schritt 1 von Algorithmus 3 getroffene Gewichtung der noch nicht überdeckten Spalten im Verhältnis zu ihren Kosten kann variiert werden, um entweder die Kosten oder die Überdeckungseigenschaften der Spalte höher zu bewerten. Einige Vorschläge dieser Art werden von Balas [2] für die Greedy-Heuristik gemacht. Diese können für Algorithmus 3 übernommen werden.

Zur einfacheren Notation definieren wir dazu im Durchlauf t des Algorithmus $k_j := |\text{cover}(j) \cap M^t|$ für alle $j \in N^t$. Dann sind mögliche Gewichtungsfunktionen gegeben durch

$$(i)\ \frac{c_j}{k_j},\quad (ii)\ \frac{c_j}{\log_2 k_j},\quad (iii)\ \frac{c_j}{k_j \log_2 k_j},\quad (iv)\ \frac{c_j}{k_j \ln k_j}.$$

In den Fällen (ii) und (iii) wird $\log_2 k_j$ durch 1 ersetzt, falls $k_j = 1$. Im Fall (iv) wird $\ln k_j$ durch 1 ersetzt, falls $k_j \in \{1, 2\}$.

Möglichkeit (i) ist diejenige, die in der Formulierung von Algorithmus 3 benutzt wurde. Die Varianten (ii), (iii) und (iv) wählen die selben Spalten wie (i) aus, falls das Problem ungewichtet ist, also $c = \underline{1}_n$. Im Fall eines gewichteten Mengenüberdeckungsproblems ordnet (ii) der Anzahl k_j der überdeckten Zeilen weniger Gewicht im Verhältnis zu den Kosten der Spalte zu als (i), wohingegen (iv) dieser Anzahl k_j größeres und (iii) ein noch größeres Gewicht im Verhältnis zu den Kosten gibt. Für keine der Varianten (ii), (iii) und (iv) kann eine so starke Güteabschätzung angegeben werden, wie sie für (i) in Lemma 3.3.4 bewiesen wurde. Für bestimmte Probleme mit großer Kostenspanne oder großer Streuung der Überdeckungszahlen der verschiedenen Spalten kann der Einsatz dieser Varianten in Algorithmus 3 aber bessere Ergebnisse als (i) erzielen.

3.3.5 Vergleich von „best-cover“- und Greedy-Blockauswahl

In Abschnitt 3.2 haben wir Algorithmus 1 zur Blockauswahl von ungewichteten Mengenüberdeckungsproblemen vorgestellt. In diesem Abschnitt soll nun auch Algorithmus 3 auf das ungewichtete Mengenüberdeckungsproblem angewendet werden. In Algorithmus 1 wurde für alle Zeilen der Matrix A jeweils der Block der „best-cover“-Spalte gewählt. Im Greedy-Verfahren hingegen werden nach und nach jeweils die „best-cover“-Spalten gewählt und dann die entsprechenden Zeilen für die nächste Auswahl der „best-cover“-Spalte herausgenommen und nicht mehr beachtet. Hierdurch entwickelt sich im Allgemeinen ein effektiveres Verfahren, da verhindert wird, dass Zeilen mehrfach überdeckt werden. Entsprechend sind die theoretischen Ergebnisse für Algorithmus 3 deutlich besser als die Ergebnisse für Algorithmus 1. Dennoch zeigt sich, dass jeder der Algorithmen für bestimmte Probleme besser sein kann als der andere.

Zunächst formulieren wir Algorithmus 3 noch einmal speziell für ungewichtete (SCP).

Algorithmus 4 Greedy-Blockauswahl für ungewichtete (SCP)

Input: Überdeckungsmatrix $A \in \mathbb{B}^{m\times n}$, $M^1 = \{1, ..., m\}$, $N^1 = \{1, ..., n\}$, $t = 1$.

Output: Zulässige Lösung $\widetilde{x}$ von (SCP).

1: Wähle $j^t \in N^t$ zu $\max\limits_{j\in N^t}\{|\text{cover}(j) \cap M^t|\}$.
(Falls $\exists j_1, j_2 \in N^t : j_1 < j_2$, $|\text{cover}(j_1)\cap M^t| = |\text{cover}(j_2)\cap M^t| = \max\limits_{j\in N^t} |\text{cover}(j)\cap M^t|$, wähle j_1.)

2: **For** $k \in \text{cover}(j^t) \cap M^t$ **do**
definiere $\mathfrak{l}(k) = i$, **if** $f_{k,i} \leq j^t \leq l_{k,i}$.

3: $M^{t+1} = M^t \backslash \text{cover}(j^t)$, $N^{t+1} = N^t \backslash \{j^t\}$.

4: **If** $M^{t+1} \neq \emptyset : t = t + 1$. **Go to** 1.

5: Löse (SCPu($\mathfrak{l}$)), sei $\widetilde{x}$ diese Lösung.

Output: $\widetilde{x}$.

Bemerkung 3.3.8. *Algorithmus 4 unterscheidet sich im Prinzip nicht von Algorithmus 3. Da die Kosten für alle Spalten gleich sind, können aber in Durch-*

lauf t des Algorithmus durch die Wahl des Maximums $\max\limits_{j\in N^t}\{|cover(j)\cap M^t|\}$ *statt* $\min\limits_{j\in N^t}\{\frac{1}{|cover(j)\cap M^t|}\}$ *im ersten Schritt des Algorithmus* $|N^t|$ *Divisionen gespart werden.*

Für die obere Schranke, die durch Algorithmus 4 berechnet wird, bekommen wir die gleiche Güteabschätzung, wie wir sie in Lemma 3.3.4 für Algorithmus 3 angegeben haben, denn diese war unabhängig von der Kostenfunktion $c \in \mathbb{R}^n_+$ des Problems.

Lemma 3.3.9. *Es sei $z^\star$ der optimale Zielfunktionswert eines ungewichteten (SCP) mit Überdeckungsmatrix A und $\widetilde{z} = \underline{1}^t_n \widetilde{x}$ der Zielfunktionswert der von Algorithmus 4 berechneten zulässigen Lösung $\widetilde{x}$ von (SCP). Außerdem sei $H(n) = \sum_{k=1}^{n} \frac{1}{k}$ für $n \in \mathbb{N}$ und d die maximale Überdeckungszahl von A. Dann gilt:*

$$\frac{\widetilde{z}}{z^\star} \leq H(d) \leq \log(d) + 1. \tag{3.6}$$

Bemerkung 3.3.10. *Für den* relativen Fehler ε_r *von Algorithmus 4 gilt:*

$$\varepsilon_r = \frac{|\widetilde{z} - z^\star|}{z^\star} = \frac{\widetilde{z} - z^\star}{z^\star} \leq H(d) - 1 \leq \log(d).$$

Der maximale relative Fehler der von Algorithmus 4 berechneten oberen Schranke für ein ungewichtetes (SCP) hängt logarithmisch von der maximalen Überdeckungszahl d der gegebenen Überdeckungsmatrix A ab. Im Gegensatz dazu ist der maximale relative Fehler der von Algorithmus 1 berechneten oberen Schranke linear von der Anzahl m der Zeilen von A abhängig. Dies ist ein deutlich schwächeres Ergebnis. Da sich die Laufzeiten der beiden Algorithmen nicht signifikant voneinander unterscheiden, ist daher grundsätzlich Algorithmus 4 gegenüber Algorithmus 1 zu bevorzugen.

Das folgende Beispiel zeigt allerdings, dass die obere Schranke, die durch Algorithmus 1 erzeugt wird, echt besser sein kann als die von Algorithmus 4 berechnete. Es kann daher sinnvoll sein, auf ein ungewichtetes (SCP) zusätzlich zu Algorithmus 4 auch Algorithmus 1 anzuwenden und das bessere der beiden Ergebnisse als obere Schranke des Problems zu nutzen.

Beispiel 3.3.11. *Gegeben sei ein ungewichtetes (SCP) mit Überdeckungsmatrix*

$$A = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \end{pmatrix}.$$

Betrachten wir zunächst Algorithmus 4. Dieser liefert die folgende Blockmatrix:

$$\widetilde{A} = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \end{pmatrix}.$$

Dann ist eine obere Schranke von (SCP), die aus einer Lösung des Problems $\min\{\underline{1}_n^T x : \widetilde{A}x \geq \underline{1}_6, x \in \mathbb{B}^6\}$ *entsteht, gegeben durch* $\widetilde{x} = (1,0,1,1,0,0)$ *mit Zielfunktionswert* $\widetilde{z} = 3$. *Andererseits bekommen wir durch Algorithmus 1 die Blockmatrix*

$$A_u = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \end{pmatrix}$$

und die obere Schranke $x_u = (0,0,1,0,1,0)$ *mit Zielfunktionswert* $z_u = 2$.

3.4 Blockauswahl für beliebige zulässige Lösungen von (SCP)

In Abschnitt 3.3.2 haben wir in Algorithmus 3 eine Blockmatrix $\widetilde{A}$ derart konstruiert, dass eine Lösung x_G der Greedy-Heuristik zulässig für das aus $\widetilde{A}$ entstehende Problem (SCPu($\mathfrak{l}$)) ist. Daraus folgt, dass die durch Algorithmus 3 berechnete obere Schranke für (SCP) mindestens so gut ist, wie die obere Schranke, die aus x_G entsteht. Die Lösung der Greedy-Heuristik wird also im schlechtesten Fall durch Algorithmus 3 nicht weiter verbessert, im positiven Fall liefert Algorithmus 3 eine echt bessere Lösung als die Greedy-Heuristik.

Dieses Verfahren lässt sich für beliebige zulässige Lösungen x von (SCP) anwenden. Das heißt, wir wollen aus einer zulässigen Lösung x von (SCP) eine Blockmatrix $\widetilde{A}$ aufstellen dergestalt, dass x auch für das aus $\widetilde{A}$ entstehende Programm (SCPu($\mathfrak{l}$)) zulässig ist. Wir stellen hierzu nun einen allgemeinen Algorithmus vor.

Algorithmus 5 Verbesserung einer zulässigen Lösung von (SCP) durch Blockauswahl

---

Input: Zulässige Lösung x von (SCP), Überdeckungsmatrix $A \in \mathbb{B}^{m \times n}$, Kosten $c \in \mathbb{R}^n$ von (SCP).

Output: Zulässige Lösung $\widetilde{x}$ von (SCP).

1: $F := \{j \in \{1, ..., n\} : x_j = 1\}$.

2: **For all** $k \in \{1, ..., m\}$ **do**

Finde $i_k \in \text{cand}(k) \cap F$.

(Falls $\exists i_1, i_2 \in \text{cand}(k) \cap F : i_1 < i_2$, wähle i_1.)

Definiere $\mathfrak{l}(k) := s$, **if** $f_{k,s} \leq i_k \leq l_{k,s}$.

3: Löse (SCPu($\mathfrak{l}$)), sei $\widetilde{x}$ diese Lösung.

Output: $\widetilde{x}$.

Bemerkung 3.4.1. *Für Algorithmus 5 wird eine zulässige Lösung x von (SCP) benötigt, damit in Schritt 2 des Algorithmus tatsächlich für jede Zeile k von A $cand(k) \cap F \neq \emptyset$ gilt und die Blockabbildung $\mathfrak{l}(k)$ für alle $k \in \{1, ..., m\}$ definiert werden kann.*

Lemma 3.4.2. *Sei $x \in \mathbb{B}^n$ eine zulässige Lösung des Problems (SCP) mit Überdeckungsmatrix $A \in \mathbb{B}^{m\times n}$ und Kosten $c \in \mathbb{R}^n$. Sei weiterhin (SCPu($\mathfrak{l}$)) das Problem, das in Algorithmus 5 (mit Input x, A und c) aufgestellt wird und $\widetilde{x}$ eine Optimallösung dieses Problems. Dann ist x zulässig für (SCPu($\mathfrak{l}$)) und es gilt $c^t\widetilde{x} \leq c^t x$.*

Beweis. Sei x eine zulässige Lösung von (SCP). Nach Bemerkung 3.4.1 ist die in Algorithmus 5 konstruierte Abbildung $\mathfrak{l} : \{1, .., m\} \to \mathbb{N}^m$ dann eine zulässige Blockabbildung für A und das Problem (SCPu($\mathfrak{l}$)) mit Überdeckungsmatrix $\widetilde{A}$ existiert. Seien a_{kj} bzw. $\widetilde{a}_{kj}$ die Einträge von A bzw. $\widetilde{A}$ in der k-ten Zeile und j-ten Spalte. Dann gilt nach Konstruktion von $\mathfrak{l}$ für die k-te Zeile $\widetilde{A}_k$ von $\widetilde{A}$, $k \in \{1, ..., m\}$:

$$\widetilde{A}_k x = \sum_{j=1}^{n} x_j \widetilde{a}_{kj} = \sum_{j=f_{k,\mathfrak{l}(k)}}^{l_{k,\mathfrak{l}(k)}} x_j a_{kj} \geq x_j a_{i_k j} = 1,$$

denn $i_k \in \mathrm{cand}(k) \cap F$ und $f_{k,\mathfrak{l}(k)} \leq i_k \leq l_{k,\mathfrak{l}(k)}$.

Somit gilt $\widetilde{A}x \geq \underline{1}_m$ und x ist zulässig für (SCPu($\mathfrak{l}$)). Insbesondere gilt $c^t x \geq c^t\widetilde{x}$. □

Beispiel 3.4.3. *Betrachten wir ein Mengenüberdeckungsproblem (SCP) mit Matrix*

$$A = \begin{pmatrix} 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 \end{pmatrix}$$

und Kosten $c = (2, 2, 3, 2, 1)^t$. Sei eine zulässige Lösung von (SCP) gegeben durch $x = (1, 0, 1, 0, 1)$ mit Zielfunktionswert $z = 6$.

Die Blockauswahl aus Algorithmus 5 führt zu der Matrix

$$\widetilde{A} = \begin{pmatrix} 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix}.$$

Eine Optimallösung des entstehenden Problems (SCPu(ι)) ist gegeben durch $\widetilde{x} = (0, 1, 0, 1, 0)$ *mit Zielfunktionswert* $\widetilde{z} = 4$.

Durch Algorithmus 5 wurde aus der Lösung x *also eine zulässigen Lösung* $\widetilde{x}$ *von (SCP) gewonnen. Dabei ist zu beachten, dass* x *auch für das Problem (SCPu(ι)) zulässig ist, wie es in Lemma 3.4.2 bewiesen wurde.*

Bemerkung 3.4.4. *Algorithmus 3 wurde als eine Erweiterung speziell für die Greedy-Heuristik entwickelt. Algorithmus 5 verallgemeinert diese Idee und entwickelt aus beliebigen zulässigen Lösungen* x *von (SCP) eine zulässige Lösung* $\widetilde{x}$ *von (SCP) mit einem nicht schlechterem Zielfunktionswert als* x*. Eine echte Verbesserung des Zielfunktionswertes durch Algorithmus 5 wird vor allem dann auftreten, wenn die Überdeckungsmatrix* A *von (SCP) eine Blockstruktur hat, d. h. wenn insgesamt nur wenige Blöcke in* A *auftreten und diese Blöcke möglichst lang sind.*

Zum Aufwand von Algorithmus 5 trägt zum einen die Auswahl der Blöcke in Schritt 2 mit einer Laufzeit von $\mathcal{O}(mn)$ *bei. Zum anderen hat die Lösung des C1P-Mengenüberdeckungsproblems (SCPu(ι)) in Schritt 3 des Algorithmus nach Bemerkung 1.3.4 eine Laufzeit von* $\mathcal{O}(m + n)$.

Der geringe Aufwand von Algorithmus 5 und die in Lemma 3.4.2 angegebene Gütegarantie für die Lösung des Algorithmus rechtfertigen in jedem Fall eine Anwendung des Algorithmus auf eine bekannte zulässige Lösung x *von (SCP). Insbesondere wenn die Überdeckungsmatrix* A *von (SCP) eine Blockstruktur aufweist, bekommen wir durch Algorithmus 5 in vielen Fällen eine echt bessere Lösung als sie durch* x *gegeben war.*

Zulässige Lösungen von (SCP) können in vielfältiger Weise gewonnen werden. Zum einen kann eine zulässige Lösung geraten werden. Zum anderen können bekannte Verfahren, wie z. B. Innere Punkte Methoden [19], Genetische Algorithmen [5], die bereits vorgestellte Greedy-Heuristik [10] oder Varianten der Greedy-Heu-

ristik [13, 2, 15] genutzt werden, um zulässige Lösungen von (SCP) zu erhalten. Für all diese Verfahren kann Algorithmus 5 für eine mögliche Verbesserung der Lösung zusätzlich eingesetzt werden.

3.5 Erweiterung einer Relaxationslösung durch Blockauswahl

Von Schöbel und Ruf [23] wird ein Verfahren vorgestellt, bei dem zunächst eine Optimallösung $\mathring{x}$ von (SCPl) bestimmt wird. Aus dieser Lösung wird dann ein reduziertes Mengenüberdeckungsproblem (Red-SCP) aufgestellt und anschließend gelöst. Durch Kombination der beiden Lösungen von (SCPl) und (Red-SCP) entsteht eine zulässige Lösung von (SCP), die insbesondere eine obere Schranke für (SCP) erzeugt.

Wir wollen in diesem Abschnitt das Verfahren erweitern, indem wir sowohl Lösungen von (SCPl), als auch von (S$\mathring{\text{C}}$Pl) zur Identifikation des Problems (Red-SCP) zulassen.

Sei nun also ein Mengenüberdeckungsproblem (SCP) mit $A \in \mathbb{B}^{m \times n}$ und $c \in \mathbb{R}^n_+$ gegeben und sei $\mathring{x}$ eine Optimallösung von (SCPl) oder (S$\mathring{\text{C}}$Pl). Wir definieren durch $\mathcal{M}^\circ := \{k \in \{1, ..., m\} : A_k \mathring{x} = 0\}$ die Menge der Zeilen von A, die nicht durch $\mathring{x}$ überdeckt werden. Hierbei ist A_k die k-te Zeile von A.

Falls $\mathcal{M}^\circ = \emptyset$, so ist $\mathring{x}$ Optimallösung von (SCP), da $\mathring{x}$ dann zulässig für (SCP) ist und $\mathring{x}$ außerdem nach Lemma 2.1.7 eine untere Schranke von (SCP) erzeugt.

Sonst definieren wir die Matrix $B \in \mathbb{B}^{|\mathcal{M}^\circ| \times n}$, welche genau die Zeilen von A enthält, die nicht durch $\mathring{x}$ überdeckt werden. Nun betrachten wir das reduzierte Mengenüberdeckungsproblem

$$\begin{array}{lll} & \text{minimiere} & c^t x \\ \textbf{(Red-SCP)} & \text{so dass} & Bx \geq \underline{1}_{|\mathcal{M}^\circ|} \\ & & x \in \mathbb{B}^n. \end{array} \tag{3.7}$$

Sei $\widetilde{x}$ eine zulässige Lösung von (Red-SCP). Wir bekommen das folgende Ergebnis:

Lemma 3.5.1. *(siehe [23]) Sei $x^\star$ eine optimale Lösung von (SCP). Weiterhin sei $\mathring{x}$ eine Optimallösung von (S$\mathring{\text{C}}$Pl) (bzw. (SCPl)) und $\widetilde{x}$ eine zulässige Lösung von (Red-SCP). Dann ist x, gegeben durch*

$$x_j = \max\{\mathring{x}_j, \widetilde{x}_j\}, \quad j \in \{1, ..., n\},$$

zulässig für (SCP). Insbesondere gilt $c^T x \geq c^T x^\star$.

Beweis. Sei $\mathring{x}$ eine Optimallösung von (S$\mathring{\text{C}}$Pl) (bzw. von (SCPl)) und $\widetilde{x}$ eine zulässige Lösung des Problems (Red-SCP).

Wir zeigen, dass x zulässig für (SCP) ist, indem wir für jede Zeile von A prüfen, ob sie von x überdeckt wird. Dazu betrachten wir $\mathcal{M}^\circ = \{k \in \{1, ..., m\} : A_k \mathring{x} = 0\}$.

Sei zunächst $k \in \{1, ..., m\} \backslash \mathcal{M}^\circ$. Dann gilt

$$A_k x \geq A_k \mathring{x} \geq 1,$$

also sind diese Zeilen von x überdeckt.

Sei andererseits $k \in \mathcal{M}^\circ$. Dann gibt es genau ein $j_k \in \{1, ..., |\mathcal{M}^\circ|\}$, so dass $B_{j_k} = A_k$. Somit ist

$$A_j x = B_{j_k} x \geq B_{j_k} \widetilde{x} \geq 1$$

und auch diese Zeilen sind von x überdeckt. Insgesamt überdeckt x also alle Zeilen von A ist damit zulässig für (SCP). Insbesondere gilt $c^T x \geq c^T x^\star$ und das Lemma ist bewiesen. □

Bemerkung 3.5.2. *Lemma 3.5.1 gilt sowohl für den Fall, dass eine Optimallösung von (S$\mathring{\text{C}}$Pl) zum Aufstellen von $\mathcal{M}^\circ$ betrachtet wird, als auch für den Fall, dass hierzu eine Optimallösung von (SCPl) genutzt wird.*

Prinzipiell funktioniert die beschriebene Idee, eine zulässige Lösung von (SCP) zu konstruieren, für beliebige $\underline{x} \in \mathbb{B}^n$. Denn entweder ist $\underline{x}$ bereits zulässig für (SCP), oder die Menge $\mathcal{M}^\circ = \{k \in \{1, ..., m\} : A_k \underline{x} = 0\}$ ist nicht leer. Dann kann das Problem (Red-SCP) aufgestellt und gelöst werden. Sei $\widetilde{x}$ eine Lösung

dieses Problems. Wie in Lemma 3.5.1 ist dann x, gegeben durch $x_j = \max\{\underline{x}_j, \widetilde{x}_j\}$ $(j \in \{1, ..., n\})$, zulässig für (SCP).

Zwar führt die beschriebene Idee für beliebige $\underline{x} \in \mathbb{B}^n$ zu einer zulässigen Lösung von (SCP), dennoch ist es sinnvoll, Lösungen von (SCPl) oder (S$\mathring{C}$Pl) zu nutzen, da diese als Optimallösungen einer Relaxation von (SCP) bereits „gute" Spalten im Hinblick auf das Problem (SCP) selbst enthalten.

Wir wollen nun aus den vorangegangenen Überlegungen einen Algorithmus zur Bestimmung einer oberen Schranke von (SCP) formulieren. Dazu müssen wir nach Lemma 3.5.1 eine zulässige Lösung des Problems (Red-SCP) bestimmen. Dieses Problem hat zwar weniger Zeilen als das ursprüngliche Problem (SCP), eine Optimallösung kann im Allgemeinen aber auch für (Red-SCP) nicht effizient berechnet werden. Wir werden daher eine zulässige Lösung von (Red-SCP) mit Algorithmus 3 berechnen, merken aber ausdrücklich an, dass hierzu auch jedes andere Verfahren zur Bestimmung einer zulässigen Lösung eines Mengenüberdeckungsproblems genutzt werden kann.

Wir bekommen die folgenden Algorithmen zur Berechnung einer oberen Schranke von (SCP).

Algorithmus 6 Obere Schranke für (SCP) mit (SCPl)

Input: Überdeckungsmatrix $A \in \mathbb{B}^{m \times n}$, Kosten $c \in \mathbb{R}^n_+$ von (SCP).

Output: Zulässige Lösung x von (SCP).

1: Berechne eine Optimallösung x^{C1P} von (SCPl).

2: $\mathcal{M}^{C1P} := \{k \in \{1, ..., m\} : A_k x^{C1P} = 0\}$.
If $\mathcal{M}^{C1P} = \emptyset$ **stop**: $x = x^{C1P}$ ist optimale Lösung von (SCP).

3: Stelle (Red-SCP) in Bezug auf $\mathcal{M}^{C1P}$ auf.

4: Bestimme eine zulässige Lösung $\widetilde{x}$ von (Red-SCP) mit Algorithmus 3.

5: Definiere für alle $j \in \{1, ..., n\}$: $x_j = \max\{x_j^{C1P}, \widetilde{x}_j\}$.

Output: x.

Algorithmus 7 Obere Schranke für (SCP) mit (SC̊Pl)

Input: Überdeckungsmatrix $A \in \mathbb{B}^{m \times n}$, Kosten $c \in \mathbb{R}^n_+$ von (SCP).

Output: Zulässige Lösung x von (SCP).

1: Berechne eine Optimallösung $\mathring{x}$ von (SC̊Pl).

2: $\mathcal{M}^\circ := \{k \in \{1, ..., m\} : A_k \mathring{x} = 0\}$.
If $\mathcal{M}^\circ = \emptyset$ **stop**: $x = \mathring{x}$ ist optimale Lösung von (SCP).

3: Stelle (Red-SCP) in Bezug auf $\mathcal{M}^\circ$ auf.

4: Bestimme eine zulässige Lösung $\widetilde{x}$ von (Red-SCP) mit Algorithmus 3.

5: Definiere für alle $j \in \{1, ..., n\} : x_j = \max\{\mathring{x}_j, \widetilde{x}_j\}$.

Output: x.

Bemerkung 3.5.3. *Falls die Matrix $A \in \mathbb{B}^{m \times n}$ keine Zeilen mit C1P hat, so hat das Problem (SCPl) in Schritt 1 von Algorithmus 6 die triviale Lösung $x^{C1P} = \underline{0}_n$ und in Schritt 2 des Algorithmus ist $\mathcal{M}^{C1P} = \{1, ..., m\}$. In den Schritten 4 und 5 wird dann einfach Algorithmus 3 auf die Matrix A angewendet. Algorithmus 6 liefert in diesem Fall also das gleiche Ergebnis wie Algorithmus 3. Dies kann verhindert werden, indem, falls die Matrix A keine C1P-Zeilen hat, die Spalten von A so permutiert werden, dass es zumindest eine Zeile mit C1P gibt und das Problem (SCPl) nicht-triviale Lösungen hat.*

Falls es in der Matrix $A \in \mathbb{B}^{m \times n}$ Zeilen mit C1P gibt, A selber aber keine C1P-Matrix ist, so werden in Algorithmus 6 zwei Mengenüberdeckungsprobleme mit C1P mit jeweils weniger als m Zeilen in ihrer Überdeckungsmatrix gelöst. Denn sei p $(1 \leq p < m)$ die Anzahl der C1P-Zeilen in A. Dann hat die Überdeckungsmatrix von (SCPl) in Schritt 1 die Größe $p \times n$. Für die Menge $\mathcal{M}^{C1P}$ gilt daher $|\mathcal{M}^{C1P}| \leq m - p$ und die Überdeckungsmatrix von (Red-SCP) hat die Größe $|\mathcal{M}^{C1P}| \times n$.

3.5.1 Vergleich von Algorithmus 6 und Algorithmus 7

Algorithmus 6 und Algorithmus 7 unterscheiden sich nur in dem einen Punkt, in dem eine Optimallösung von (SCPl) bzw. (SC̊Pl) berechnet wird. Wir wissen aus Abschnitt 2.3, dass die für (SC̊Pl) erzeugte untere Schranke von (SCP) mindestens so gut ist wie die untere Schranke, die aus einer Lösung von (SCPl) gewonnen wird.

Die Vermutung, dass nun Algorithmus 7 eine mindestens so gute obere Schranke wie Algorithmus 6 liefert, liegt nah. Die folgenden Beispiele zeigen, dass dies vorkommen kann, im Allgemeinen aber nicht der Fall ist. Zunächst geben wir ein Beispiel an, in dem Algorithmus 6 eine echt kleinere obere Schranke für ein Mengenüberdeckungsproblem liefert als Algorithmus 7.

Beispiel 3.5.4. *Wir betrachten das ungewichtete (SCP) mit Überdeckungsmatrix*

$$A = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 \end{pmatrix}.$$

Eine Optimallösung von (SCP) ist gegeben durch $x^\star = (0,1,0,0,0,1)$ *mit Zielfunktionswert* $z^\star = 2$.

Zunächst berechnen wir eine obere Schranke mit Algorithmus 7. Die konvexifizierte Matrix ist gegeben durch

$$\mathring{A} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 \end{pmatrix}.$$

Eine optimale Lösung von (SC̊Pl) ist z. B. gegeben durch $\mathring{x} = (0,1,0,1,0,0)$. *Dann ist* $\mathcal{M}^\circ = \{1,2\}$ *und die Überdeckungsmatrix* $B^1 \in \mathbb{B}^{|\mathcal{M}^\circ| \times n}$ *des Problems (Red-SCP)*$^\circ$ *ist*

$$B^1 = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}.$$

Mit Algorithmus 3 berechnen wir $\widetilde{x} = (1,0,0,0,0,0)$ *als zulässige Lösung des Problems (Red-SCP)*$^\circ$.

Mit $x_j^1 := \max\{\mathring{x}_j, \widetilde{x}_j\}$ $(j = 1, ..., 6)$ *ist* $x^1 = (1, 1, 0, 1, 0, 0)$ *die von Algorithmus 7 berechnete zulässige Lösung von (SCP) mit Zielfunktionswert* $z^1 = 3$.

Im Folgenden berechnen wir nun eine obere Schranke von (SCP) mit Algorithmus 6. Der C1P-Anteil von A, und damit die Überdeckungsmatrix von (SCPl), besteht nur aus der letzten Zeile. Eine Optimallösung von (SCPl) ist daher durch $x^{C1P} = (0, 1, 0, 0, 0, 0)$ *gegeben. Die nicht von* x^{C1P} *überdeckten Zeilen von A sind* $\mathcal{M}^{C1P} = \{1, 2, 5\}$. *Somit ist die Überdeckungsmatrix* $B^2 \in \mathbb{B}^{|\mathcal{M}^{C1P}| \times n}$ *des Problems (Red-SCP)*^{C1P} *gegeben durch*

$$B^2 = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 \end{pmatrix}.$$

Als zulässige Lösung des Problems (Red-SCP)^{C1P} *berechnen wir* $\widetilde{x} = (0, 0, 0, 0, 0, 1)$ *mit Algorithmus 3.*

Mit $x_j^2 := \max\{x_j^{C1P}, \widetilde{x}_j\}$ $(j = 1, ..., 6)$ *ist* $x^2 = (0, 1, 0, 0, 0, 1)$ *die von Algorithmus 6 berechnete zulässige Lösung von (SCP) mit Zielfunktionswert* $z^2 = 2$.

Da $z^2 = z^\star$ *gilt, ist* x^2 *sogar eine Optimallösung von (SCP). Insbesondere liefert die durch Algorithmus 6 berechnete zulässige Lösung* x^2 *eine echt besser obere Schranke für (SCP) als die von Algorithmus 7 berechnete Lösung* x^1.

Im folgenden Beispiel für ein Mengenüberdeckungsproblem hat die von Algorithmus 7 berechnete zulässige Lösung x^1 des Problems einen echt kleineren Zielfunktionswert als die von Algorithmus 6 bestimmte zulässige Lösung x^2.

Beispiel 3.5.5. *Wir betrachten das ungewichtete (SCP) mit Überdeckungsmatrix*

$$A = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 \end{pmatrix}.$$

Eine Optimallösung von (SCP) ist gegeben durch $x^\star = (1,0,1,0,0,0)$ *mit Zielfunktionswert* $z^\star = 2$.

Betrachten wir zunächst Algorithmus 7. Die konvexifizierte Matrix ist gegeben durch

$$\mathring{A} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 \end{pmatrix}.$$

Das Problem (SC̊Pl) wird somit z. B. gelöst durch $\mathring{x} = (0,0,1,0,0,0)$. *Die nicht durch* $\mathring{x}$ *überdeckten Zeilen von A sind* $\mathcal{M}^\circ = \{1,3\}$. *Somit ist die Überdeckungsmatrix* $B^1 \in \mathbb{B}^{|\mathcal{M}^\circ| \times n}$ *des Problems (Red-SCP)*$^\circ$ *gegeben durch*

$$B^1 = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 & 0 \end{pmatrix}.$$

Mit Algorithmus 3 berechnen wir die zulässige Lösung $\widetilde{x} = (1,0,0,0,0,0)$ *von (Red-SCP)*$^\circ$.

Für die von Algorithmus 7 berechnete zulässige Lösung von (SCP) ergibt sich $x^1 = (1,0,1,0,0,0)$ *mit Zielfunktionswert* $z^1 = 2$. x^1 *ist zugleich optimal für (SCP).*

Nun wenden wir Algorithmus 6 auf (SCP) an. Der C1P-Anteil von A und somit die Überdeckungsmatrix von (SCPl) besteht nur aus der letzten Zeile der Matrix. Eine Lösung von (SCPl) ist also gegeben durch $x^{C1P} = (0,1,0,0,0,0)$, *die Zeilen* $\mathcal{M}^{C1P} = \{1,2,5\}$ *von A werden nicht durch* x^{C1P} *überdeckt. Die Überdeckungsmatrix* $B^2 \in \mathbb{B}^{|\mathcal{M}^{C1P}| \times n}$ *des Problems (Red-SCP)*^{C1P} *ist nun gegeben durch*

$$B = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \end{pmatrix}.$$

Durch Algorithmus 3 berechnen wir die zulässige Lösung $\widetilde{x} = (1,0,1,0,0,0)$ *von (Red-SCP)*^{C1P}.

Insgesamt ergibt sich $x^2 = (1, 1, 1, 0, 0, 0)$ *als die von Algorithmus 6 berechnete zulässige Lösung von (SCP). Deren Zielfunktionswert ist* $z^2 = 3$ *und somit echt größer als der Zielfunktionswert* $z^1 = 2$ *der Lösung* x^1.

Die beiden Beispiele zeigen, dass im Allgemeinen keine Aussage darüber getroffen werden kann, ob die Wahl von (SCPl) oder (S$\mathring{\text{C}}$Pl) im ersten Schritt von Algorithmus 6 bzw. Algorithmus 7 zu einer besseren oberen Schranke von (SCP) führt. Die Lösung $\mathring{x}$ von (S$\mathring{\text{C}}$Pl) überdeckt zwar mindestens so viele Zeilen von A wie die Lösung x^{C1P} von (SCPl) (siehe Lemma 2.1.7). Aber daraus folgt nicht, dass auch eine Lösung $\widetilde{x}$ des Problems (Red-SCP) und damit insgesamt die Lösung x^1, die durch Algorithmus 7 berechnet wird, einen kleinen Zielfunktionswert hat.

Andererseits hat die Lösung x^{C1P} einen kleineren (oder den gleichen) Zielfunktionswert als die Lösung $\mathring{x}$. Falls der Zielfunktionswert der Lösung $\widetilde{x}$ von (Red-SCP) ebenfalls klein ist, so ist insgesamt auch der Zielfunktionswert der Lösung x^2, die durch Algorithmus 6 berechnet wird, klein.

Kapitel 4

Numerische Ergebnisse

Die in den Kapiteln 2 und 3 vorgestellten Methoden zur Berechnung von unteren und oberen Schranken für Mengenüberdeckungsprobleme werden wir in diesem Kapitel an zufälligen Problemen auf ihre Güte testen und miteinander vergleichen. Der Großteil der vorgestellten Verfahren wurde auf Basis eines Mengenüberdeckungsproblems mit fast C1P entwickelt. Wir werden die Methoden daher auch an Problemen testen, die eine solche Struktur aufweisen.

Zunächst gehen wir in Abschnitt 4.1 speziell auf einen Vergleich der Algorithmen 2 und 3 ein. Wir wissen aus Abschnitt 3.3.2, dass Algorithmus 3 eine mindestens so gute obere Schranke wie Algorithmus 2 liefert. Wir wollen nun untersuchen, für welche Art von Problemen eine echte Verbesserung durch Algorithmus 3 eintritt und wie groß diese ist.

In Abschnitt 4.2 wollen wir untersuchen, ob es einen Zusammenhang zwischen der Güte der unteren Schranken, die durch die Probleme (SCPl) und (SC̊Pl) erzeugt werden, und dem Anteil der Zeilen mit C1P in der Überdeckungsmatrix des Mengenüberdeckungsproblems gibt.

Abschließend wollen wir in Abschnitt 4.3 alle vorgestellten Methoden zur Berechnung von oberen und unteren Schranken für ein gewichtetes Mengenüberdeckungsproblem miteinander vergleichen.

Alle Berechnungen wurden auf einem PC mit AMD Athlon XP 2500+ Prozessor bei 1.84 GHz und 1,25 GB RAM ausgeführt.

Alle Algorithmen wurden in `MATLAB 7.5.0` unter Windows XP implementiert. Die in den Algorithmen auftretenden Mengenüberdeckungsprobleme mit C1P wurden jeweils durch das Simplex-Verfahren mit dem Matlab-Befehl `linprog` gelöst. Optimale Lösungen von (SCP) wurden mit dem Matlab-Befehl `bintprog` berechnet.

4.1 Vergleich von Algorithmus 2 und Algorithmus 3

In diesem Abschnitt wollen wir die oberen Schranken, die durch Algorithmus 2 (Greedy-Heuristik) und Algorithmus 3 berechnet werden, miteinander vergleichen. Da Algorithmus 3 die Greedy-Heuristik durch eine Blockauswahl erweitert, ist eine Blockstruktur der Überdeckungsmatrix A von (SCP) Voraussetzung dafür, dass Algorithmus 3 eine echt bessere Lösung als Algorithmus 2 liefert. Wir werden daher Testprobleme mit fast C1P, also nur einer kleinen Anzahl von Blöcken insgesamt, betrachten. Hierbei werden wir zum einen die *Länge* der Blöcke und zum anderen die *Anzahl* der Blöcke pro Zeile variieren.

Es werden zufällige (SCP) mit den folgenden Eigenschaften erzeugt:

- Kostenvektor $c \in \{1, ..., 10\}^{200}$, c_j zufällig gewählt für alle $j \in \{1, ...200\}$.
- Überdeckungsmatrix $A \in \mathbb{B}^{200\times 200}$.
- Jede Zeile von A hat eine zufällige Anzahl an Blöcken. Diese liegt zwischen 1 und einer vorgegebenen maximalen Anzahl *maxBlöcke*.
- Jeder Block hat eine zufällige Länge zwischen 1 und einer vorgegebenen maximalen Blocklänge *maxLänge*.

Nun führen wir zwei verschiedene Testreihen aus. Zum einen geben wir eine feste maximale Blockanzahl pro Zeile vor und variieren die maximale Blocklänge, zum anderen geben wir die maximale Blocklänge fest vor und variieren die maximale Anzahl von Blöcken pro Zeile. Für jede Testinstanz führen wir 300 unabhängige Tests durch.

Testreihe 1:

- maximale Anzahl von Blöcken pro Zeile: $maxBlöcke = 4$.
- maximale Blocklänge $maxLänge \in \{1, ..., 12\}$.
- für alle $maxLänge \in \{1, ..., 12\}$ werden jeweils 300 zufällige Probleme erzeugt und mit Algorithmus 2 und Algorithmus 3 gelöst. Außerdem wird jeweils eine Optimallösung bestimmt.

Testreihe 2:

- maximale Blocklänge: $maxLänge = 4$.
- maximale Anzahl von Blöcken $maxBlöcke \in \{1, ..., 12\}$.
- für alle $maxBlöcke \in \{1, ..., 12\}$ werden jeweils 300 zufällige Probleme erzeugt und mit Algorithmus 2 und Algorithmus 3 gelöst. Außerdem wird jeweils eine Optimallösung bestimmt.

In Tabelle 4.1 und Tabelle 4.2 geben wir die Ergebnisse der Testreihen 1 und 2 wieder. Dabei haben die Spalten folgende Bedeutung:

- *maxBlöcke*: maximale Anzahl der Blöcke pro Zeile.
- *maxLänge*: maximale Blocklänge.
- z_G: Zielfunktionswert von Algorithmus 2.
- $\widetilde{z}$: Zielfunktionswert von Algorithmus 3.
- $z^\star$: optimaler Zielfunktionswert.
- $\widetilde{z} < z_G[\%]$: Anteil der 300 Tests, bei denen Algorithmus 3 einen echt kleineren Zielfunktionswert als Algorithmus 2 hat, in Prozent.
- $\overline{\frac{z_G - \widetilde{z}}{z_G}}$: arithmetisches Mittel der relativen Abweichung $\frac{z_G - \widetilde{z}}{z_G}$ von Algorithmus 2 und Algorithmus 3 bei 300 Tests.
- $\max \frac{z_G - \widetilde{z}}{z_G}$: maximale relative Abweichung von Algorithmus 2 und Algorithmus 3 bei den 300 Tests.

- $\overline{\frac{\widetilde{z}-z^\star}{z^\star}}$: arithmetisches Mittel des relativen Fehlers $\frac{\widetilde{z}-z^\star}{z^\star}$ von Algorithmus 3 bei 300 Tests.
- max $\frac{\widetilde{z}-z^\star}{z^\star}$: maximaler relativer Fehler von Algorithmus 3 bei den 300 Tests.
- *Zeit Algo3[s]*: Durchschnittlicher Zeitaufwand von Algorithmus 3 pro Test in Sekunden bei den 300 Tests.
- *Zeit Opt [s]*: Durchschnittlicher Zeitaufwand zur Berechnung einer Optimallösung pro Test in Sekunden bei den 300 Tests.

In den Abbildungen 4.1 und 4.2 tragen wir $\widetilde{z} < z_G[\%]$ auf der linken y-Achse (rot) und $\overline{\frac{z_G-\widetilde{z}}{z_G}}$ auf der rechten y-Achse (grün) auf. In diesen Abbildungen wollen wir das Verhalten dieser beiden Größen abhängig von der maximalen Länge der Blöcke bzw. von der maximalen Anzahl von Blöcken pro Zeile verdeutlichen.

Tabelle 4.1 – Testreihe 1: *maxBlöcke* $= 4$, *maxLänge* $\in \{1, ..., 12\}$, jeweils 300 Tests.

maxLänge	$\widetilde{z} < z_G[\%]$	$\overline{\frac{z_G-\widetilde{z}}{z_G}}$	max $\frac{z_G-\widetilde{z}}{z_G}$	$\overline{\frac{\widetilde{z}-z^\star}{z^\star}}$	max $\frac{\widetilde{z}-z^\star}{z^\star}$	*Zeit Algo3 [s]*	*Zeit Opt [s]*
1	05,33	0,01	0,01	0,06	0,14	0,17	0,58
2	86,00	0,01	0,06	0,07	0,15	0,22	0,64
3	96,00	0,03	0,08	0,07	0,16	0,27	0,61
4	96,00	0,03	0,11	0,07	0,16	0,30	0,61
5	98,33	0,04	0,12	0,07	0,16	0,35	0,65
6	98,67	0,05	0,11	0,07	0,20	0,39	0,65
7	99,00	0,05	0,13	0,07	0,19	0,37	0,67
8	98,67	0,05	0,16	0,07	0,18	0,36	0,55
9	98,67	0,06	0,14	0,07	0,19	0,35	0,55
10	99,33	0,06	0,15	0,07	0,16	0,37	0,57
11	99,67	0,06	0,17	0,07	0,22	0,34	0,53
12	98,33	0,07	0,16	0,06	0,20	0,34	0,51

Beobachtung 4.1.1. *In den Abbildungen 4.1 und 4.2 ist auf der linken y-Achse (rot) der Prozentsatz der 300 Tests aufgetragen, bei denen Algorithmus 3 (Lösung $\widetilde{z}$) eine echt bessere untere Schranke für (SCP) als Algorithmus 2 (Lösung z_G) liefert. Der Zielfunktionswert von $\widetilde{z}$ ist dabei nicht nur ε-besser als z_G, denn die Kosten sind ganzzahlig, und somit liegt die Verbesserung bei mindestens* 1. *Auf der*

Tabelle 4.2 – Testreihe 2: *maxLänge* = 4, *maxBlöcke* $\in \{1, ..., 12\}$, jeweils 300 Tests.

maxBlöcke	$\widetilde{z} < z_G[\%]$	$\overline{\frac{z_G - \widetilde{z}}{z_G}}$	$\max \frac{z_G - \widetilde{z}}{z_G}$	$\overline{\frac{\widetilde{z} - z^\star}{z^\star}}$	$\max \frac{\widetilde{z} - z^\star}{z^\star}$	*Zeit Algo3 [s]*	*Zeit Opt [s]*
1	100,00	0,07	0,13	0,00	0,00	0,32	0,52
2	100,00	0,05	0,09	0,04	0,11	0,31	0,53
3	99,00	0,04	0,10	0,06	0,15	0,30	0,55
4	99,00	0,03	0,09	0,10	0,19	0,29	0,61
5	94,67	0,03	0,08	0,09	0,19	0,30	0,65
6	91,33	0,03	0,10	0,09	0,20	0,29	0,74
7	85,67	0,02	0,10	0,10	0,25	0,29	0,88
8	81,00	0,02	0,09	0,10	0,20	0,31	0,90
9	81,66	0,02	0,10	0,11	0,23	0,29	0,94
10	72,00	0,02	0,08	0,12	0,23	0,28	1,06
11	73,00	0,02	0,10	0,12	0,30	0,24	0,98
12	71,33	0,02	0,10	0,12	0,28	0,24	0,87

rechten y-Achse (grün) ist die durchschnittliche relative Verbesserung $\overline{\frac{z_G - \widetilde{z}}{z_G}}$ *von Algorithmus 3 gegenüber Algorithmus 2 bei 300 Tests aufgetragen.*

Testreihe 1: *In Abbildung 4.1 lässt sich erkennen, dass sowohl der Anteil der Tests mit* $\widetilde{z} < z_G$*, als auch die durchschnittliche Verbesserung* $\overline{\frac{z_G - \widetilde{z}}{z_G}}$ *abhängig von der maximalen Länge der Blöcke der Überdeckungsmatrix A ist. Je längere Blöcke auftreten, desto größer sind* $\overline{\frac{z_G - \widetilde{z}}{z_G}}$ *und der Anteil der Probleme mit* $\widetilde{z} < z_G$*. Der Vorteil von Algorithmus 3 gegenüber Algorithmus 2 nimmt also mit der Länge der Blöcke zu. In Tabelle 4.2 lässt sich außerdem ablesen, dass der Wert von* $\overline{\frac{\widetilde{z} - z^\star}{z^\star}}$ *für alle Blocklängen nahezu konstant ist. Das heißt, Algorithmus 3 hat im Allgemeinen einen von der maximalen Blocklänge unabhängigen relativen Fehler gegenüber der Optimallösung. Die mit der maximalen Blocklänge ansteigende relative Verbesserung* $\overline{\frac{z_G - \widetilde{z}}{z_G}}$ *ist darauf zurückzuführen, dass Algorithmus 2 mit größerer maximaler Blocklänge einen ansteigenden relativen Fehler hat, der relative Fehler von Algorithmus 3 aber annähernd konstant bleibt.*

Weiter lässt sich in Tabelle 4.1 ablesen, dass auch die maximale relative Verbesserung $\max \frac{z_G - \widetilde{z}}{z_G}$ *der 300 Tests mit der maximalen Blocklänge zunimmt. Des Weiteren ist die durchschnittliche Laufzeit von Algorithmus 3 unabhängig von der maximalen Blocklänge. Sie liegt unter der Zeit, die für die Berechnung einer Optimallösung benötigt wird.*

Abbildung 4.1 – Vergleich von Algorithmus 2 und Algorithmus 3 für (SCP) mit $A \in \mathbb{B}^{200\times 200}$, $c \in 1, ..., 10^n$. *maxBlöcke* $= 4$, *maxLänge* $\in \{1, ..., 12\}$. Jeweils 300 Tests.

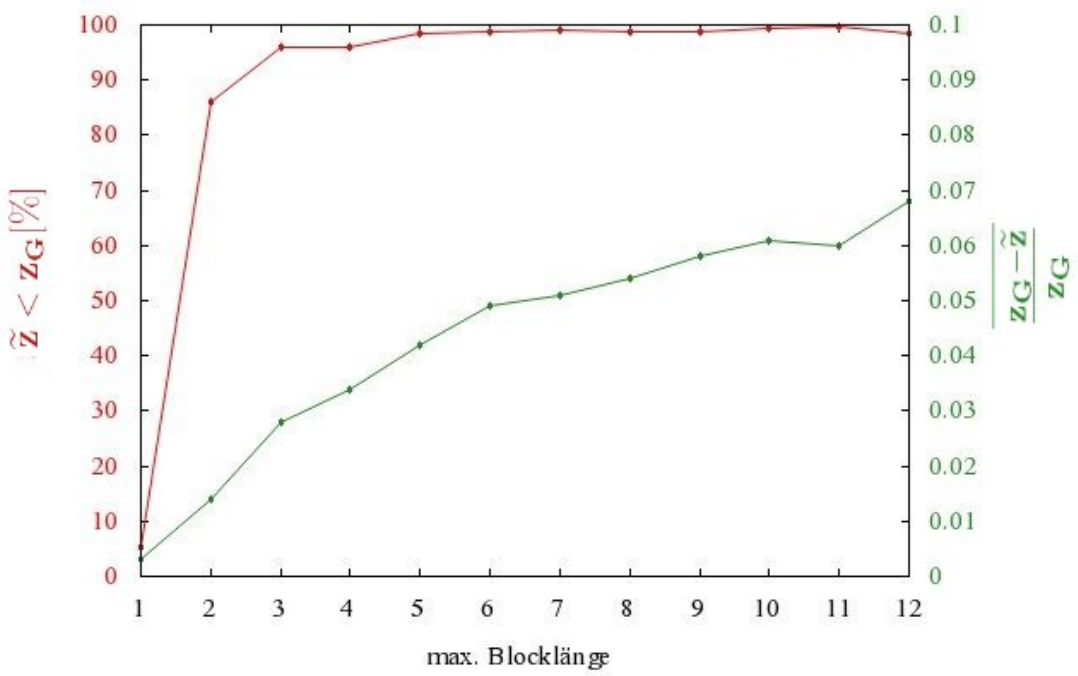
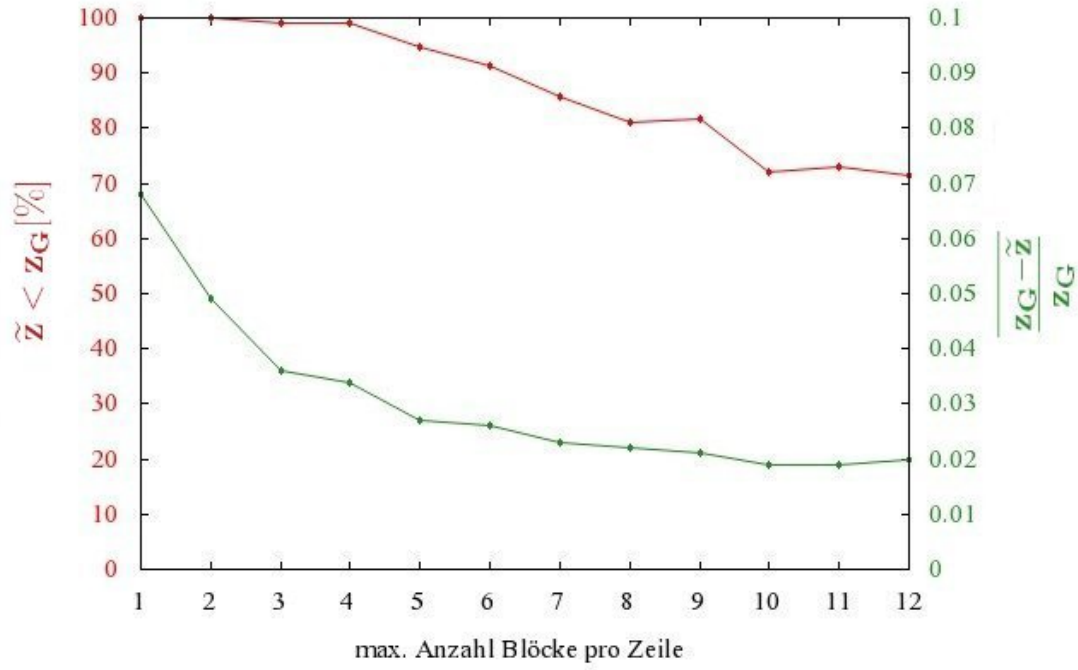


Abbildung 4.2 – Vergleich von Algorithmus 2 und Algorithmus 3 für (SCP) mit $A \in \mathbb{B}^{200\times 200}$, $c \in 1, ..., 10^n$. *maxLänge* $= 4$, *maxBlöcke* $\in \{1, ..., 12\}$. Jeweils 300 Tests.



Testreihe 2: *In Abbildung 4.2 lässt sich erkennen, dass sowohl der Anteil der Tests mit* $\widetilde{z} < z_G$*, als auch die durchschnittliche relative Verbesserung* $\overline{\frac{z_G - \widetilde{z}}{z_G}}$ *abhängig von der maximalen Anzahl der Blöcke pro Zeile der Überdeckungsmatrix A ist. Je mehr Blöcke pro Zeile auftreten, desto kleiner sind der Wert von* $\overline{\frac{z_G - \widetilde{z}}{z_G}}$ *und der Anteil der Probleme mit* $\widetilde{z} < z_G$*. Der Vorteil von Algorithmus 3 gegenüber Algorithmus 2 nimmt also mit der Anzahl der Blöcke pro Zeile ab. In Tabelle 4.2 lässt sich ablesen, dass der durchschnittliche relative Fehler* $\overline{\frac{\widetilde{z} - z^\star}{z^\star}}$ *von Algorithmus 3 mit der maximalen Anzahl der Blöcke pro Zeile zunimmt.*

Die mit der Größe von maxBlöcke *kleiner werdende durchschnittliche Verbesserung* $\overline{\frac{z_G - \widetilde{z}}{z_G}}$ *ist darauf zurückzuführen, dass der relative Fehler von Algorithmus 3 mit der maximalen Anzahl von Blöcken pro Zeile ansteigt, die Güte von Algorithmus 2 dagegen weniger abhängig von* maxBlöcke *ist.*

In Tabelle 4.2 lässt sich außerdem erkennen, dass $\max \frac{z_G - \widetilde{z}}{z_G}$ *nahezu unabhängig von der maximalen Anzahl von Blöcken pro Zeile ist. Der Wert von* $\max \frac{\widetilde{z} - z^\star}{z^\star}$ *nimmt dagegen stark mit* maxBlöcke *zu. Des Weiteren ist die durchschnittliche Laufzeit von Algorithmus 3 unabhängig von der maximalen Anzahl von Blöcken pro Zeile, die Laufzeit zur Berechnung einer optimalen Lösung steigt dagegen mit* maxBlöcke *an.*

4.1.1 Vergleich von Algorithmus 2 und Algorithmus 4 für das ungewichtete (SCP)

Wir wollen Algorithmus 2 und Algorithmus 4 (Variante von Algorithmus 3 für das ungewichtete (SCP)) für ungewichtete Mengenüberdeckungsprobleme mit fast C1P vergleichen. Hierzu konstruieren wir wie in Abschnitt 4.1 zufällige Matrizen mit fester maximaler Anzahl von Blöcken pro Zeile (*maxBlöcke*) und fester maximaler Blocklänge (*maxLänge*). Wir führen für jede Eingabeinstanz 300 unabhängige Tests durch. In Tabelle 4.3 sind die Ergebnisse für verschiedene Matrixgrößen abhängig von *maxBlöcke* und *maxLänge* eingetragen. $\widetilde{z} < z_G[\%]$, $\overline{\frac{z_G - \widetilde{z}}{z_G}}$, $\max \frac{z_G - \widetilde{z}}{z_G}$, $\overline{\frac{\widetilde{z} - z^\star}{z^\star}}$, $\max \frac{\widetilde{z} - z^\star}{z^\star}$, *Zeit Algo3 [s]* und *Zeit Opt [s]* haben dabei die gleiche Bedeutung wie in Abschnitt 4.1.

Tabelle 4.3 – Vergleich von Algorithmus 4 und Algorithmus 2 für ungewichtete (SCP). Jeweils 300 Tests.

Matrixgröße	*maxBlöcke*	*maxLänge*	$\widetilde{z} < z_G[\%]$	$\overline{\frac{z_G-\widetilde{z}}{z_G}}$	max $\frac{z_G-\widetilde{z}}{z_G}$	$\overline{\frac{\widetilde{z}-z^\star}{z^\star}}$	max $\frac{\widetilde{z}-z^\star}{z^\star}$	*Zeit Algo4 [s]*	*Zeit Opt [s]*
100×100	4	4	9,00	0,01	0,05	0,09	0,23	0,11	0,71
100×100	4	6	15,00	0,01	0,10	0,11	0,25	0,12	0,89
100×100	4	8	25,00	0,01	0,11	0,10	0,26	0,14	0,67
100×100	6	4	5,33	0,01	0,06	0,12	0,24	0,10	2,53
100×100	8	4	3,33	0,01	0,06	0,13	0,29	0,10	4,47
200×200	2	4	51,00	0,01	0,05	0,05	0,13	0,36	0,77
200×200	4	4	19,00	0,01	0,06	0,10	0,20	0,24	5,70
200×200	4	6	26,00	0,01	0,07	0,11	0,21	0,29	7,35
200×200	4	8	36,00	0,01	0,09	0,11	0,30	0,29	3,73
200×200	6	4	13,66	0,01	0,05	0,13	0,23	0,25	81,06

Beobachtung 4.1.2. *In Tabelle 4.3 können wir erkennen, dass der Anteil der Probleme, bei denen Algorithmus 4 eine echte Verbesserung gegenüber Algorithmus 2 darstellt, bei ungewichteten Mengenüberdeckungsproblemen viel kleiner als bei gewichteten (SCP) ist. Dementsprechend liegt auch die durchschnittliche relative Verbesserung* $\overline{\frac{z_G - \widetilde{z}}{z_G}}$ *weit unter den Ergebnissen aus Abschnitt 4.1. Wie im Fall gewichteter Mengenüberdeckungsprobleme ist auch für das ungewichtete (SCP) zu erkennen, dass eine kleine Anzahl von Blöcken je Zeile der Überdeckungsmatrix und eine große maximale Länge der Blöcke für den Vorteil von Algorithmus 4 gegenüber der Greedy-Heuristik sorgen.*

Die kleinere durchschnittliche Verbesserung $\overline{\frac{z_G - \widetilde{z}}{z_G}}$ *der Greedy-Heuristik durch Algorithmus 4 für ungewichtete (SCP) gegenüber den Ergebnissen aus Abschnitt 4.1 für gewichtete Mengenüberdeckungsprobleme ist auf den kleineren durchschnittlichen Fehler der Greedy-Heuristik bei ungewichteten Mengenüberdeckungsproblemen zurückzuführen. Denn mit einem kleineren relativen Fehler der Greedy-Lösung ist auch die Chance für eine echte Verbesserung der Greedy-Lösung durch Algorithmus 4 geringer.*

Die Laufzeit von Algorithmus 4 ist durchschnittlich klein gegenüber der Laufzeit für die Berechnung einer Optimallösung. Dies rechtfertigt eine Nutzung von Algorithmus 4 als Erweiterung der Greedy-Heuristik auch für den Fall eines ungewichteten (SCP), obwohl eine Verbesserung der Greedy-Lösung durch Algorithmus 4 im Allgemeinen nicht zu erwarten ist.

4.2 Untersuchung von (SCPl) und (SC̊Pl)

In diesem Abschnitt wollen wir den relativen Fehler der unteren Schranken von (SCP), die durch (SCPl) und (SC̊Pl) erzeugt werden, abhängig vom Anteil der Zeilen mit C1P in der Überdeckungsmatrix des (SCP) untersuchen.

Hierzu konstruieren wir Mengenüberdeckungsprobleme mit fast C1P, dessen Überdeckungsmatrizen einen festgelegten Anteil an Zeilen mit C1P haben.

Konkret erzeugen wir zufällige (SCP) mit den folgenden Eigenschaften:

- Kostenvektor $c \in \{1, ..., 10\}^{200}$, c_j zufällig gewählt für alle $j \in \{1, ...200\}$.
- Überdeckungsmatrix $A \in \mathbb{B}^{200 \times 200}$.

- Die Matrix A hat eine vorgegebene Anzahl b von Zeilen, die genau einen Block aufeinanderfolgender Einsen enthalten.
- Die übrigen $200 - b$ Zeilen von A haben eine zufällige Anzahl an Blöcken. Diese liegt zwischen 2 und 5.
- Jeder Block hat eine zufällige Länge zwischen 1 und 5.

Wir variieren nun die Anzahl der Zeilen b mit genau einem Block. Wir untersuchen $b \in \{10, 20, 30, 40, 50, 60, 70, 80, 90, 100\}$, d. h. der Anteil der Zeilen mit C1P in A liegt zwischen 5 % und 50 %. Für jedes b führen wir 300 unabhängige Tests durch. Für jeden Test berechnen wir die Optimallösung des (SCP) und die durch (SCPl) und (SC̊Pl) erzeugten unteren Schranken.

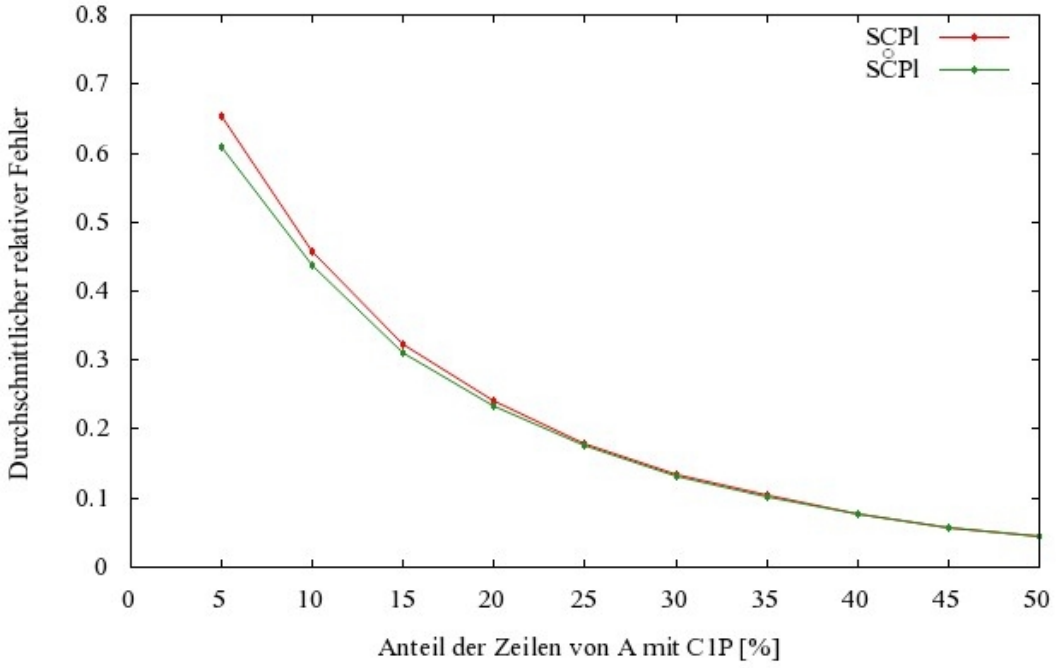
In Abbildung 4.3 tragen wir den durchschnittlichen relativen Fehler der beiden durch (SCPl) (rot) und (SC̊Pl) (grün) erzeugten unteren Schranken abhängig vom Anteil der Zeilen mit C1P (in Prozent) in der Überdeckungsmatrix der Mengenüberdeckungsprobleme auf.

Beobachtung 4.2.1. *In Abbildung 4.3 lässt sich ein Zusammenhang des relativen Fehlers der durch (SCPl) und (SC̊Pl) erzeugten unteren Schranken von (SCP) mit dem Anteil der Zeilen mit C1P in der Überdeckungsmatrix des (SCP) erkennen. Bei einem kleinen Anteil von 5 % von Zeilen mit C1P in der Matrix liegt der durchschnittliche relative Fehler der beiden Schranken bei ca.* 0.6*. Mit zunehmendem Anteil von C1P-Zeilen in der Überdeckungsmatrix wird auch der durchschnittliche relative Fehler der beiden Schranken kleiner, bei einem Anteil von über 40 % liegt er unter* 0.1*.*

Außerdem lässt sich erkennen, dass die relativen Fehler der von (SCPl) und (SC̊Pl) erzeugten unteren Schranken nah beieinander liegen, wobei letzterer etwas kleiner ist als ersterer.

Insgesamt kann beobachtet werden, dass die Lösungen von (SCPl) und (SC̊Pl) umso bessere Schranken für (SCP) liefern, je größer der Anteil der Zeilen mit C1P in der Überdeckungsmatrix des Problems ist.

Abbildung 4.3 – Relativer Fehler von (SCPl) und (S$\mathring{C}$Pl) abhängig von der Anzahl der Zeilen mit C1P.



4.3 Vergleich verschiedener oberer und unterer Schranken für gewichtete (SCP)

Abschließend wollen wir verschiedene der angegebenen Methoden zur Berechnung von oberen und unteren Schranken eines gewichteten Mengenüberdeckungsproblems miteinander vergleichen. Hierbei wollen wir zum einen den Approximationsfaktor der verschiedenen Schranken studieren und zum anderen die Laufzeit der Verfahren zur Berechnung der Schranken untersuchen.

Da wir den Großteil der Schranken basierend auf Mengenüberdeckungsproblemen mit fast C1P entwickelt haben, beschränken wir uns bei den Testinstanzen auf eben solche Probleme, bei denen die Überdeckungsmatrix fast C1P hat.

Die folgenden Schranken werden für alle Testinstanzen berechnet (in Klammern ist jeweils der Abschnitt angegeben, in dem die Schranke vorgestellt wurde):

Untere Schranken: *Duale Schranke* (2.2), *(SCPl)* (2.1.1), (*S$\mathring{C}$Pl*) (2.1.2).

Obere Schranken: *Greedy-Heuristik* (3.3.1), *Greedy-Blockauswahl* (3.3.2), *Algorithmus 6* (3.5), *Algorithmus 7* (3.5).

Wir testen die Schranken an zufälligen Matrizen mit einer festen Blockstruktur (wie in Abschnitt 4.1), d.h. jede Zeile der erzeugten Matrizen hat eine zufällige, aber beschränkte maximale Anzahl an Blöcken. Die Blöcke wiederum haben jeweils eine zufällige, aber beschränkte maximale Länge.

Die Schranken werden für quadratische Matrizen mit variabler Zeilen- und Spaltenanzahl (100, 200, 300, 400, 500, 600, 700, 800) getestet. Die maximale Anzahl von Blöcken jeder Zeile und die maximale Länge der Blöcke ist in Tabelle 4.4 angegeben und jeweils so gewählt, dass der erwartete Anteil von Nicht-Null-Elementen der Matrix bei ca. 6 % liegt.

Der Kostenvektor c für die erzeugten Mengenüberdeckungsprobleme besteht aus zufälligen ganzen Zahlen zwischen 1 und 10.

Tabelle 4.4 – Eigenschaften der erzeugten Matrizen

Matrixgröße	Anz. Blöcke pro Zeile	Länge je Block	Erwartungswert Nicht-Null-Elemente
100×100	1-4	1-4	6,25 %
200×200	1-6	1-6	6,13 %
300×300	1-8	1-8	6,75 %
400×400	1-9	1-9	6,25 %
500×500	1-10	1-10	6,05 %
600×600	1-11	1-11	6,00 %
700×700	1-12	1-12	6,04 %
800×800	1-13	1-13	6,13 %

Für jede Matrixgröße werden 100 unabhängige Matrizen und Kostenvektoren erzeugt und jeweils alle angegebenen Schranken berechnet. In Abbildung 4.4 ist der durchschnittliche Quotient der Schranke und des optimalen Zielfunktionswertes, also der durchschnittliche Approximationsfaktor der Schranke bei 100 Tests, für alle Schranken abhängig von der Matrixgröße aufgetragen.

In Abbildung 4.5 ist für jede Matrixgröße die durchschnittliche Laufzeit bei 100 Tests für die Berechnung jeder Schranke aufgetragen. Zum Vergleich ist hier noch die Laufzeit für die Berechnung der Optimallösung angegeben.

Abbildung 4.4 – Vergleich der Güte verschiedener Schranken für Matrizen mit Blockstruktur verschiedener Größe.

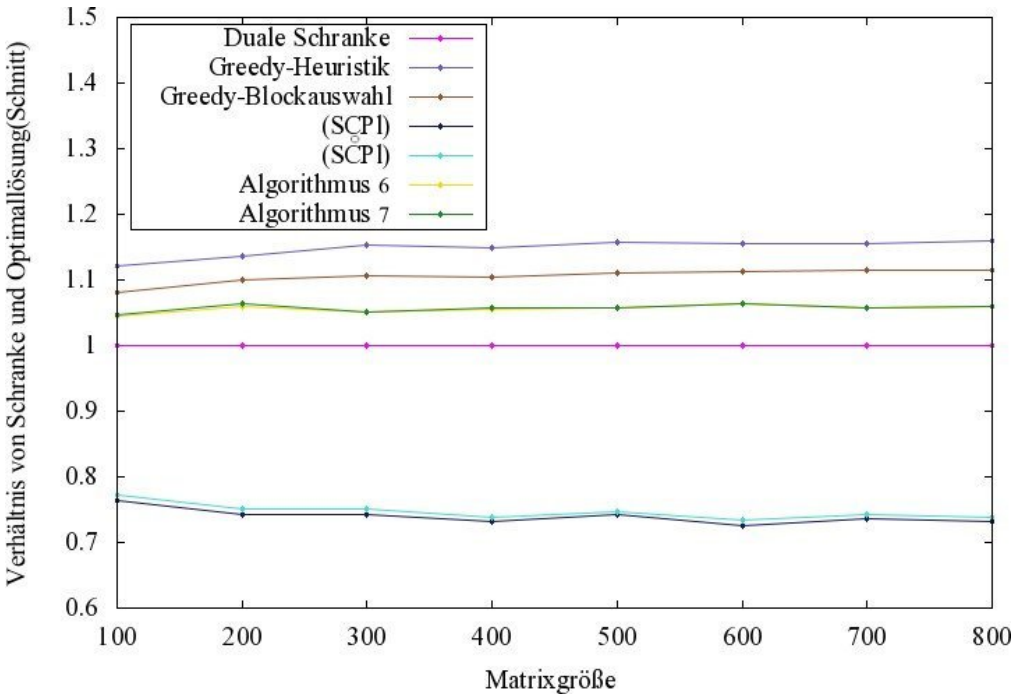
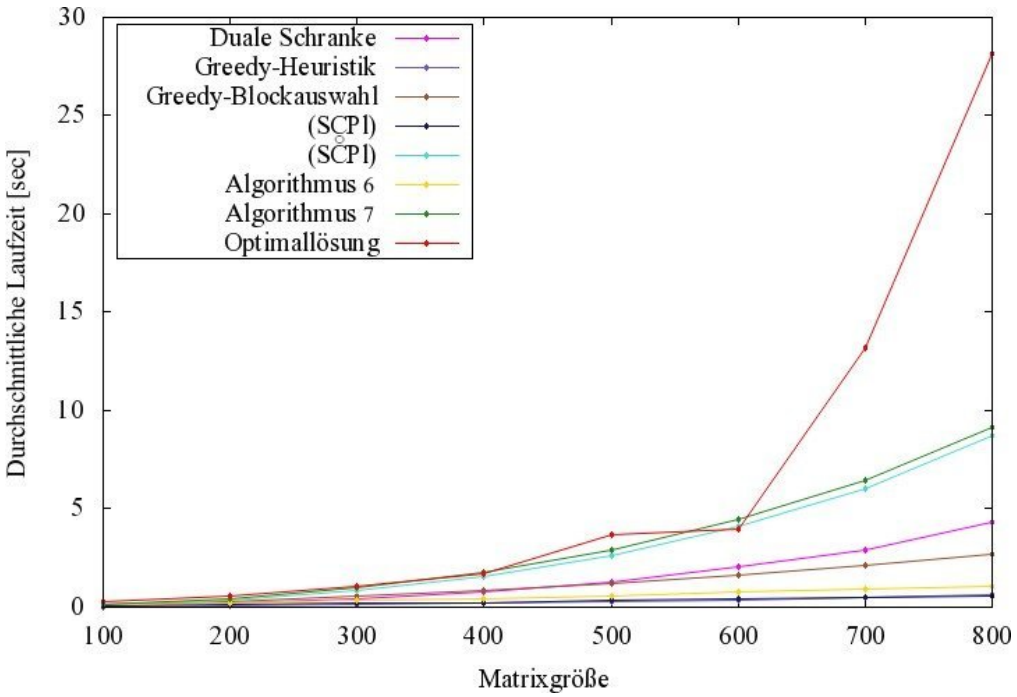


Abbildung 4.5 – Laufzeit zur Berechnung der Schranken.



Beobachtung 4.3.1. *Bei Problemen mit der angegebenen festen Blockstruktur zeigt Abbildung 4.4, dass von den getesteten Schranken die duale Schranke den durchschnittlich am nächsten bei Eins liegenden Approximationsfaktor hat. Die duale Schranke gibt also einen sehr guten Näherungswert für den optimalen Zielfunktionswert und wie Abbildung 4.5 zeigt, liegt die durchschnittliche Laufzeit zur Berechnung der dualen Schranke deutlich unter der durchschnittlichen Laufzeit zur Berechnung der Optimallösung.*

Die beiden Probleme (SCPl) und ($S\mathring{C}Pl$) erzeugen durchschnittlich deutlich schlechtere untere Schranken für (SCP) als die duale Schranke. Dabei kann (SCPl) sehr schnell gelöst werden, für ($S\mathring{C}Pl$) ist allerdings ein hoher Aufwand nötig, der durch die Ergebnisse nicht gerechtfertigt wird. Allerdings ist hierbei zu beachten, dass das C1P-Problem ($S\mathring{C}Pl$) nach Bemerkung 1.3.4 mit einem in [17] angegebenen Algorithmus deutlich schneller gelöst werden könnte, als mit dem hier genutzten Simplex-Verfahren. Der Grund für die stark unterschiedliche Laufzeit des Simplex-Verfahrens zur Lösung von ($S\mathring{C}Pl$) und (SCPl) liegt darin, dass die Überdeckungsmatrix des Problems (SCPl) dünn besetzt ist, dies bei dem Problem (SCPl) im Allgemeinen aber nicht der Fall ist.

Für die Güte der oberen Schranken ist zu beobachten, dass Algorithmus 6 und Algorithmus 7 durchschnittlich die besten Ergebnisse liefern. Die Approximationsfaktoren der beiden Algorithmen liegen dabei sehr nah beieinander, was dadurch zu erklären ist, dass sie sich nur geringfügig voneinander unterscheiden. Abbildung 4.5 zeigt aber, dass Algorithmus 7 deutlich langsamer läuft als Algorithmus 6. Dies ist dadurch zu erklären, dass Algorithmus 7 auf der Lösung des Problems ($S\mathring{C}Pl$) basiert. Dieses könnte mit dem in [17] vorgestellten Algorithmus erheblich schneller gelöst werden als mit dem von uns genutzten Simplex-Verfahren.

Die Berechnung von oberen Schranken durch die Greedy-Heuristik und Greedy-Blockauswahl (Algorithmus 3) liefert für die erzeugten Matrizen, unabhängig von der Problemgröße, durchschnittlich etwas schlechtere Ergebnisse als Algorithmus 6 bzw. Algorithmus 7. Dabei ist aber zu beachten, dass Algorithmus 3 von Algorithmus 6 und Algorithmus 7 implizit genutzt wird, also seinen Beitrag zu der Güte dieser Algorithmen leistet. Die durchschnittliche Laufzeit für die Greedy-Heuristik und Algorithmus 3 ist dabei auch für größere Probleme klein im Vergleich zur Laufzeit der Berechnung einer Optimallösung.

Kapitel 5

Fazit

5.1 Untere Schranken

Die drei vorgestellten Möglichkeiten, untere Schranken für (SCP) zu berechnen, haben nach den Aussagen von Lemma 2.3.1 und Lemma 2.1.7 eine klare Gütehierarchie. Die duale Schranke ist nie schlechter als die Schranke, die durch ($\mathring{\text{SC}}$Pl) berechnet wird. Diese wiederum ist mindestens so gut wie die Schranke, die wir durch Lösen des Problems (SCPl) bekommen.

5.1.1 Duale Schranke

Für die Berechnung von unteren Schranken für (SCP) zeigt sich, dass die duale Schranke die besten Ergebnisse unter den drei angegebenen Schranken erzielt. Zum einen ist die duale Schranke nie schlechter als die unteren Schranken, die durch (SCPl) und ($\mathring{\text{SC}}$Pl) berechnet werden (Lemma 2.3.1, Lemma 2.1.7). Zum anderen zeigen die Ergebnisse aus Kapitel 4, dass die duale Schranke insbesondere für Mengenüberdeckungsprobleme mit fast C1P erheblich näher am optimalen Zielfunktionswert liegt, als die beiden anderen unteren Schranken. Außerdem lässt sich die duale Schranke in der Praxis auch im Vergleich mit den anderen Schranken relativ schnell berechnen. Die duale Schranke gibt keine Hinweise auf zu wählende Spalten der Überdeckungsmatrix von (SCP). Daher kann sie nicht verwendet werden, um mit den Ideen aus Abschnitt 3.5 für einen Algorithmus zur Berechnung einer oberen Schranke von (SCP) genutzt zu werden.

5.1.2 (SCPl)

Durch Lösen von (SCPl) kann sehr schnell eine untere Schranke für (SCP) berechnet werden. Diese ist nach theoretischen Aspekten (Lemma 2.1.7) allerdings die schlechteste der drei vorgestellten unteren Schranken. Je größer der Anteil an Zeilen mit C1P in der Überdeckungsmatrix A ist, umso brauchbarer wird die Schranke (siehe Abschnitt 4.2). Für Matrizen, die keine C1P-Zeile haben, müssen die Spalten der Matrix zunächst mit zusätzlichem Aufwand permutiert werden, damit Zeilen mit C1P auftreten, sonst liefert (SCPl) nur die triviale Schranke $z_l = 0$. Da in einer Lösung von (SCPl) bestimmte Spalten der Überdeckungsmatrix gewählt werden, kann diese Lösung in Algorithmus 6 zur Berechnung einer oberen Schranke von (SCP) weiterverwendet werden.

5.1.3 (S$\mathring{\text{C}}$Pl)

Die Lösung von (S$\mathring{\text{C}}$Pl) liefert eine untere Schranke für (SCP), die mindestens genau so gut ist, wie die Schranke, die durch Lösen von (SCPl) berechnet wird (Lemma 2.1.7). Die Ergebnisse aus Abschnitt 4.3 zeigen, dass der Approximationsfaktor von (S$\mathring{\text{C}}$Pl) für Mengenüberdeckungsprobleme mit fast C1P kaum besser ist als der Approximationsfaktor von (SCPl). Der Aufwand um (S$\mathring{\text{C}}$Pl) zu lösen ist dabei aber erheblich größer. Allerdings ist zu beachten, dass wir alle C1P - Mengenüberdeckungsprobleme mit dem Simplex-Verfahren gelöst haben. Mit dem in Bemerkung 1.3.4 beschriebenen Verfahren könnte (S$\mathring{\text{C}}$Pl) erheblich schneller gelöst werden.

Im Gegensatz zu (SCPl) kann (S$\mathring{\text{C}}$Pl) für beliebige Mengenüberdeckungsprobleme aufgestellt und gelöst werden. Das heißt, es sind keine C1P-Zeilen in der Überdeckungsmatrix des Problems notwendig.

Wie bei dem Problem (SCPl) kann auch eine Lösung von (S$\mathring{\text{C}}$Pl) zur Berechnung einer oberen Schranke von (SCP) mit Algorithmus 7 weiterverwendet werden.

5.2 Obere Schranken

Für die Berechnung von oberen Schranken für beliebige Mengenüberdeckungsprobleme haben wir fünf Verfahren vorgestellt. Dabei ist für ein gegebenes (SCP)

nicht ohne weiteres zu sagen, welches Verfahren zu der besten oberen Schranke führt. Denn für die Algorithmen 3, 5, 6 und 7 kann keine Aussage darüber getroffen werden, welcher der Algorithmen das beste Ergebnis für ein Mengenüberdeckungsproblem liefert. Allerdings zeigen die numerischen Ergebnisse aus Abschnitt 4.3, dass die Algorithmen 6 und 7 für Mengenüberdeckungsprobleme, bei denen die Überdeckungsmatrix fast C1P hat, im Allgemeinen bessere Ergebnisse erzielen als Algorithmus 3. Dabei ist allerdings zu beachten, dass die Algorithmen 6 und 7 implizit Algorithmus 3 nutzen.

Die wichtige Aussage, dass Algorithmus 3 in jedem Fall eine mindestens so gute obere Schranke für (SCP) wie der Greedy-Algorithmus (Algorithmus 2) liefert, konnte in Abschnitt 3.3.3 bewiesen werden.

Des Weiteren wurde mit Algorithmus 5 ein Verfahren vorgestellt, mit dem eine beliebige zulässige Lösung von (SCP) verbessert werden kann.

Für ungewichtete (SCP) haben wir zusätzlich noch Algorithmus 1 entwickelt. Dieser hat im Gegensatz zu Algorithmus 4 eine schlechtere Gütegarantie, kann in der Praxis aber durchaus ein besseres Ergebnis liefern.

5.2.1 Greedy-Heuristik

Die wohlbekannte Greedy-Heuristik (Algorithmus 2) liefert mit einer kurzen Laufzeit eine zulässige Lösung für ein Mengenüberdeckungsproblem (SCP). Diese Lösung hat eine bestmögliche Gütegarantie für polynomielle Näherungsverfahren von Mengenüberdeckungsproblemen, falls $P \neq NP$ (Bemerkung 3.3.7). Die Greedy-Heuristik stellt daher ein sinnvolles Verfahren zur Berechnung oberer Schranken für Mengenüberdeckungsprobleme dar.

5.2.2 Algorithmus 3 und Algorithmus 4

Der in Abschnitt 3.3 vorgestellte Algorithmus 3 (Greedy-Blockauswahl) ist eine Erweiterung der Greedy-Heuristik. Die obere Schranke, die durch diesen Algorithmus berechnet wird, ist dabei immer mindestens so gut wie die obere Schranke, die von der Greedy-Heuristik geliefert wird. In Algorithmus 3 muss zusätzlich zur Anwendung der Greedy-Heuristik ein Mengenüberdeckungsproblem mit C1P gelöst werden. Dies kann nach Bemerkung 1.3.4 mit einer linearen Laufzeit erledigt

werden. Insgesamt ist der zusätzliche Aufwand von Algorithmus 3 gegenüber Algorithmus 2 also nur minimal.

Diese Eigenschaften von Algorithmus 3 verdeutlichen, dass die präsentierte Erweiterung der Greedy-Heuristik überaus sinnvoll ist. Die Ergebnisse aus Abschnitt 4.1 zeigen, dass der Vorteil von Algorithmus 3 gegenüber Algorithmus 2 vor allem bei Mengenüberdeckungsproblemen mit fast C1P auftritt. Hierbei wiederum ist es von Vorteil für Algorithmus 3, wenn die Überdeckungsmatrix wenige Blöcke pro Zeile hat, und diese Blöcke möglichst lang sind.

Für das ungewichtete (SCP) haben wir Algorithmus 4 als Variante von Algorithmus 3 vorgestellt. Der Vorteil von Algorithmus 4 gegenüber der Greedy-Heuristik ist für ungewichtete Mengenüberdeckungsprobleme kleiner und tritt seltener auf. Da Algorithmus 4 aber ebenfalls nur eine wenig längere Laufzeit als Algorithmus 2 hat, ist diese Erweiterung der Greedy-Heuristik auch für ungewichtete Mengenüberdeckungsprobleme sinnvoll.

5.2.3 Algorithmus 5

Algorithmus 5 ist eine Verallgemeinerung von Algorithmus 3, in dem - statt aus einer Lösung der Greedy-Heuristik - aus einer beliebigen zulässigen Lösung x von (SCP) eine C1P-Matrix aufgestellt wird.

Die wichtigste Aussage für diesen Algorithmus ist, dass seine Lösung $\widetilde{x}$ einen höchstens so großen Zielfunktionswert wie die Eingabe x hat. Somit kann jede zulässige Lösung x von (SCP) durch Algorithmus 5 zu $\widetilde{x}$ verbessert werden, bzw. haben $\widetilde{x}$ und x im schlechtesten Fall den gleichen Zielfunktionswert.

Diese Eigenschaften sprechen dafür, Algorithmus 5 als zusätzlichen Schritt in beliebige Verfahren zur Bestimmung einer zulässigen Lösung von (SCP) zu implementieren.

Eine echte Verbesserung einer Lösung x von (SCP) durch Algorithmus 5 tritt im Allgemeinen aber nur dann auf, wenn die Überdeckungsmatrix A des zugrunde liegenden (SCP) fast C1P hat. Daher sollte der Algorithmus vor allem für solche Mengenüberdeckungsprobleme angewendet werden.

5.2.4 Algorithmus 6

Algorithmus 6 kombiniert die beiden Verfahren, eine untere Schranke von (SCP) durch Lösen des Problems (SCPl) und eine obere Schranke von (SCP) durch Algorithmus 3 zu berechnen. Ein Nachteil dieses Verfahrens ist, dass es sich für Überdeckungsmatrizen A ohne C1P-Zeilen nur dann von Algorithmus 3 unterscheidet, wenn durch zusätzlichen Aufwand zunächst die Spalten von A so permutiert werden, dass es Zeilen mit C1P gibt.

Falls es in der Überdeckungsmatrix A Zeilen mit C1P gibt, so liefert Algorithmus 6 mit kurzer Laufzeit eine obere Schranke für (SCP). Die Ergebnisse aus Abschnitt 4.3 zeigen, dass sich das Verfahren für Probleme mit fast C1P gut eignet. Zum einen ist der Approximationsfaktor des Algorithmus in diesem Fall sehr gut, zum anderen kann der Algorithmus mit kurzer Laufzeit durchgeführt werden.

5.2.5 Algorithmus 7

Algorithmus 7 nutzt die gleiche Idee wie Algorithmus 6, statt des Problems (SCPl) wird aber das Problem (S$\mathring{\text{C}}$Pl) zur Berechnung einer unteren Schranke von (SCP) genutzt. Der Vorteil dieses Verfahrens gegenüber Algorithmus 6 liegt darin, dass es sich für Probleme mit beliebigen Überdeckungsmatrizen von Algorithmus 3 unterscheidet, insbesondere also auch für solche, die keine C1P-Zeilen haben. Der Nachteil liegt allerdings darin, dass die Berechnung einer Lösung von (S$\mathring{\text{C}}$Pl) einen erheblich größeren Aufwand benötigt als das Lösen von (SCPl).

Die Ergebnisse aus Abschnitt 4.3 zeigen, dass der Approximationsfaktor von Algorithmus 7 für Mengenüberdeckungsprobleme mit fast C1P im selben Bereich wie der Approximationsfaktor von Algorithmus 6 liegt. Der Aufwand von Algorithmus 7 ist dabei deutlich größer. Allerdings ist zu beachten, dass wir die auftretenden C1P-Mengenüberdeckungsprobleme mit dem Simplex-Verfahren gelöst haben. Durch Nutzung des Verfahrens aus Bemerkung 1.3.4 könnte die Laufzeit von Algorithmus 7 beschleunigt werden.

5.2.6 Algorithmus 1

Für ungewichtete Mengenüberdeckungsprobleme kann mit Algorithmus 1 eine obere Schranke berechnet werden. Die Güteabschätzungen für den absoluten und relativen Fehler dieser Schranke hängen linear von der Anzahl der Zeilen der Überdeckungsmatrix A des Problems ab. Bei großen Problemen können wir daher im Allgemeinen nicht erwarten, dass Algorithmus 1 eine gute obere Schranke liefert. Da der Algorithmus eine kurze Laufzeit hat und im Einzelfall sogar besser als Algorithmus 4 ist, kann es aber durchaus sinnvoll sein, ihn zu nutzen.

Literaturverzeichnis

[1] Balas, E. und M.C. Carrera: *A dynamic subgradient-based branch-and-bound procedure for set covering.* Operations Research, 44:875–890, 1996.

[2] Balas, E. und A. Ho: *Set covering algorithms using cutting planes, heuristics, and subgradient optimization: a computational study.* Mathematical Programming, 12:37–60, 1980.

[3] Beasley, J.E.: *An algorithm for set covering problems.* European Journal of Operational Research, 31:85–93, 1987.

[4] Beasley, J.E.: *A Lagrangian heuristic for set covering problems.* Naval Research Logistics Quarterly, 37:151–164, 1990.

[5] Beasley, J.E. und P.C. Chu: *A genetic algorithm for the set covering problem.* European Journal of Operational Research, 94:392–404, 1996.

[6] Beasley, J.E. und K. Jornsten: *Enhancing an algorithm for set covering problems.* European Journal of Operational Research, 58:293–300, 1992.

[7] Caprara, A., M. Fischetti und P. Toth: *Algorithms for the set covering problem.* Annals of Operations Research, 98:353–371, 2000.

[8] Ceria, S., P. Nobili und A. Sassano: *Set covering problem.* In: *Dell'Amico, M., Moffioli, F., Martelle, S. (Eds.), Annotated bibliographies in combinatorial optimization*, Seiten 415–428. J. Wiley and Sons, 1997.

[9] Ceria, S., P. Nobili, A. Sassano und R. Marzo: *A Lagrangian-based heuristic for large-scale set covering problems.* Mathematical Programming, 81:215–228, 1995.

[10] Chvatal, V.: *A greedy heuristic for the set-covering problem.* Mathematics of Operations Research, 4:233–235, 1979.

[11] Corman, T.H. et al.: *Introduction to Algorithms.* MIT Press, 2. Auflage, 2003.

[12] Feige, U.: *A threshold of ln n for approximating set cover.* Journal of the ACM, 45:314–318, 1998.

[13] Feo, T.A. und M.G.C. Resende: *A probabilistic heuristic for a computationally difficult set covering problem.* Operations Research Letters, 8:67–71, 1989.

[14] Garey, M.R. und D.S. Johnson: *Computers and Intractability(A guide to the theory of NP-completness).* Freeman, New York, 1979.

[15] Goldschmidt, O., D.S. Hochbaum und G. Yu: *A modified greedy heuristic for the set covering problem with improved worst case bound.* Information Processing Letters, 48:305–310, 1993.

[16] Hamacher, H.W. und K. Klamroth: *Lineare und Netzwerk-Optimierung - a bilingual textbook.* Vieweg, 2. Auflage, 2006.

[17] Hochbaum, D.S. und A. Levin: *Optimizing over consecutive 1's and circular 1's constraints.* SIAM Journal on Optimization, 17:311–330, 2006.

[18] Hoffman, K.L. und M.W. Padberg: *Solving airline crew scheduling problems by branch-and-cut.* Management Science, 39:657–682, 1993.

[19] Karmarkar, N., M.G.C. Resende und K.G. Ramakrishnan: *An interior point algorithm to solve computationally difficult set covering problems.* Mathematical Programing, 52:597–618, 1991.

[20] Meidanis, J. und G. Telles: *On the consecutive ones property.* Discrete Applied Mathematics, 88:325–354, 1998.

[21] Nemhauser, G.L. und L.A. Wolsey: *Integer and Combinatorial Optimization.* John Wiley & Sons, Inc., New York, 1988.

[22] Paias, A. und J. Paixão: *State space relaxation for set covering problems related to bus driver scheduling.* European Journal of Operational Research, 71:303–316, 1993.

[23] Ruf, N. und A. Schoebel: *Set covering with almost consecutive ones property.* Discrete Optimization, 1:215–228, 2004.

[24] Schöbel, A.: *Optimization in Public Transportation: Stop Location, Delay Management and Tariff Zone Design in a Public Transportation Network (Springer Optimization and Its Applications).* Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[25] Slavík, P.: *A Tight Analysis of the Greedy Algorithm for Set Cover.* Journal of Algorithms, 25:237–254, 1997.

[26] Toregas, C. und C. ReVelle: *Binary logic solutions to a class of location problems.* Geographical Analysis, 5:145–155, 1973.

[27] Toregas, C., R. Swain, C. ReVelle und L. Bergman: *The location of emergency service facilities.* Operations Research, 19:1363–1373, 1971.

[28] Vasko, F.J., F.E. Wolf und K.L Stott: *Optimal selection of ingot sizes via set covering.* Operations Research, 35:346–353, 1987.

[29] Vasko, F.J., F.E. Wolf und K.L Stott Jr.: *A set covering approach to metallurgical grade assignment.* European Journal of Operational Research, 38:27–34, 1989.

[30] Walker, W.: *Using the set-covering problem to assign fire companies to fire houses.* Operations Research, 22:275–277, 1974.

Algorithmenverzeichnis

1 „best-cover“-Blockauswahl 29
2 Greedy-Heuristik 37
3 Greedy-Blockauswahl 39
4 Greedy-Blockauswahl für ungewichtete (SCP) 45
5 Verbesserung einer zulässigen Lösung von (SCP) durch Blockauswahl 48
6 Obere Schranke für (SCP) mit (SCPl) 53
7 Obere Schranke für (SCP) mit ($\mathring{\text{SC}}$Pl) 54

Erklärung

Hiermit versichere ich, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Göttingen, den 7. Januar 2009.