FAKULTÄT FÜR MATHEMATIK, INFORMATIK
UND NATURWISSENSCHAFTEN

RHEINISCH-WESTFÄLISCHE TECHNISCHE HOCHSCHULE AACHEN

Bachelorarbeit in Mathematik

im September 2011

Eine neue Formulierung für das Frequenzzuweisungsproblem

Sebastian Goderbauer

LEHRSTUHL II FÜR MATHEMATIK

Prof. Dr. Arie M.C.A. Koster
Professor für Diskrete Optimierung

Ich versichere, dass ich diese Bachelorarbeit selbstständig verfasst und nur die angegebenen Quellen und Hilfsmittel verwendet habe.

Aachen, den 29. September 2011 Sebastian Goderbauer

Inhaltsverzeichnis

Tabellenverzeichnis

Abbildungsverzeichnis

1 Einführung

1.1 Frequenzzuweisung in Mobilfunknetzen

"UMTS steht für Unerwartete Mehreinnahmen zur Tilgung von Staatsschulden"[1] – Zu dieser Aussage ließ sich im Jahre 2000 der damalige deutsche Bundesfinanzminister Hans Eichel hinreißen. Zuvor brachte die erste Versteigerung von UMTS-Mobilfunklizenzen allein in Deutschland die Rekordsumme von umgerechnet über 50 Milliarden Euro ein. In Wahrheit steht UMTS für "Universal Mobile Telecommunications System" und ist ein moderner Mobilfunkstandard, der hohe Übertragungsgeschwindigkeiten möglich macht. Diese Technologie ist der Nachfolger der analogen Übertragungen sowie dem ersten Digital-Standard GSM und wird deshalb auch als Mobilfunk der dritten Generation 3G bezeichnet. Doch warum bezahlten die vier Auktionsgewinner E-Plus, O2, T-Mobile und Vodafone eine derartig hohe Summe für die Lizenzen von UMTS-Frequenzen?

Die rasante Entwicklung drahtloser Kommunikation, man denke beispielsweise an die mobile Nutzung des Internets über Smartphone und Netbook, hat das Frequenzspektrum zu einer sehr begehrten Ressource gemacht. Ein Beleg dafür ist u.a. der Preis der oben angesprochenen UMTS-Lizenzen. Trotz weiter ansteigender Nachfrage ist die Anzahl der verfügbaren Frequenzen begrenzt und so muss äußerst effizient mit ihnen umgegangen werden. Dabei ist die Mehrfachverwendung von Frequenzen unumgänglich, jedoch wegen gegenseitig auftretender Störungen, sogenannten Interferenzen, in nur begrenztem Umfang möglich. Drahtlose Datenübertragung findet mit Hilfe von elektromagnetischen Wellen statt. Überlagern sich verschiedene Wellen, können sich diese gegenseitig verstärken, abschwächen oder sogar ganz auslöschen. Wird die Interferenz zu hoch, so lassen sich die Signale nicht mehr eindeutig auseinanderhalten. Interferenz kann somit zu einem Datenverlust während der Übertragung führen. Es ist demnach sehr wichtig, die Frequenzen unter Einhaltung gewisser Forderungen möglichst günstig den einzelnen Übertragungskanälen zu zuweisen. Genau diese Aufgabe wird mit dem Frequenzzuweisungsproblem, kurz FAP für *Frequency Assignment Problem*, beschrieben. Durch Lösen des FAPs ist die optimale Balance zwischen Wiederverwendung von Frequenzen und Qualitätseinbußen durch Interferenz in einem Netzwerk gefunden.

Seit der ersten Verwendung von drahtloser Datenübertragung wurde das Frequenzzuweisungsproblem schon von vielen Wissenschaftlern mit den verschiedesten Ansätzen aus der diskreten Optimierung untersucht. Abhängig von der genauen Anwendung und dem Ziel werden eine Vielzahl von Lösungstechniken und Modellen in der Literatur behandelt. Die große Anzahl

[1] SPIEGEL online, P. Wittrock: *Laufzeit zu verkaufen, meistbietend*, 13.07.2010, `http://www.spiegel.de/politik/deutschland/0,1518,706264,00.html`, zuletzt aufgerufen am 20.09.2011

liegt nicht zuletzt darin begründet, dass das FAP zu der Klasse der NP-vollständigen Problemen gehört. Dies bedeutet, dass kein Algorithmus existiert, der das Problem in polynomineller Zeit in Abhändigkeit der Größe der Eingabedaten löst, sofern nicht P=NP gilt. Die einfachste Form des FAP kann auf das NP-vollständige *Graphenfärbungsproblem* reduziert werden [7].

In dieser Arbeit wird das Frequenzzuweisungsproblem unter dem Gesichtspunkt der *ganzzahligen linearen Optimierung* betrachtet. Hierzu liefert der folgende Abschnitt eine knappe Einführung in die Thematik dieser Methode der diskreten Optimierung.

1.2 Ganzzahlige Lineare Optimierung

Beispiel.
Ein Betrieb stellt zwei Produkte P_1 und P_2 auf zwei Maschinen M_1 und M_2 her. Mit P_1 wird ein Gewinn von 5 Euro pro Stück erzielt und mit P_2 einer von 4 Euro pro Stück. Maschine M_1 steht höchstens 9 Stunden zur Verfügung, Maschine M_2 höchstens 11 Stunden. Für die Herstellung von P_1 werden pro Einheit 2 Stunden auf Maschine M_1 und 3 Stunden auf Maschine M_2 benötigt. Bei Produkt P_2 sind es pro Einheit 2 Stunden auf Maschine M_1 und 1 Stunde auf Maschine M_2.
Welches Produktionsprogramm hat der Betrieb zu wählen, damit der Gewinn maximal ist?

Die Suche nach dem optimalen Produktionsprogramm ist ein typisches Problem der linearen Optimierung und kann mathematisch als lineares Programm formuliert werden. Seien x_1 und x_2 die von P_1 bzw. P_2 produzierten Einheiten.

$$\begin{aligned} \text{maximiere} \quad & 5x_1 + 4x_2 \\ \text{sodass} \quad & 2x_1 + 2x_2 \leq 9 \\ & 3x_1 + x_2 \leq 11 \\ & x_1, x_2 \geq 0 \end{aligned}$$

Dieses Programm kann leicht per Hand graphisch, siehe Abbildung 1.1, sowie mit dem *Simplex-Algorithmus* [18] oder aber auch mit Software [15, 10] optimal gelöst werden.

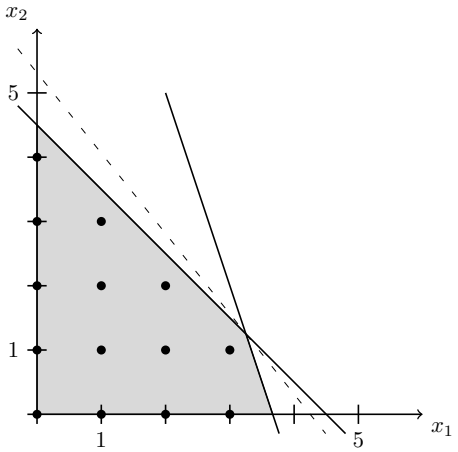

Abbildung 1.1: Polytop des linearen Programms mit zulässigen ganzzahligen Punkten

Die maximale Zielfunktion unter Einhaltung der Nebenbedingungen wird mit $x_1^{\text{opt}} = \frac{13}{4}$ und $x_2^{\text{opt}} = \frac{5}{4}$ angenommen. Aus komplexitätstheoretischer Sicht ist das Lösen eines linearen Programms ein einfaches Problem, da es sich beispielsweise mit dem *Inneren-Punkt-Verfahren* in polynominaler Zeit lösen lässt. Um ein in dem Beispiel gesuchtes Produktionsprogramm umzusetzen, wird allerdings, wie in vielen anderen Anwendungen auch, eine ganzzahlige Lösung benötigt. Durch Hinzufügen der Bedingungen $x_1, x_2 \in \mathbb{Z}$ wird das obrige Programm zu einem ganzzahligen linearen Programm.

Definition 1.1.
Seien $c \in \mathbb{R}^n$, $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$ gegebene Parameter sowie $x \in \mathbb{R}^n$ Variablen. Ein *ganzzahliges lineares Programm* wird geschrieben als:

$$\begin{aligned} \max\ & c^T x \\ \text{s.t. } & Ax \leq b \\ & x \geq 0 \\ & y \text{ ganzzahlig} \end{aligned}$$

Wie die lineare Optimierung beschäftigt sich auch die ganzzahlige lineare Optimierung mit der Optimierung linearer Zielfunktionen über einer Menge, die durch lineare Gleichungen und Ungleichungen eingeschränkt ist. Der Unterschied liegt darin, dass in der ganzzahligen Optimierung für einige oder alle Variablen ganzzahlige Werte gefordert werde und nicht beliebige reelle wie in der linearen Optimierung. Im Gegensatz zur linearen Programmierung ist bei der ganzzahligen Optimierung die zugrundeliegende Menge meist nicht genau bekannt bzw. nicht effizient beschreibbar, was das Problem aus komplexitätstheoretischer Sicht NP-schwer macht.

Neben der Produktionsprogrammplanung sind der öffentliche Verkehr, Tourenplanungen, Zeitplanerstellung, Schnittprobleme sowie Telekomunikationsnetze weitere Anwendungsbereiche der ganzzahligen Optimierung.

Zur Lösung ganzzahliger Optimierungsprobleme können exakte Lösungsverfahren wie beispielsweise *Branch-and-Bound* und das *Schnittebenenverfahren* verwendet werden. Diese basieren auf der Lösung vieler ähnlicher linearer Programme. Nichtsdestotrotz ist das optimale Lösen ganzzahliger linearer Programme in der Praxis eine schwere Aufgabe, die je nach Größe und Beschaffenheit des zu lösenden Problems eine geschickte Modellierung sowie vorherige Analyse des Lösungsraumes erfordert. Letzteres beinhaltet das Auffinden von *gültigen Ungleichungen* (siehe Abschnitt 4.2.3) und im besten Falle *facettendefinierenden Ungleichungen*.

Sämtliche genannten Verfahren und Analysestrategien sowie weitere Theorie zur ganzzahligen linearen Optimierung ist [17] zu entnehmen.

1.3 Gliederung der Arbeit

In dieser Bachelorarbeit wird ein neues Modell für ein bestimmtes Frequenzzuweisungsproblem eingeführt und mit bekannten Modellen verglichen.

Um eine Formulierung für ein Optimierungsproblem zu finden, ist in einem ersten Schritt das Optimierungsproblem mathematisch zu beschreiben. Die Mathematisierung des Problems erfolgt in Kapitel 2. Dort werden erste Definitionen und Notationen eingeführt, sowie drei verschiedene Probleme der Frequenzzuweisung erläutert. Das *Minimum Order Frequency Assignment Problem* in Abschnitt 2.2 sowie das *Minimum Span Frequency Assignment Problem* in Abschnitt 2.3 sollen einen ersten Eindruck der Problematik liefern. Das in Abschnitt 2.4 erläuterte *Minimum Interference Frequency Assignment Problem* ist das in dieser Arbeit behandelte Problem. Im Anschluss wird noch die Beschaffenheit bekannter Instanzen für dieses Problem angegeben. Diese sind die Grundlage jeden Modells.

Kapitel 3 enthält die erste Formulierung dieser Arbeit, die *Partial Constraint Satisfaction Formulierung* (PCS). Nach einer ersten allgemeinen Erläuterung des Ursprungs dieser Formulierung wird es passend für das Frequenzzuweisungsproblem eingeführt. Um dieses Modell effektiv umzusetzen, ist die Form der Instanzen zu modifizieren, die Größe dieser kann dadurch sogar reduziert werden. Das Preprocessing der Instanzen wird in Abschnitt 3.2.1 erklärt. Besonders das erweiterte Preprocessing nimmt in dem späteren Kapitel 4 einen großen Stellenwert ein. Zum Abschluss dieses Kapitels werden in Abschnitt 3.2.2 die Ergebnisse der mit dieser Formulierung durchgeführten Rechenstudie ausgewertet.

In dem Kapitel 4 wird das Hauptergebnis dieser Bachelorarbeit dokumentiert, die *Penalty Block Formulierung*. Zuerst wird in Abschnitt 4.1 die auslösende Idee für die neue Formulierung dargelegt. Es handelt sich dabei um eine erkennbare Struktur in den aus dem Preprocessing ergebenen Daten. Über die Erste Reformulierung in Abschnitt 4.2 sowie die Zweite Reformulierung in Abschnitt 4.3 wird schließlich in Abschnitt 4.5 das Modell der Penalty Block Formulierung angegeben. Die Erste Reformulierung wird außerdem in einer theoretischen Analyse mit der PCS-Formulierung aus Kapitel 3 verglichen. Dazu wird zusätzlich die bekannte Theorie zu Formulierungsvergleichen erweitert, damit eine vertiefende Analyse dieser beiden spezifischen Formulierungen möglich ist. Zudem werden gültige Ungleichungen für die Erste Reformulierung angegeben.
Um das neue Modell umzusetzen, ist ein Algorithmus zum Auffinden von Blöcken in Matrizen zu implementieren. Der in dieser Arbeit verwendete Algorithmus wird in dem Abschnitt 4.5.1 angegeben und ausführlich an einem Beispiel erläutert. Anschließend werden in Abschnitt 4.5.2 die Ergebnisse der mit dieser neuen Formulierung durchgeführten Rechenstudie ausgewertet und mit den bisherigen Ergebnissen verglichen. In zwei weiteren Abschnitten 4.5.3 und 4.5.4 werden Ansätze erklärt, die manuell zu besseren Schranken der nicht optimal gelösten Instanzen führen.

Die dritte in dieser Arbeit vorgestellte Formulierung ist die *Frequency Assignment Formulierung* in Kapitel 5. Dieses nur kurz erläuterte Modell verfolgt einen anderen Ansatz als die vorherigen

Formulierungen. Die Ergebnisse der Rechenstudie werden in Abschnitt 5.1.1 ausgewertet.

Zum Abschluss dieser Arbeit enthält Kapitel 6 die Zusammenfassung aller Ergebnisse und weitere Arbeitsansätze, die aufgrund der begrenzten Zeit für diese Bachelorarbeit nicht weiter verfolgt werden konnten.

Die in den nachfolgenden Definitionen eingeführte Notation sei in dieser Arbeit vereinbart:

Definition 1.2.
Mit $\text{Pot}(X) := \{U : U \subseteq X\}$ sei die *Potenzmenge*, also die Menge aller Teilmengen der Grundmenge X bezeichnet.

Definition 1.3.
Mit

$$\delta(A) := \begin{cases} 1 & \text{falls die logische Aussage } A \text{ wahr ist} \\ 0 & \text{sonst} \end{cases}$$

sei die *Kronecker-Delta-Funktion* bezeichnet, welche gleich 1 ist, falls A eine wahre Aussage ist und in allen anderen Fällen gleich 0 ist.

1.4 Implementierung der Modelle und Rechenstudien

Im Rahmen dieser Bachelorarbeit wurden sämtliche angegebene Modelle implementiert: die Partial Constraint Satisfaction Formulierung (Kapitel 3), die Penalty Block Formulierung (Kapitel 4) und die Frequency Assignment Formulierung (Kapitel 5). Für die beiden erstgenannten wurden das einfache sowie das erweiterte Preprocessing (Abschnitt 3.2.1) programmiert. Dieses Programm enthält auch einen Output von drei Textdateien, in denen der gesamte Interferenz-Graph, inklusive aller Penalty-Matrizen, abgespeichert ist. Für die Penalty Block Formulierung wurde des Weiteren der Algorithmus 4.22 zum Auffinden der Blöcke in den Matrizen implementiert.

Die Programmierung fand in C++ statt. Die Umsetzung der Modelle erfolgte dabei in Verbindung mit dem MIP Interface [4]. Dies sorgt für eine Schnittstelle zu den verwendeten Solvern Scip $2.0.1$ [15] und Cplex 12.2 [10]. Für die Modellierung der mathematischen Graphen innerhalb von C++ wurde die Bibliothek LEMON [14] verwendet. Im Zuge von Abschnitt 4.2.3 wurde die Software PORTA [3] benutzt.

Zu jedem implementierten Modell wurden Rechenstudien an Rechnern am Lehrstuhl II für Mathematik der RWTH Aachen mit $2{,}93$ GHz Quad-Core Prozessor und 12 GB RAM durchgeführt. Jede Rechnung war dabei auf einen Kern begrenzt und lief maximal 12 Stunden. Falls vom Default abweichende Einstellungen der Solver verwendet wurden, ist dies explizit in der Arbeit angegeben. Ergebnisse der Rechenstudien sind in Tabellen zusammengefasst, die für jede Probleminstanz die Ergebnisse des Solvers Scip sowie Cplex enthalten. Falls eine Instanz innerhalb der 12 Stunden optimal gelöst werden konnte, ist die dazu benötigte Zeit in dem Format $h{:}mm{:}ss$ (h Stunden, mm Minuten, ss Sekunden) angegeben.

2 Das Frequency Assignment Problem

Dieses Kapitel enthält die Mathematisierung des Frequenzzuweisungsproblems. In einem ersten Abschnitt wird das Problem allgemein abstrahiert und abgegrenzt. Außerdem werden erste Bezeichnungen eingeführt. In den weiteren Abschnitten werden drei Blickwinkel auf das FAP erläutert. Diese Ansätze münden jeweils in einem Modell zum Lösen des Problems. Dabei steht das *Minimum Interference Frequency Assignment Problem*, kurz MI-FAP, im Vordergrund dieser Arbeit. Zu diesem wird zusätzlich noch auf die behandelten Probleminstanzen eingegangen.

Entnommen wurde der Inhalt dieses Kapitels zum größten Teil aus [1, 6, 12, 16].

2.1 Mathematisierung und allgemeine Definition des Problems

Das Frequenzzuweisungsproblem besteht zum Einen aus der Zuweisung von Frequenzen gegenüber Kommunikationsverbindungen und zum Anderen aus dem Auftreten von Störungen im Sinne von Interferenzen zwischen zwei zugewiesenen Frequenzen.
Einer Menge von kabellosen Kommunikationsverbindungen sollen Frequenzen zugewiesen werden, sodass bei jeder Verbindung ein Datentransfer zwischen Sender und Empfänger möglich ist. Hierfür steht eine gewisse Menge von Frequenzen zur Verfügung. Häufig ist eine Verbindung bidirektional. Das heißt, es müssen jeder solchen Verbindung zwei Frequenzen zugewiesen werden (für jede Richtung eine).
Frequenzen zweier Verbindungen können interferieren, dies führt zu verminderter Qualität des übermittelten Signals. Im Allgemeinen sind zwei Bedingungen zu erfüllen, damit Interferenz zweier Wellen auftritt. Einerseits müssen die Frequenzen im elektromagnetischen Spektrum nah beieinander liegen. Andererseits müssen die Wellen stark genug, also der räumliche Abstand der beiden Sender klein sein. Außerdem müssen die beiden Signale an der Position, wo sie interferieren, ein ähnliches Energielevel besitzen.

Die Verfügbarkeit von Frequenzen wird meist von staatlichen Institutionen reguliert. Etwaige Unternehmen können Lizenzen erwerben, um bestimmte Frequenzen in einem gewissen Gebiet nutzen zu dürfen. Das dem Betreiber verfügbare Frequenzband $[f_{min}, f_{max}]$ wird für gewöhnlich in eine Menge von Kanälen der Bandbreite Δ unterteilt. Somit können diese Kanäle von 1 bis zu einem Maximum $N = (f_{max} - f_{min})/\Delta$ durchnummeriert werden. Die Menge der verfügbaren Kanäle wird mit $D = \{1, \ldots, N\}$ bezeichnet. Grundsätzlich sind für eine einzelne Verbindung nicht alle Kanäle aus D verwendbar. Beispielsweise unterliegen Verbindungen nahe Landesgrenzen gewissen Einschränkungen, die die Menge der verfügbaren Kanäle verkleinern. Die Menge der verfügbaren Kanäle für eine Verbindung v wird mit $D_v \subseteq D$ bezeichnet.
Wie schon angesprochen, werden für bidirektionalen Datenverkehr zwei Kanäle benötigt. Dies

wird in den meisten Modellen aus folgendem Grund vernachlässigt: Anstatt nur eines Frequenzbandes $[f_{min}, f_{max}]$, sind zwei Bänder $[f^1_{min}, f^1_{max}]$ und $[f^2_{min}, f^2_{max}]$ mit jeweils N Kanälen gegeben. Das erste enthält die Kanäle $\{1, \ldots, N\}$ und das zweite die Kanäle $\{s+1, \ldots, s+N\}$ mit $s \gg N$. Somit wird für die Rückrichtigung einfach ein um s Kanäle nach oben verschobener Kanal verwendet. Dabei ist s ausreichend groß, sodass zwischen der Hin- und Rückrichtung einer Verbindung keine Interferenz auftreten kann. Schließlich ist es so möglich, aus jeder zugewiesenen Frequenz für die Hinrichtung, die Frequenz für die Rückrichtig zu folgern.

In den meisten Modellen wird ignoriert, dass sich an einem Punkt mehr als zwei Signale stören und interferieren können. Das Hauptaugenmerk liegt auf der Interferenz zwischen Paaren von Verbindungen. Für die Modellierung eines FAPs sind des Weiteren Bewertungskriterien für die auftretende Interferenz aufzustellen. Es wird angenommen, dass gewisse Strafwerte für einzelne Interferenzausprägungen als Input für ein Modell zur Verfügung stehen.

Es ist möglich, dass mehr als eine Verbindung zwischen den gleichen Punkten gefordert wird. Dies ist im Allgemeinen mit einem Parameter $c_v \in \mathbb{Z}^+$ modelliert, der angibt, wie viele Frequenzen einer Verbindung v zugewiesen werden müssen. In der Praxis ändert sich dieser Wert mit der Zeit, abhängig von der aktuellen Nachfrage einer Verbindung. In dieser Arbeit wird das FAP als *Fixed Channel Assignment* (FCA) behandelt. Das bedeutet, auf eine Änderung des Parameters c_v kann nicht kurzfristig reagiert werden. Es findet eine feste und statische Kanalvergabe statt. Eine *online* Änderung [13] um neuen Bedarf zu stillen ist nicht möglich. Somit ist ein solches Problem stets neu optimal zu lösen, wenn sich die Situation ändert.

Grundlage eines FAPs ist standardmäßig ein *Interferenz-Graph*.

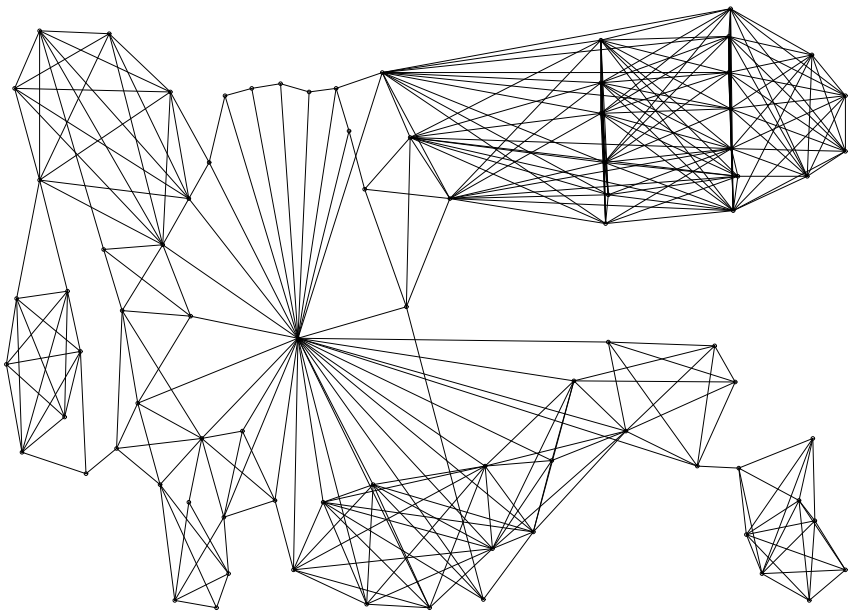

Abbildung 2.1: Interferenz-Graph der Instanz `CELAR06`
entnommen aus: Koster, *Frequency Assignment - Models and Algorithms*, 1999 [12]

Definition 2.1.
In einem *Interferenz-Graphen* $G = (V, E)$ wird jede Verbindung durch einen Knoten $v \in V$ dargestellt. Die verfügbaren Kanäle oder Frequenzen für einen Knoten sind beschrieben durch die Menge $D_v \subseteq D$. Weiter ist c_v die Anzahl der zuzuweisenden Frequenzen einer Verbindung $v \in V$. Eine Kante $\{v, w\} \in E$ zwischen zwei Knoten $v, w \in V$ besagt, dass zu den Verbindungen v, w mindestens ein Frequenzpaar $f \in D_v, g \in D_w$ vorhanden ist, sodass diese interferieren können. Für jedes derartige Paar von Frequenzen wird die kombinierte Auswahl bestraft, wobei die Höhe von dem Ausmaß der Interferenz abhängig ist. Dieser Penalty-Wert wird mit p_{vwfg} bezeichnet. Für je eine Kante $\{v, w\} \in E$ bilden diese Werte eine sogenannte Penalty-Matrix. Bei einigen Instanzen sind schon Frequenzen diversen Verbindungen zugewiesen, diese zu modifizieren wird dementsprechend bestraft. Die Wahl der Frequenz $f \in D_v$ für Verbindung $v \in V$ kostet q_{vf}.

Beispiel.
Abbildung 2.1 zeigt den Interferenz-Graphen der Instanz `CELAR06`. Die Herkunft dieser Daten wird in Abschnitt 2.4.1 genauer erläutert.

In vielen Instanzen ist das Ausmaß der Interferenz mit der Distanz zwischen zwei zugewiesenen Frequenzen verknüpft: Je kleiner der Abstand zweier Frequenzen, desto größer ist p_{vwfg}. Dabei ist Interferenz als inakzeptabel definiert, falls für zwei Frequenzen $f \in D_v$ und $g \in D_w$ der Abstand kleiner als ein d_{vw} ist. Folglich ist das Ausmaß an Interferenz akzeptabel, falls $|f - g| \geq d_{vw}$ erfüllt ist. Verallgemeinert sei eine Frequenzzuweisung für Verbindungen v und w nicht zulässig, falls der Wert der Differenz in einer Menge T_{vw} enthälten ist. Für eine zulässige Zuweisung muss also stets $|f - g| \notin T_{vw}$ gelten.

Es werden übergeordnet zwei Ziele zum Lösen eines FAPs betrachtet. Es sollen Frequenzen zugewiesen werden, sodass entweder die Summe aller Interferenzen minimiert oder das Maximum der Interferenzen minimiert wird.

Auf Ersterem liegt der Schwerpunkt dieser Arbeit, die exakte Problemdefinition des *Minimum Interference Frequency Assignment Problems* wird in Abschnitt 2.4 eingeführt. Um einen Einblick in die Thematik der Frequenzzuweisung zu bekommen, werden zunächst in kürzerer Form Problemstellungen aus der zweiten Kategorie behandelt.

Anstatt die maximale Strafe zu minimieren, wird oft eine Frequenzzuweisung gesucht, bei der jede Interferenz einen gegebenen Wert nicht überschreitet. Somit sind einige Frequenzkombinationen verboten. Dies hat zur Folge, dass die Penalty-Matrix zu einer $\{0, 1\}$-Matrix reduziert werden kann. Kombinationen mit Penalty-Wert 0 sind erlaubt, alle andere nicht. Es wird nun gefordert, eine zulässige Frequenzzuweisung zu finden. In einem solchen Fall wird oft eine zweite Zielfunktion definiert, die eine Wertung aller zulässigen Lösungen ermöglicht. Dies führt zum Einen zu dem *Minimum Order Frequency Assignment Problem*, mit der Zielsetzung die Anzahl der verwendeten Frequenzen zu minimieren (siehe Abschnitt 2.2), und zum Anderen zu dem *Minimum Span Frequency Assignment Problem*, mit der Zielsetzung die Differenz zwischen der größten und kleinsten verwendeten Frequenz zu minimieren (siehe Abschnitt 2.3).

2.2 Minimum Order Frequency Assignment Problem

In dem Minimum Order Frequency Assignment Problem (MO-FAP) wird eine Frequenzzuweisung gefordert, die keine inakzeptablen Interferenzen beinhaltet und die Anzahl der verschiedenen, genutzten Frequenzen minimiert.

Definition 2.2.
Minimum Order Frequency Assignment Problem

INSTANZ: Ungerichteter Graph $G = (V, E)$ mit $\{v, v\} \in E$ für alle $v \in V$,
Mengen $T_{v,w} \subset \mathbb{Z}$ mit $0 \in T_{vw}$ für alle $\{v, w\} \in E$,
Bedarf $c_v \in \mathbb{Z}^+$,
Werteteilmengen $D_v \subseteq \mathbb{Z}^+$ für alle $v \in V$ mit $D = \cup_{v \in V} D_v$,
positives, ganzzahliges K

FRAGE: Existiert eine Zuweisung von Teilmengen $f : V \to \text{Pot}(D)$, sodass

(i) $|f(v)| = c_v$ für alle $v \in V$

(ii) $f(v) \subseteq D_v$ für alle $v \in V$

(iii) $|\bar{f} - \bar{g}| \notin T_{vw}$ mit $\bar{f} \in f(v)$, $\bar{g} \in f(w)$, $v \neq w$ oder $\bar{f} \neq \bar{g}$, für alle $\{v, w\} \in E$ und

(iv) $|\cup_{v \in V} f(v)| \leq K$?

Das MO-FAP ist das in der Literatur als erstes diskutierte Frequenzzuweisungsproblem und kann als ganzzahliges lineares Programm modelliert werden. Dazu sei für jeden Knoten $v \in V$ und jede verfügbare Frequenz $f \in D_v$ eine binäre Variable x_{vf} definiert.

$$x_{vf} = \begin{cases} 1 & \text{falls Frequenz } f \in D_v \text{ dem Knoten } v \in V \text{ zugewiesen} \\ 0 & \text{sonst} \end{cases}$$

Sowie für alle $f \in D$ eine binäre Variable y_f.

$$y_f = \begin{cases} 1 & \text{falls Frequenz } f \in D \text{ genutzt} \\ 0 & \text{sonst} \end{cases}$$

Minimum Order Frequency Assignment Problem (MO-FAP)

$$\textbf{min} \sum_{f \in D} y_f \tag{2.1}$$

$$\textbf{s.t.} \sum_{f \in D_v} x_{vf} = c_v \qquad \forall v \in V \tag{2.2}$$

$$x_{vf} + x_{wg} \leq 1 \qquad \forall \{v, w\} \in E, f \in D_v, g \in D_w \text{ mit } (|f - g| \in T_{vw}) \wedge ((f \neq g) \vee (v \neq w)) \tag{2.3}$$

$$x_{vf} \leq y_f \qquad \forall v \in V, f \in D_v \tag{2.4}$$

$$x_{vf} \in \{0, 1\} \qquad \forall v \in V, f \in D_v \tag{2.5}$$

$$y_f \in \{0, 1\} \qquad \forall f \in D \tag{2.6}$$

Die Nebenbedingungen (2.2) stellen sicher, dass c_v Frequenzen der Verbindung $v \in V$ zugeordnet werden. Die verbotenen Frequenzkombinationen sind durch (2.3) modelliert. Des Weiteren garantiert (2.4), dass die zugehörige y-Variable einer zugewiesenen Frequenz gesetzt wird. Die Zielfunktion (2.1) enthält die Anzahl der verwendeten Frequenzen.

2.3 Minimum Span Frequency Assignment Problem

In dem Minimum Span Frequency Assignment Problem (MS-FAP) wird eine Frequenzzuweisung gefordert, die keine inakzeptablen Interferenzen beinhaltet und die die Differenz zwischen der maximalen und minimalen verwendeten Frequenz, die Spanne, minimiert.

Definition 2.3.
Minimum Span Frequency Assignment Problem
INSTANZ: Ungerichteter Graph $G = (V, E)$ mit $\{v, v\} \in E$ für alle $v \in V$,
Mengen $T_{v,w} \subset \mathbb{Z}$ mit $0 \in T_{vw}$ für alle $\{v, w\} \in E$,
Bedarf $c_v \in \mathbb{Z}^+$,
Werteteilmengen $D_v \subseteq \mathbb{Z}^+$ für alle $v \in V$ mit $D = \cup_{v \in V} D_v$,
positives, ganzzahliges K
FRAGE: Existiert eine Zuweisung von Teilmengen $f : V \to \text{Pot}(D)$, sodass

(i) $|f(v)| = c_v$ für alle $v \in V$

(ii) $f(v) \subseteq D_v$ für alle $v \in V$

(iii) $|\bar{f} - \bar{g}| \notin T_{vw}$ mit $\bar{f} \in f(v)$, $\bar{g} \in f(w)$, $v \neq w$ oder $\bar{f} \neq \bar{g}$, für alle $\{v, w\} \in E$ und

(iv) $\max \cup_{v \in V} f(v) - \min \cup_{v \in V} f(v) \leq K$?

Zusätzlich zu den definierten Variablen aus Abschnitt 2.2 seien die Parameter $f_{max} := \max_{f \in D} f$ sowie $f_{min} := \min_{f \in D} f$ benannt. Dann lässt sich das MS-FAP als ganzzahliges lineares Programm modellieren.

Minimum Span Frequency Assignment Problem (MS-FAP)

$$\textbf{min } z_{max} - z_{min} \tag{2.7}$$

$$\textbf{s.t.} \sum_{f \in D_v} x_{vf} = c_v \qquad \forall v \in V \tag{2.8}$$

$$x_{vf} + x_{wg} \leq 1 \qquad \forall \{v, w\} \in E, f \in D_v, g \in D_w \text{ mit } (|f - g| \in T_{vw}) \wedge ((f \neq g) \vee (v \neq w)) \tag{2.9}$$

$$x_{vf} \leq y_f \qquad \forall v \in V, f \in D_v \tag{2.10}$$

$$z_{max} \geq f \cdot y_f \qquad \forall f \in D \tag{2.11}$$

$$z_{min} \leq f_{max} - (f_{max} - f) \cdot y_f \qquad \forall f \in D \tag{2.12}$$

$$x_{vf} \in \{0, 1\} \qquad \forall v \in V, f \in D_v \tag{2.13}$$

$$y_f \in \{0, 1\} \qquad \forall f \in D \tag{2.14}$$

$$z_{min}, z_{max} \in \mathbb{Z}^+ \tag{2.15}$$

Die Nebenbedingungen (2.11) und (2.12) stellen sicher, dass die Variablen z_{min} und z_{max} auf die kleinste bzw. größte verwendete Frequenz gesetzt werden.

2.4 Minimum Interference Frequency Assignment Problem

In dem Minimum Interference Frequency Assignment Problem wird eine Frequenzzuweisung gefordert, sodass die Summe aller Interferenzen minimal ist.

Definition 2.4.
Minimum Interference Frequency Assignment Problem
INSTANZ: Ungerichteter Graph $G = (V, E)$ mit $\{v, v\} \in E$ für alle $v \in V$,
Mengen $T_{v,w} \subset \mathbb{Z}$ mit $0 \in T_{vw}$ für alle $\{v, w\} \in E$,
Bedarf $c_v \in \mathbb{Z}^+$,
Werteteilmengen $D_v \subseteq \mathbb{Z}^+$ für alle $v \in V$ mit $D = \cup_{v \in V} D_v$,
Penalty-Werte $p_{vwfg} \in \mathbb{Z}^+$ für alle $\{v, w\} \in E$, $f \in D_v$, $g \in D_w$
positives, ganzzahliges K
FRAGE: Existiert eine Zuweisung von Teilmengen $f : V \to \text{Pot}(D)$, sodass

(i) $|f(v)| = c_v$ für alle $v \in V$

(ii) $f(v) \subseteq D_v$ für alle $v \in V$ und

(iii) $\sum_{\{v,w\} \in E} \sum_{\substack{\bar{f} \in f(v), \bar{g} \in f(w) \\ (v \neq w) \vee (\bar{f} \neq \bar{g})}} p_{vw\bar{f}\bar{g}} \delta(|\bar{f} - \bar{g}| \in T_{vw}) \leq K$?

Hierbei ist $\delta(A)$ die *Kronecker-Delta-Funktion* aus Definition 1.3 und folglich werden in (iii) alle $p_{vw\bar{f}\bar{g}}$ aufsummiert, mit zugewiesenen $\bar{f} \in D_v$, $\bar{g} \in D_w$, die Interferenz implizieren.

Der Spezialfall, in dem ein interferenzfreie Zuweisung gesucht wird, d.h. $K = 0$, wird auch *Feasibility Frequency Assignment Problem* genannt.

Definition 2.5.
Feasibility Frequency Assignment Problem
INSTANZ: Ungerichteter Graph $G = (V, E)$ mit $\{v, v\} \in E$ für alle $v \in V$,
Mengen $T_{v,w} \subset \mathbb{Z}$ mit $0 \in T_{vw}$ für alle $\{v, w\} \in E$,
Bedarf $c_v \in \mathbb{Z}^+$,
Werteteilmengen $D_v \subseteq \mathbb{Z}^+$ für alle $v \in V$ mit $D = \cup_{v \in V} D_v$,
Penalty-Werte $p_{vwfg} \in \mathbb{Z}^+$ für alle $\{v, w\} \in E$, $f \in D_v$, $g \in D_w$
FRAGE: Existiert eine Zuweisung von Teilmengen $f : V \to \text{Pot}(D)$, sodass

(i) $|f(v)| = c_v$ für alle $v \in V$

(ii) $f(v) \subseteq D_v$ für alle $v \in V$ und

(iii) $|\bar{f} - \bar{g}| \notin T_{vw}$ für alle $\{v, w\} \in E$, $\bar{f} \in f(v), \bar{g} \in f(w), v \neq w$ oder $\bar{f} \neq \bar{g}$?

Um das MI-FAP als ganzzahliges lineares Programm zu modellieren, werden neue binäre Variablen z_{vwfg} für alle $\{v, w\} \in E, f \in D_v, g \in D_w$ mit $|f - g| \in T_{vw}$ und entweder $v \neq w$ oder $f \neq g$ eingeführt.

$$z_{vwfg} = \begin{cases} 1 & \text{falls } x_{vf} = 1 \text{ und } x_{wg} = 1 \\ 0 & \text{sonst} \end{cases}$$

Dann lautet ein Programm für das MI-FAP wie folgt:

Minimum Interference Frequency Assignment Problem (MI-FAP)

$$\textbf{min} \sum_{\{v,w\}\in E} \sum_{\substack{f\in D_v, g\in D_w \\ |f-g|\in T_{vw}\wedge(v\neq w\vee f\neq g)}} p_{vwfg}\cdot z_{vwfg} + \sum_{v\in V}\sum_{f\in D_v} q_{vf}\cdot x_{vf} \tag{2.16}$$

$$\textbf{s.t.} \sum_{f\in D_v} x_{vf} = c_v \qquad \forall v\in V \tag{2.17}$$

$$x_{vf} + x_{wg} \leq 1 + z_{vwfg} \qquad \begin{array}{l}\forall\{v,w\}\in E, f\in D_v, g\in D_w \text{ mit} \\ (|f-g|\in T_{vw})\wedge((f\neq g)\vee(v\neq w))\end{array} \tag{2.18}$$

$$x_{vf}\in\{0,1\} \qquad \forall v\in V, f\in D_v \tag{2.19}$$

$$z_{vwfg}\in\{0,1\} \qquad \begin{array}{l}\forall\{v,w\}\in E, f\in D_v, g\in D_w \text{ mit} \\ (|f-g|\in T_{vw})\wedge((f\neq g)\vee(v\neq w))\end{array} \tag{2.20}$$

Durch Nebenbedingungen (2.18) können Frequenzen f und g nur genau dann den Verbindungen v und w zugewiesen werden, wenn z_{vwfg} gesetzt ist. Dies hat eine Erhöhung der Zielfunktion (2.16) um den Penalty-Wert p_{vwfg} zur Folge. Falls für alle Penalty-Werte $p_{vwfg} > 0$ gilt, wird nie grundlos eine z-Variable gesetzt, es sei denn, es wird von (2.18) erzwungen. Im Falle von Penalty-Werten mit $p_{vwfg} < 0$ sind die Nebenbedingungen

$$z_{vwfg} \leq x_{vf} \qquad \begin{array}{l}\forall\{v,w\}\in E, f\in D_v, g\in D_w \text{ mit} \\ (|f-g|\in T_{vw}\wedge((f\neq g)\vee(v\neq w))\end{array} \tag{2.21}$$

der Formulierung hinzuzufügen. Neben den Penalty-Werten für gewisse Kombinationen von Frequenzen wird in einigen Instanzen in der Zielfunktion (2.16) mit $q_{vf} > 0$ die Wahl von Frequenz $f \in D_v$ für Knoten $v \in V$ bestraft.

Eine andere Möglichkeit die Nebenbedingungen (2.18) zu modellieren ergibt sich durch Einführung der Variablen z_{vwfg} für alle $\{v,w\}\in E$ und $f\in D_v, g\in D_w$ in Form der folgenden Nebenbedingungen.

$$\sum_{g\in D_w} z_{vwfg} = c_w\cdot x_{vf} \qquad \forall\{v,w\}\in E, f\in D_v \tag{2.22}$$

Im Falle von $x_{vf} = 0$ bleiben auch die Variablen z_{vwfg} ungesetzt. Falls $x_{vf} = 1$, dann garantiert (2.22), dass genau c_w der z_{vwfg}-Variablen gesetzt werden und zwar genau die z_{vwfg} mit $x_{wg} = 1$. Wichtig hierbei ist, dass aufgrund des ungerichteten Graphen (V, E) für je zwei Knoten $v, w \in V$, die eine Kante in E bilden, die Gleichung (2.22) für $\{v,w\}\in E$ sowie für $\{w,v\}\in E$ gebildet wird. Die Formulierung des MI-FAP mit Nebenbedingungen (2.22) und $c_v = 1$ ist Thema von Kapitel 3 dieser Arbeit.

In dem folgenden Abschnitt 2.4.1 werden die elf in dieser Arbeit betrachteten Instanzen für das Minimum Interference Frequency Assignment Problem vorgestellt.

2.4.1 Instanzen des EUCLID CALMA Projekts

Die in dieser Arbeit verwendeten Probleminstanzen haben ihren Ursprung im Militär. Dort führt die Verwendung von Feldtelefonen zu Frequenszuweisungsproblemen. Zu jeder Verbindung gehören zwei bewegbare Telefone und es müssen zu jeder Verbindung zwei Frequenzen mit einer festen Distanz zueinander zugewiesen werden.

Die Instanzen sind aus dem EUCLID CALMA (Combinatorial Algorithms for Military Applications) Projekt hervorgegangen, bei dem Wissenschaftler aus England, Frankreich und den Niederlanden verschiedene, kombinatorische Algorithmen an der selben Menge von Frequenzzuweisungsproblemen getestet haben. Diese Menge enthält Minimum Order Probleme (siehe 2.2) sowie Minimum Span (siehe 2.3) und Minimum Interference Probleme (siehe 2.4). Für das MI-FAP wurden fünf Instanzen durch CELAR (Centre d'Electronique de l'Armement, Frankreich) und weitere sechs Instanzen durch die Technische Universität Delft, Niederlande, zu Verfügung gestellt. Die `CELAR`-Instanzen enthalten vereinfachte Daten aus echten Netzwerken. Die andere Menge von Instanzen, genannt `GRAPH` (Generating Radio Link Frequency Assignment Problems Heuristically), wurde zufällig generiert. Die Daten sind verfügbar auf der FAP Webseite [6].

Die elf Instanzen für das MI-FAP sind inklusive der Größe ihres Interferenz-Graphen und der durchschnittlichen Anzahl der verfügbaren Frequenzen pro Knoten in Tabelle 3.1 aufgeführt.

Instanz	$\|V\|$	$\|E\|$	$\varnothing\|D_v\|$
`CELAR06`	200	1.322	40
`CELAR07`	400	2.865	40
`CELAR08`	916	5.744	40
`CELAR09`	680	4.103	39
`CELAR10`	680	4.103	39
`GRAPH05`	200	1.134	37
`GRAPH06`	400	2.170	38
`GRAPH07`	400	2.170	37
`GRAPH11`	680	3.757	38
`GRAPH12`	680	4.017	38
`GRAPH13`	916	5.273	38

Tabelle 2.1: Instanzen für MI-FAP

Jede derartige Instanz besteht aus vier Textdateien: `var.txt`, `dom.txt`, `ctr.txt` und `cst.txt`. Da diese Textdateien die Grundsteine für das in Abschnitt 3.2.1 beschriebene Preprocessing sind, aus dem letztendlich der Input für die Modelle in Kapitel 3 und 4 hervorgeht, seien diese im Folgenden genauer dargelegt. Die Abbildungen 2.2 - 2.5 enthalten Ausschnitte der Textdateien der Instanz `CELAR10`.

In `var.txt` ist pro Zeile ein Knoten des Interferenz-Graphen, d.h eine Verbindung aufgelistet. Jede Zeile enthält bis zu vier Knoteninformationen, von denen zwei immer vorhanden sind. Als erstes die Nummer dieser Verbindung und als zweites die Nummer der Menge der verfügbaren

Frequenzen dieser Verbindung (siehe `dom.txt`). Es folgt ggf. die schon dieser Verbindung zugewiesene Frequenz und der Kostenindex einer Modifikation dieser Zuweisung. Ein Kostenindex von 0 repräsentiert, dass die Frequenz keinesfalls mehr geändert werden darf. Die Kostenindizes $1, 2, 3, 4$ korrespondieren mit aufsteigenden Strafkosten, die dem Vektor b in der Textdatei `cst.txt` zu entnehmen sind.

```
  1   1 442 0
  2   1 680 0
  7   1 554 0
  8   1 792 0
  9   1 694 0
 10   1 456 0
 13   1 282 1
 14   1  44 1
 15   1 282 2
 16   1  44 2
 21   3 512 1
 22   3 750 1
 23   3 512 3
 24   3 750 3
 25   1 680 0
 26   1 442 0
 27   1 310 0
 28   1  72 0
 49   1
 50   1
 51   1 414 1
...
```

Abbildung 2.2: `var.txt`

```
0  48  16  30  44  58  72  86 100 114 128 142 156 170 240 254 268 282 ...
1  44  16  30  44  58  72  86 100 114 128 142 156 254 268 282 296 310 ...
2  22  30  58  86 114 142 268 296 324 352 380 414 442 470 498 526 554 ...
3  36  30  44  58  72  86 100 114 128 142 268 282 296 310 324 338 352 ...
4  24  16  30  58  86 114 142 254 268 296 324 352 380 414 442 470 498 ...
5   6 142 170 240 380 408 478
6  42  30  44  58  72  86 100 114 128 142 156 268 282 296 310 324 338 ...
7  22  16  30  44  58  72  86 100 114 128 142 156 254 268 282 296 310 ...
```

Abbildung 2.3: `dom.txt`

Die Datei `dom.txt` enthält alle Mengen an verfügbaren Frequenzen. Pro Zeile ist eine Menge definiert, wobei die erste Menge die Vereinigung aller anderen darstellt. Jede Zeile enthält nach der Wertebereichsnummer, auf die in `var.txt` verwiesen wird, und der Kardinalität sämtliche in dieser Menge enthaltenen Frequenzen.

```
1   2 D = 238 0
1 171 L > 202 3
1 172 C >  60 2
1 199 C >  70 3
1 200 L > 218 4
1 871 L > 278 4
1 872 C > 130 3
1 897 C >  65 4
1 898 L > 208 3
1 899 C >  69 2
1 900 L > 215 4
2 171 C > 377 4
...
```

Abbildung 2.4: `ctr.txt`

Die Datei `ctr.txt` enthält die an die Frequenzzuweisung gestellten Bedingungen der Instanz, die die Kantenmenge E des Interferenz-Graphen bilden. Pro Zeile ist eine Bedingung aufgetragen. Beispielsweise sagt die erste Zeile in Abbildung 2.4 aus, dass für die Knoten 1 und Knoten 2 zugewiesenen Frequenzen f_1 und f_2 gelten muss: $|f_1 - f_2| = 238$. Die zweite Zeile fordert, dass entsprechend $|f_1 - f_{171}| > 202$ gilt. Falls eine Bedingung nicht erfüllt wird, fallen zusätzliche Strafen an, die durch weitere Kostenindizes in der letzten Spalte benannt sind. Dabei bedeutet der Kostenindex 0 erneut, dass diese Bedingung auf jeden Fall zu erfüllen ist. Die Kostenindizes $1, 2, 3, 4$ korrespondieren abermalig mit aufsteigenden Strafkosten, die dem Vektor a in der Textdatei `cst.txt` zu entnehmen sind. Die Buchstaben in der dritten Spalte deuten lediglich auf den Ursprung der Bedingung hin (D $\hat{=}$ difference, C $\hat{=}$ cosite, F $\hat{=}$ fixe, P $\hat{=}$ préfixé, L $\hat{=}$ far fields) und sind in der mathematischen Betrachtung irrelevant.

```
a1 =   1000
a2 =    100
a3 =      2
a4 =      1
b1 = 100000
b2 =  10000
b3 =    100
b4 =     10
```

Abbildung 2.5: `cst.txt`

In `cst.txt` sind mit Vektor a die Strafkosten bei Nichterfüllung einer Bedingung aus `ctr.txt` und mit Vektor b die Strafkosten bei Modifikation vorgegebener Frequenzen angegeben.

Die Instanzen enthalten pro Hin- und Rückrichtung einen Knoten, dessen zugewiesene Frequenzen durchweg eine feste Distanz (238) haben müssen. Dies ist der erste Ansatz des in Abschnitt 3.2.1 erläuterten Preprocessing der Daten.

3 Die Partial Constraint Satisfaction Formulierung

In diesem Kapitel wird die erste Formulierung dieser Arbeit für das Minimum Interference Frequency Assignment Problem behandelt, die *Partial Constraint Satisfaction Formulierung*. Dieses Modell wurde in [12] entwickelt. Es wird zunächst in dem Abschnitt 3.1 das Partial Constraint Satisfaction Problem mit binären Relationen definiert und die Verknüpfung zum MI-FAP hergestellt. In dem nachfolgenden Abschnitt 3.2 wird das MI-FAP im Zuge der Partial Constraint Satisfaction Formulierung als ganzzahliges lineares Programm modelliert. Um diese Formulierung zu implementieren sind die Instanzen aus Abschnitt 2.4.1 in eine andere Form zu bringen, dabei ist es sogar aufgrund der Struktur der Instanzen möglich, den Umfang der Daten zu verkleinern. Dieses Preprocessing wird in Abschnitt 3.2.1 dargelegt. Die Ergebnisse der Rechenstudie folgen in Abschnitt 3.2.2.

Der Abschnitt 3.1 sowie der Beginn von Abschnitt 3.2 sind inhaltlich zum größten Teil [12] entnommen.

3.1 Das Partial Constraint Satisfaction Problem

Viele Probleme der kombinatorischen Optimierung und künstlichen Intelligenz lassen sich als Bedingungserfüllungsproblem, *Constraint Satisfaction Problem* (CSP), modellieren. Ein solches Problem besteht aus einer Menge von Variablen, einer Menge von möglichen Werten für jede Variable, den sogenannten Wertemengen, und einer Menge von Bedingungen, die auf den Variablen definiert sind. Jede Bedingung impliziert verbotene Kombinationen von Werten für eine Menge von Variablen. Ziel ist es, eine Belegung der Variablen zu finden, sodass alle Bedingungen erfüllt werden. Ein binäres CSP besteht nur aus Bedingungen, welche die Kombination von Werten für Mengen von zwei Variablen einschränkt. Ein derartiges Problem wird gerne in Form eines *Constraint-Graphen* dargestellt. Dort bilden die Variablen die Menge der Knoten und eine Kante repräsentiert eine Bedingung bezüglich der inzidenten Variablen. Eine solche Struktur errinnert stark an den in Definition 2.1 eingeführten Interferenz-Graphen.

Es ist möglich, dass die Menge der Bedingungen so sehr einschränkend ist, dass es keine Belegung der Variablen gibt, die alle Bedingungen erfüllt. Ist dies der Fall, wird für die Klassifizierung der Zuweisungen eine zweite Zielfunktion benötigt. Das *Partial Constraint Satisfaction Problem* (PCSP) fordert eine Belegung der Variablen, sodass eine gegebene Zielfunktion maximiert oder minimiert wird. Hierbei wird angenommen, dass die Zielfunktion als eine Summe von Funktionen geschrieben werden kann, die nur jeweils die Belegung einer oder zwei adjazenter Variablen

enthält. Beispielsweise lautet bei dem *Maximal Constraint Satisfaction Problem* das Ziel, eine Belegung zu finden, die die meisten Bedinungen erfüllt.

Im Folgenden liegt der Fokus auf dem PCSP mit binären Bedingungen. Die zugehörige und zu minimierende Zielfunktion bestraft die Zuweisung gewisser Werte sowie die Kombination von einigen Werten für jeweils zwei Variablen. Formeller sei hierzu der *Constraint-Graph* $G = (V, E)$ wie folgt definiert: Jeder Knoten $v \in V$ repräsentiert eine Entscheidungsvariable, der ein kostenbehafteter Wert aus der gegebenen Menge D_v zugewiesen werden soll. Eine Kante $\{v, w\} \in E$ zeigt an, dass eine Kombination von Werten der Knoten v und w existiert, die bestraft wird.

Definition 3.1.
Partial Constraint Satisfaction Problem
INSTANZ: Ungerichteter Graph $G = (V, E)$ mit $\{v, v\} \in E$ für alle $v \in V$,
endliche Werteteilmengen D_v für alle $v \in V$ mit $D = \cup_{v \in V} D_v$,
Kostenfunktion $Q_v : D_v \to \mathbb{Z}^+$ für alle $v \in V$,
Strafenfunktion $P_{vw} : D_v \times D_w \to \mathbb{Z}^+$ für alle $\{v, w\} \in E$,
positives, ganzzahliges K
FRAGE: Existiert eine Belegung $f : V \to D_v$ mit $d_v := f(v)$, sodass

$$\sum_{v \in V} Q_v(d_v) + \sum_{\{v,w\} \in E} P_{vw}(d_v, d_w) \leq K \text{ gilt?}$$

Es ist schnell ersichtlich, dass das MI-FAP zu der Klasse der PCSPs gehört. Der Infererenz-Graph aus Definition 2.1 ist hierbei der Constraint-Graph.

Im Folgenden wird aufgezeigt, dass PCSP ein NP-vollständiges Problem ist. Dazu wird eine Reduktion des *Maximum Satisfiability Problems* (MAX SAT) auf PCSP angegeben. Das MAX SAT Problem ist, eine Belegung der boolschen Variablen einer aussagelogischen Formel in konjunktiver Normalform zu finden, sodass die Anzahl der erfüllten Klauseln maximal ist. Eine Klausel ist erfüllt, falls mindestens eines ihrer Literale erfüllt ist. Grundlagen der Mathematischen Logik sind beispielweise [9] zu entnehmen.

Definition 3.2.
Maximum Satisfiability Problem
INSTANZ: Aussagelogische Formel ϕ in konjunktiver Normalform mit
Variablenmenge $\{x_1, \ldots, x_n\}$ und Klauselmenge $C = \{c_1, \ldots, c_m\}$,
positives, ganzzahliges $k \leq |C|$
FRAGE: Existiert eine Belegung der Variablen, sodass mindestens k Klauseln erfüllt sind?

Satz 3.3.
Das Partial Constraint Satisfaction Problem ist NP-vollständig.
Beweis: Nach [5] (Cook, 1971) ist das Erfüllbarkeitsproblem der Aussagenlogik, *Satisfiability Problem* (SAT), NP-vollständig. Dieses Problem enthält die Frage, ob die Variablen einer gegebenen aussagenlogischen Formel so belegt werden können, dass die Formel wahr ist. SAT ist genau dann erfülltbar, wenn MAX SAT mit $k := m$ erfüllbar ist. Somit ist leicht einzusehen, dass MAX SAT ebenso NP-vollständig ist. Um die Aussage zu beweisen, wird eine Reduktion von MAX SAT auf

PCSP angegeben. Es wird also ein beliebig gegebener Fall von MAX SAT in polynomieller Zeit in einen Fall des PCSP in einer derartigen Weise transformiert, dass die Lösung des PCSPs (ja oder nein) gleich der Lösung des MAX SATs ist.
Der zugehörige Constraint-Graph wird wie folgt gebildet: Für jede Klausel c_i sei ein Knoten v_{c_i} eingeführt sowie für jede Variable x_j einen Knoten v_{x_j}. Die Wertemengen von v_{c_i} enthalten die Literale der Klausel c_i. Die Wertemengen von v_{x_j} ist gegeben durch $\{\textit{true},\textit{false}\}$. Es gebe eine Kante zwischen dem Knoten v_{c_i} und dem Knoten v_{x_j} genau dann, wenn $x_j \in c_i$ oder $\neg x_j \in c_i$. Für $x_j \in c_i$ ist der Penalty-Wert der Kombination $(x_j,\textit{false})$ gleich Eins. Analog ist für $\neg x_j \in c_i$ der Penalty-Wert der Kombination $(\neg x_j, \textit{true})$ gleich Eins. Alle anderen Penalty-Wert sind gleich Null. Das PCSP ist mit K erfüllbar genau dann, wenn das zugehörigen MAX SAT mit $k := m - K$ erfüllbar ist. Des Weiteren ist eine Lösung des MAX SATs gegeben durch die Belegungen der Knoten des PCSPs, die mit den Variablen korrespondieren. Es folgt die Behauptung: Das Partial Constraint Satisfaction Problem ist NP-vollständig. □

3.2 PCS Integer Linear Programm für das MI-FAP

Um das Partial Constraint Satisfaction Problem als ganzzahliges lineares Programm zu modellieren, sei für alle $v \in V$ und $d_v \in D_v$ die binäre Variable $y(v, d_v)$ definiert.

$$y(v, d_v) = \begin{cases} 1 & \text{falls } d_v \in D_v \text{ gewählt} \\ 0 & \text{sonst} \end{cases} \tag{3.1}$$

Weiter sei für alle $\{v, w\} \in E$, $d_v \in D_v$ und $d_w \in D_w$ die binäre Variable $z(v, d_v, w, d_w)$ eingeführt.

$$z(v, d_v, w, d_w) = \begin{cases} 1 & \text{falls } (d_v, d_w) \in D_v \times D_w \text{ gewählt} \\ 0 & \text{sonst} \end{cases} \tag{3.2}$$

Im Folgenden seien $q(v, d_v) := Q_v(d_v)$ und $p(v, d_v, w, d_w) := P_{\{v,w\}}(d_v, d_w)$ als neue Notation vereinbart. Dann lautet ein ganzzahliges lineares Programm für das Partial Constraint Satisfaction Problem und somit insbesondere für das MI-FAP wie folgt:

Partial Constraint Satisfaction Formulierung (PCS)

$$\textbf{min} \sum_{\{v,w\}\in E} \sum_{d_v \in D_v} \sum_{d_w \in D_w} p(v, d_v, w, d_w) \cdot z(v, d_v, w, d_w) + \sum_{v \in V} \sum_{d_v \in D_v} q(v, d_v) \cdot y(v, d_v) \tag{3.3}$$

$$\textbf{s.t.} \sum_{d_v \in D_v} y(v, d_v) = 1 \qquad \forall v \in V \tag{3.4}$$

$$\sum_{d_w \in D_w} z(v, d_v, w, d_w) = y(v, d_v) \qquad \forall \{v, w\} \in E, d_v \in D_v \tag{3.5}$$

$$y(v, d_v) \in \{0, 1\} \qquad \forall v \in V, d_v \in D_v \tag{3.6}$$

$$z(v, d_v, w, d_w) \in \{0, 1\} \qquad \forall \{v, w\} \in E, d_v \in D_v, d_w \in D_w \tag{3.7}$$

Die Nebenbedingungen (3.4) garantieren, dass jedem Knoten genau ein Wert zugewiesen wird. Außerdem stellen die Nebenbedingungen (3.5) sicher, dass die Kombination von zugewiesenen Werten zweier adjazenter Knoten konsistent mit dem zugewiesenen Wert der einzelnen Knoten

ist. Für eine Kante $\{v, w\} \in E$ bleiben im Falle von $x(v, d_v) = 0$ die Variablen $z(v, d_v, w, d_w)$ für alle $d_w \in D_w$ ungesetzt. Falls $x(v, d_v) = 1$, dann garantiert (3.5), dass genau eine $z(v, d_v, w, d_w)$-Variable gesetzt wird und zwar genau die mit $x(w, d_w) = 1$.

Den Daten der Probleminstanzen für das MI-FAP, wie sie in Abschnitt 2.4.1 vorgestellt wurden, ist noch nicht direkt der Input für die PCS-Formulierung zu entnehmen. Für jede Kante $\{v, w\} \in E$ werden die Penalty-Werte $p(v, d_v, w, d_w)$ eine sogenannte Penalty-Matrix bilden. Wie diese genau zustande kommen und wie sich die Probleminstanzen aufgrund ihrer Gegebenheiten um ungefähr die Hälfte reduzieren lassen, wird in dem folgenden Abschnitt erläutert.

3.2.1 Preprocessing der Instanzen

Die CALMA Instanzen besitzen spezielle Distanzbedingungen, siehe Abbildung 2.4. Neben den Bedingungen, die eine Mindestdistanz zwischen zwei zugewiesenen Frequenzen fordern, existieren auch Gleichheitsbedingungen. Diese modellieren, dass zwei Frequenzen mit einer festen Distanz δ den zugehörigen Knoten zugewiesen werden müssen. Wie schon in Abschnitt 2.4.1 erläutert, sind die Gleichheitsbedingungen auf jeden Fall zu erfüllen und können nicht im Gegenzug einer Strafe verletzt werden. Die geforderte Distanz δ ist bei allen Gleichheitsbedingungen die selbe und jeder Knoten gehört zu genau einer solchen Bedingung. Darüberhinaus ist der Frequenzwertebereich, siehe Abbildung 2.3, so geschaffen, dass zu jeder Frequenz nur genau eine passende Frequenz existiert. Hierbei sind zwei Frequenzen zueinander passend, wenn sie sich genau um die feste Distanz δ unterscheiden. Insgesamt führt diese Charakteristik dazu, dass die Instanzen um die Hälfte verkleinert werden können. Der Interferenz-Graph $G = (V, E)$ kann auf den Graphen G_{red} reduziert werden.

	Original		Reduziert	
Instanz	$\|V\|$	$\|E\|$	$\|V\|$	$\|E\|$
CELAR06	200	1.322	100	350
CELAR07	400	2.865	200	817
CELAR08	916	5.744	458	1.655
CELAR09	680	4.103	340	1.130
CELAR10	680	4.103	340	1.130
GRAPH05	200	1.134	100	416
GRAPH06	400	2.170	200	843
GRAPH07	400	2.170	200	843
GRAPH11	680	3.757	340	1.425
GRAPH12	680	4.017	340	1.256
GRAPH13	916	5.273	458	1.877

Tabelle 3.1: Originale und reduzierte Instanzen für MI-FAP

Je zwei Knoten, zu denen eine Gleichheitsbedingung existiert, werden zu einem neuen Knoten zusammengefasst. In der Notation von Abbildung 2.4 sind dies durchweg zwei aufeinander folgende Knoten. D.h. mit $k \in \mathbb{Z}$ wird aus den Knoten $2k+1$ und $2k+2$ der neue Knoten k gebildet. Dabei gilt für die Wertebereiche $D_{2k+1} = D_{2k+2}$ und somit für den neuen Knoten $D_k := D_{2k+1}$. Es sind

ebenso die Penalty-Werte für die Modifikation von schon vorgegebenen Frequenzen identisch. Aus der dann dem neuen Knoten zugewiesenen Frequenz ergibt sich die zweite Frequenz leicht durch die feste Distanz δ. Es werden implizit jedem Knoten ein Frequenzenpaar $(d_k, d_k \pm \delta) \in D_k^2$ zugewiesen. Das Vorzeichen von δ ist durch d_k eindeutig festgelegt. Insgesamt halbiert sich die Knotenmenge V. Die reduzierte Knotenmenge sei mit V_{red} bezeichnet.
Es gilt nun, die Kantenmenge inklusive der einzelnen Penalty-Werte zu der reduzierten Knotenmenge V_{red} aufzustellen. Diese sei mit E_{red} bezeichnet. Es existiert eine Kante $\{k, j\} \in E_{\text{red}}$ zwischen den zwei neuen Knoten $k \in V_{\text{red}}$ und $j \in V_{\text{red}}$, falls mindestens eine der Kanten $\{2k+1, 2j+1\}$, $\{2k+1, 2j+2\}$, $\{2k+2, 2j+1\}$, $\{2k+2, 2j+2\}$ in E enthalten ist. Darauf aufbauend wird für jede mögliche Kombination von zwei Frequenzpaaren $(d_k, d_k \pm \delta) \in D_k^2$, $(d_j, d_j \pm \delta) \in D_j^2$ der Penalty-Wert $p(k, d_k, j, d_j)$ berechnet. Dieses Vorgehen sei beispielhaft an der Instanz `CELAR06` dargestellt. Hierzu zeigen Abbildung 3.1 und Abbildung 3.2 Ausschnitte aus den Daten der Instanz.

```
13  14 D = 238 0
13  15 F >  59 1
13  16 C > 186 2
...
14  15 C > 186 2
14  16 F >  59 1
...
15  16 D = 238 0
...
```

Abbildung 3.1: `ctr.txt`

```
a1 = 1000
a2 =  100
a3 =   10
a4 =    1
...
```

Abbildung 3.2: `cst.txt`

Es werden die Knoten $13 \in V$ und $14 \in V$ sowie $15 \in V$ und $16 \in V$ zu je einem Knoten zusammengefasst. Dann lassen sich die Bedingungen mit folgendem Multigraphen darstellen:

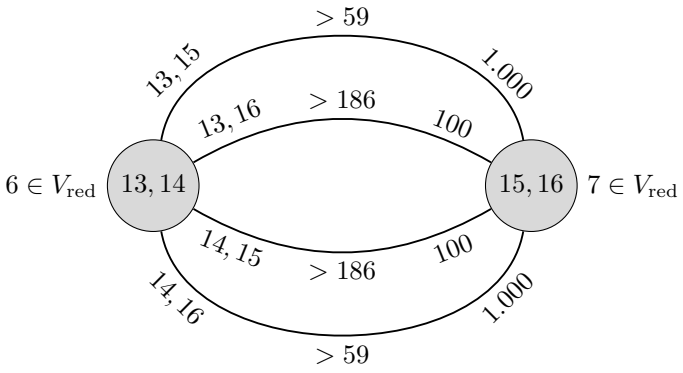

Abbildung 3.3: Bedingungen zwischen den zwei neuen Knoten $6, 7 \in V_{\text{red}}$

Jede Kante enthält die involvierten Knoten aus dem originalen Graphen, die Mindestdistanz der zugewiesenen Frequenzen und den Penalty-Wert für Nichteinhaltung dieser Bedingung. Für jede Kombination von Frequenzen kann nun der Penalty-Wert angegeben werden. Sei beispielsweise dem Knoten $6 \in V_{\text{red}}$ die Frequenz $16 \in D_6$ sowie dem Knoten $7 \in V_{\text{red}}$ ebenso die Frequenz $16 \in D_7$ zugewiesen. Die jeweils zweite Frequenz ist dann $16 + 238 = 254$. Es sind die Bedingung der oberen und unteren Kante verletzt. Für den Penalty-Wert gilt folglich: $p(v, d_v, w, d_w) = p(6, 16, 7, 16) = 1000 + 0 + 0 + 1000 = 2000$. Analog gilt: $p(6, 16, 7, 86) = 0 + 0 + 100 + 0 = 100$. Insgesamt bilden die $p(v, d_v, w, d_w)$ pro Kante $\{v, w\} \in E_{\text{red}}$ eine Matrix. Abbildung 4.4 zeigt die Penalty-Matrix der Kante $\{6, 7\}$ der Instanz `CELAR06`.

2000	2000	2000	2000	2100	100	100	100	100	100	100	200	200	200	200	200	200	200	200	200	200	200	100	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2000	2000	2000	2000	2000	2100	100	100	100	100	100	200	200	200	200	200	200	200	200	200	200	200	100	100	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2000	2000	2000	2000	2000	2000	2100	100	100	100	100	200	200	200	200	200	200	200	200	200	200	200	100	100	100	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2000	2000	2000	2000	2000	2000	2000	2100	100	100	100	200	200	200	200	200	200	200	200	200	200	200	100	100	100	100	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2100	2000	2000	2000	2000	2000	2000	2000	2100	100	100	200	200	200	200	200	200	200	200	200	200	200	100	100	100	100	100	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
100	2100	2000	2000	2000	2000	2000	2000	2000	2100	100	200	200	200	200	200	200	200	200	200	200	200	100	100	100	100	100	100	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
100	100	2100	2000	2000	2000	2000	2000	2000	2000	2100	200	200	200	200	200	200	200	200	200	200	200	100	100	100	100	100	100	100	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0
100	100	100	2100	2000	2000	2000	2000	2000	2000	2000	200	200	200	200	200	200	200	200	200	200	200	100	100	100	100	100	100	100	100	100	0	0	0	0	0	0	0	0	0	0	0	0	0
100	100	100	100	2100	2000	2000	2000	2000	2000	2000	200	200	200	200	200	200	200	200	200	200	200	100	100	100	100	100	100	100	100	100	100	0	1000	0	0	0	0	0	0	0	0	0	0
100	100	100	100	100	2100	2000	2000	2000	2000	2000	200	200	200	200	200	200	200	200	200	200	200	100	100	100	100	100	100	100	100	100	100	100	1000	1000	0	0	0	0	0	0	0	0	0
100	100	100	100	100	100	2100	2000	2000	2000	2000	200	200	200	200	200	200	200	200	200	200	200	100	100	100	100	100	100	100	100	100	100	100	1000	1000	1000	0	0	0	0	0	0	0	0
200	200	200	200	200	200	200	200	200	200	200	2000	2000	2000	2000	2100	100	100	100	100	100	100	0	0	0	0	0	0	0	0	0	0	0	100	100	0	0	0	0	0	0	0	0	0
200	200	200	200	200	200	200	200	200	200	200	2000	2000	2000	2000	2000	2100	100	100	100	100	100	0	0	0	0	0	0	0	0	0	0	0	100	100	100	0	0	0	0	0	0	0	0
200	200	200	200	200	200	200	200	200	200	200	2000	2000	2000	2000	2000	2000	2100	100	100	100	100	0	0	0	0	0	0	0	0	0	0	0	100	100	100	100	0	0	0	0	0	0	0
200	200	200	200	200	200	200	200	200	200	200	2000	2000	2000	2000	2000	2000	2000	2100	100	100	100	0	0	0	0	0	0	0	0	0	0	0	100	100	100	100	100	0	0	0	0	0	0
200	200	200	200	200	200	200	200	200	200	200	2100	2000	2000	2000	2000	2000	2000	2000	2100	100	100	0	0	0	0	0	0	0	0	0	0	0	100	100	100	100	100	100	0	0	0	0	0
200	200	200	200	200	200	200	200	200	200	200	100	2100	2000	2000	2000	2000	2000	2000	2000	2100	100	0	0	0	0	0	0	0	0	0	0	0	100	100	100	100	100	100	100	0	0	0	0
200	200	200	200	200	200	200	200	200	200	200	100	100	2100	2000	2000	2000	2000	2000	2000	2000	2100	0	0	0	0	0	0	0	0	0	0	0	100	100	100	100	100	100	100	100	0	0	0
200	200	200	200	200	200	200	200	200	200	200	100	100	100	2100	2000	2000	2000	2000	2000	2000	2000	0	0	0	0	0	0	0	0	0	0	0	100	100	100	100	100	100	100	100	100	0	0
200	200	200	200	200	200	200	200	200	200	200	100	100	100	100	2100	2000	2000	2000	2000	2000	2000	1000	0	0	0	0	0	0	0	0	0	0	100	100	100	100	100	100	100	100	100	100	0
200	200	200	200	200	200	200	200	200	200	200	100	100	100	100	100	2100	2000	2000	2000	2000	2000	1000	1000	0	0	0	0	0	0	0	0	0	100	100	100	100	100	100	100	100	100	100	100
200	200	200	200	200	200	200	200	200	200	200	100	100	100	100	100	100	2100	2000	2000	2000	2000	1000	1000	1000	0	0	0	0	0	0	0	0	100	100	100	100	100	100	100	100	100	100	100
100	100	100	100	100	100	100	100	100	100	100	0	0	0	0	0	0	0	0	1000	1000	1000	2000	2000	2000	2000	2100	100	100	100	100	100	100	200	200	200	200	200	200	200	200	200	200	200
100	100	100	100	100	100	100	100	100	100	100	0	0	0	0	0	0	0	0	0	1000	1000	2000	2000	2000	2000	2000	2100	100	100	100	100	100	200	200	200	200	200	200	200	200	200	200	200
0	100	100	100	100	100	100	100	100	100	100	0	0	0	0	0	0	0	0	0	0	1000	2000	2000	2000	2000	2000	2000	2100	100	100	100	100	200	200	200	200	200	200	200	200	200	200	200
0	0	100	100	100	100	100	100	100	100	100	0	0	0	0	0	0	0	0	0	0	0	2000	2000	2000	2000	2000	2000	2000	2100	100	100	100	200	200	200	200	200	200	200	200	200	200	200
0	0	0	100	100	100	100	100	100	100	100	0	0	0	0	0	0	0	0	0	0	0	2100	2000	2000	2000	2000	2000	2000	2000	2100	100	100	200	200	200	200	200	200	200	200	200	200	200
0	0	0	0	100	100	100	100	100	100	100	0	0	0	0	0	0	0	0	0	0	0	100	2100	2000	2000	2000	2000	2000	2000	2000	2100	100	200	200	200	200	200	200	200	200	200	200	200
0	0	0	0	0	100	100	100	100	100	100	0	0	0	0	0	0	0	0	0	0	0	100	100	2100	2000	2000	2000	2000	2000	2000	2000	2100	200	200	200	200	200	200	200	200	200	200	200
0	0	0	0	0	0	100	100	100	100	100	0	0	0	0	0	0	0	0	0	0	0	100	100	100	2100	2000	2000	2000	2000	2000	2000	2000	200	200	200	200	200	200	200	200	200	200	200
0	0	0	0	0	0	0	100	100	100	100	0	0	0	0	0	0	0	0	0	0	0	100	100	100	100	2100	2000	2000	2000	2000	2000	2000	200	200	200	200	200	200	200	200	200	200	200
0	0	0	0	0	0	0	0	100	100	100	0	0	0	0	0	0	0	0	0	0	0	100	100	100	100	100	2100	2000	2000	2000	2000	2000	200	200	200	200	200	200	200	200	200	200	200
0	0	0	0	0	0	0	0	0	100	100	0	0	0	0	0	0	0	0	0	0	0	100	100	100	100	100	100	2100	2000	2000	2000	2000	200	200	200	200	200	200	200	200	200	200	200
0	0	0	0	0	0	0	0	1000	1000	1000	100	100	100	100	100	100	100	100	100	100	100	200	200	200	200	200	200	200	200	200	200	200	2000	2000	2000	2000	2100	100	100	100	100	100	100
0	0	0	0	0	0	0	0	0	1000	1000	100	100	100	100	100	100	100	100	100	100	100	200	200	200	200	200	200	200	200	200	200	200	2000	2000	2000	2000	2000	2100	100	100	100	100	100
0	0	0	0	0	0	0	0	0	0	1000	0	100	100	100	100	100	100	100	100	100	100	200	200	200	200	200	200	200	200	200	200	200	2000	2000	2000	2000	2000	2000	2100	100	100	100	100
0	0	0	0	0	0	0	0	0	0	0	0	0	100	100	100	100	100	100	100	100	100	200	200	200	200	200	200	200	200	200	200	200	2000	2000	2000	2000	2000	2000	2000	2100	100	100	100
0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	100	100	100	100	100	100	100	200	200	200	200	200	200	200	200	200	200	200	2100	2000	2000	2000	2000	2000	2000	2000	2100	100	100
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	100	100	100	100	100	100	200	200	200	200	200	200	200	200	200	200	200	100	2100	2000	2000	2000	2000	2000	2000	2000	2100	100
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	100	100	100	100	100	200	200	200	200	200	200	200	200	200	200	200	100	100	2100	2000	2000	2000	2000	2000	2000	2000	2100
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	100	100	100	100	200	200	200	200	200	200	200	200	200	200	200	100	100	100	2100	2000	2000	2000	2000	2000	2000	2000
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	100	100	100	200	200	200	200	200	200	200	200	200	200	200	100	100	100	100	2100	2000	2000	2000	2000	2000	2000
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	100	100	200	200	200	200	200	200	200	200	200	200	200	100	100	100	100	100	2100	2000	2000	2000	2000	2000
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	100	200	200	200	200	200	200	200	200	200	200	200	100	100	100	100	100	100	2100	2000	2000	2000	2000

Abbildung 3.4: Penalty-Matrix der Kante $\{6, 7\}$ der Instanz CELAR06

In einer Penalty-Matrix ist der Wertebereich D_v des erst genannten Knotens v den Zeilen und der Wertebereich D_w des zwei genannten Knotens w den Spalten zugeordnet. Der oben berechnete Penalty-Wert $p(6, 16, 7, 16) = 2000$ ist in dem Eintrag der ersten Zeile und ersten Spalte zu finden. Weiter ist $p(6, 16, 7, 86) = 100$ in dem Eintrag der ersten Zeile und sechsten Spalte eingetragen, da die Frequenz 86 der sechstkleinste Wert in D_w ist.

Die Struktur der Penalty-Matrizen nimmt eine wichtige Stellung im Zuge der neu eingeführten Formulierung in Kapitel 4 ein. Für die PCS-Formulierung stellt die Penalty-Matrix zunächst lediglich eine Präsentation aller Penalty-Werte einer Kante dar.

Es müssen noch die in der PCS-Formulierung (3.3) - (3.7) mit $q(v, d_v)$ bezeichneten Kosten in Falle der Zuweisung der Frequenz $d_v \in D_v$ dem Knoten $v \in V$ auf den reduzierten Graphen $(V_{\text{red}}, E_{\text{red}})$ übertragen werden. Schon vorgegebene Frequenzen erfüllen natürlich die Gleichheitsbedingungen zwischen zwei Knoten $2k + 1, 2k + 2 \in V$. Die Kosten haben ihren Ursprung in der Modifikation dieser vorgegebenen Frequenzen und sind für die Knoten $2k + 1, 2k + 2 \in V$ identisch. Somit erhält ein neuer Knoten $k \in V_{\text{red}}$ in dem reduzierten Graphen die doppelten Kosten. Im Hinblick auf die Penalty-Matix können die $q(v, d_v)$ als übergeordnete Zeilen- bzw. Spaltenkosten angesehen werden. Wird die einer Zeile zugehörige Frequenz zugewiesen, werden die korrespondierenden Kosten $q(v, d_v)$ fällig.

Von nun an sei nicht mehr zwischen dem originalen Interferenz-Graphen und dem reduzierten unterschieden. Als Input für die PCS-Formulierung wird selbstverständlich der reduzierte Graph verwendet.

Vor dem Hintergrund von Zeilen- und Spaltenkosten einer Penalty-Matrix kann zusätzlich ein erweitertes Preprocessing auf den Instanzen durchgeführt werden. Dieses im Folgenden thematisierte Verfahren stellt vordergründig keine Vorteile für die PCS-Formulierung dar. Jedoch reduziert es die Anzahl der benötigten Variablen und Nebenbedingungen des in Kapitel 4 neu eingeführten Modells.

Das erweiterte Preprocessing besagt, dass pro Zeile und pro Spalte der minimale Eintrag subtrahiert und dem zugehörigen $q(v, d_v)$ Wert hinzuaddiert werden kann. Bildlich gesprochen, wird der minimale Zeilen- bzw. Spalteneintrag der Penalty-Matrix in die Kopfzeile bzw. -spalte mit dem $q(v, d_v)$ Werten gezogen. Dies ist zulässig, da bei Zuweisung einer Frequenz der minimale Penalty-Wert der zugehörigen Zeile bzw. Spalte auf jeden Fall fällig wird. Durch das Herausziehen der Minima können neue Null-Einträge in der Penalty-Matrix entstehen. Diese sind vorteilhaft für die *Penalty Block Formulierung* aus Kapitel 4. Da das erweiterte Preprocessing im Gegensatz zu dem einfachen Preprocessing nicht eindeutig ist, sei im Folgenden erläutert, wie dies im Rahmen dieser Arbeit umgesetzt wurde: Pro Penalty-Matrix wird zuerst das Minimum jeder Zeile herausgezogen und falls noch vorhanden wird anschließend das Minimum jeder Spalte herausgezogen.

Instanz		Optimum Wert	Optimum Zeit	Schranken nach 12 Std. duale	Schranken nach 12 Std. primale
CELAR06	Scip	–	–	4,00	5.191
	Cplex	–	–	4,38	6.491
CELAR07	Scip	–	–	*31.453,08	*30.113.660
	Cplex	–	–	36.279,00	61.987.500
CELAR08	Scip	–	–	48,32	4.797
	Cplex	–	–	48,49	2.709
CELAR09	Scip	15.571	1:08:34	–	–
	Cplex	15.571	0:00:52	–	–
CELAR10	Scip	31.516	0:00:57	–	–
	Cplex	31.516	0:00:11	–	–
GRAPH05	Scip	221	0:00:56	–	–
	Cplex	221	0:00:26	–	–
GRAPH06	Scip	4.123	0:11:42	–	–
	Cplex	4.123	0:11:42	–	–
GRAPH07	Scip	4.324	0:00:35	–	–
	Cplex	4.324	0:00:08	–	–
GRAPH11	Scip	–	–	*2.861,02	*221.495
	Cplex	–	–	2.861,02	12.509
GRAPH12	Scip	11.827	0:01:00	–	–
	Cplex	11.827	0:00:12	–	–
GRAPH13	Scip	–	–	9.477,48	417.539
	Cplex	–	–	0,00	793.308

Tabelle 3.2: Ergebnisse der Rechenstudie der PCS Formulierung ohne erweitertem Preprocessing

Instanz		Optimum Wert	Optimum Zeit	Schranken nach 12 Std. duale	Schranken nach 12 Std. primale
CELAR06	Scip	–	–	3,50	4.890
	Cplex	–	–	3,50	39.903
CELAR07	Scip	–	–	31.453,08	228.192.388
	Cplex	–	–	36.689,07	46.789.048
CELAR08	Scip	–	–	48,32	4.797
	Cplex	–	–	48,32	3.059
CELAR09	Scip	15.571	1:12:06	–	–
	Cplex	15.571	0:00:43	–	–
CELAR10	Scip	31.516	0:00:53	–	–
	Cplex	31.516	0:00:09	–	–
GRAPH05	Scip	221	0:00:40	–	–
	Cplex	221	0:00:14	–	–
GRAPH06	Scip	4.123	0:03:16	–	–
	Cplex	4.123	0:01:40	–	–
GRAPH07	Scip	4.324	0:00:37	–	–
	Cplex	4.324	0:00:08	–	–
GRAPH11	Scip	–	–	*2.861,02	*221.937
	Cplex	–	–	2.861,02	13.495
GRAPH12	Scip	11.827	0:00:53	–	–
	Cplex	11.827	0:00:08	–	–
GRAPH13	Scip	–	–	9.477,48	417.539
	Cplex	–	–	0,00	710.589

Tabelle 3.3: Ergebnisse der Rechnerstudie der PCS Formulierung mit erweitertem Preprocessing

3.2.2 Ergebnisse der Rechenstudie

Die Ergebnisse der Rechenstudie der Partial Constraint Satisfaction Formulierung sind in Tabelle 3.2 und Tabelle 3.3 aufgeführt. Neben dem Einsatz des einfachen Preprocessings wurden auch Berechnungen unter zusätzlicher Verwendung des erweiterten Preprocessings durchgeführt. In den Einstellungen des Solvers Scip wurde der Presolver deaktiviert, da dieser innerhalb mehrerer Stunden nicht über die nullte Presolverrunde hinwegkam.

Ein * deutet darauf hin, dass diese Rechnung vorzeitig durch den Solver aufgrund eines Speicherfehlers abgebrochen wurde. Genauer Auslöser war hierbei meist der voll ausgenutzte RAM von 12 GB.

Wie schon vermutet, verbessert die Verwendung des erweiterten Preprocessings die Ergebnisse nicht merklich. Neben stärkeren, meist schlechteren, Abweichungen bei den Instanzen `CELAR06`, `CELAR07` und `CELAR08` sind die Ergebnisse sonst nahezu gleich. Der Vorteil des erweiterten Preprocessings wird sich erst in Kapitel 4 im Zuge der *Penalty Block Formulierung* offenbaren.

Es konnten sechs der elf Instanzen optimal gelöst werden, dabei lag die Lösungszeit meist im Sekundenbereich. Ausnahme hierbei ist die Instanz `CELAR09` unter Verwendung von Scip. Dort waren etwa eine Stunde nötig, um das Optimum zu finden. Zu den restlichen Probleminstanzen können nach 12 Stunden Rechnung lediglich untere und obere Schranken angegeben werden. Mit Blick auf die bekannten optimalen Werte der Instanzen, siehe Tabelle 4.8, sind die obere Schranke 4.890 der Instanz `CELAR06`, die untere Schranke 2.861,02 der Instanz `GRAPH11` sowie die untere Schranke 9.477,48 am ehesten befriedigend.

4 Die Penalty Block Formulierung

4.1 Über die Struktur der Daten zu einer neuen Formulierung

Bei dem Betrachten einer Penalty-Matrix, wie beispielsweise der in Abbildung 4.1 der Instanz CELAR06 nach erweitertem Preprocessing, fallen sofort zwei Dinge auf: Erstens, dass die Matrix eine gewisse Blockstruktur besitzt und zweitens, dass nur wenige verschiedene Penalty-Werte in der Matrix enthalten sind. Es wird zunächst auf Letzteres genauer eingegangen, die Blockstruktur wird im späteren Verlauf analysiert. Falls nicht explizit angegeben, liegen diesem Kapitel die Instanzen nach erweitertem Preprocessing zugrunde.
In der Penalty-Matrix in Abbildung 4.1 sind neben der 0 nur noch 100, 999, 1000, 1001, 1100, 1101, 2000, 2001, 2100 und 2101 als Penalty-Werte vorhanden. Wie sich herausstellen wird, ist der Penalty-Wert 0 nicht zu beachten, sodass nur zehn zu berücksichtigende, verschiedenwertige Einträge in dieser 36×36 Penalty-Matrix vorhanden sind.
Eine Analyse aller Penalty-Matrizen der Instanz CELAR06 zeigt, dass durchschnittlich nur sechs bis sieben verschiedenwertige Einträge pro Kante beachtet werden müssen. In der folgenden Tabelle 4.1 sind die minimalen, maximalen und durchschnittlichen Anzahlen für alle Instanzen eingetragen.

			# Penalty-Werte / Kante		
Instanz	$\|V\|$	$\|E\|$	min.	max.	∅
CELAR06	100	350	1	19	6,4
CELAR07	200	817	1	22	6,0
CELAR08	458	1.655	1	13	4,9
CELAR09	340	1.130	1	25	6,2
CELAR10	340	1.130	1	18	5,2
GRAPH05	100	416	1	23	4,2
GRAPH06	200	843	1	27	4,0
GRAPH07	200	843	1	27	4,0
GRAPH11	340	1.425	1	26	4,1
GRAPH12	340	1.256	1	27	5,4
GRAPH13	458	1.877	1	26	4,4

Tabelle 4.1: Analyse der Anzahl verschiedenwertiger Penalty-Werte pro Kante

In der PCS Formulierung aus Kapitel 3 wurde für jeden einzelnen Penalty-Matrix-Eintrag eine binäre Variable $z(v, d_v, w, d_w)$ eingeführt. Dies führt neben einer immens großen Anzahl an Variablen auch dazu, dass viele Variablen mit dem gleichen Faktor in der Zielfunktion vorhanden sind, obwohl nur eine dieser binären Variablen auf 1 gesetzt werden kann.

2000	2000	2000	2000	2000	2000	2000	2000	2000	2100	2100	2100	2100	1100	1100	1100	1100	1100	0	0	0	0	0	0	0	0	0	1000	1000	1000	1000	1000	999	999	999	999
2000	2000	2000	2000	2000	2000	2000	2000	2000	2100	2100	2100	2100	2100	1100	1100	1100	1100	0	0	0	0	0	0	0	0	0	1000	1000	1000	1000	1000	1000	999	999	999
2000	2000	2000	2000	2000	2000	2000	2000	2000	2100	2100	2100	2100	2100	2100	1100	1100	1100	0	0	0	0	0	0	0	0	0	1000	1000	1000	1000	1000	1000	1000	999	999
2000	2000	2000	2000	2000	2000	2000	2000	2000	2100	2100	2100	2100	2100	2100	2100	1100	1100	0	0	0	0	0	0	0	0	0	1000	1000	1000	1000	1000	1000	1000	1000	999
2000	2000	2000	2000	2000	2000	2000	2000	2000	1100	2100	2100	2100	2100	2100	2100	2100	1100	0	0	0	0	0	0	0	0	0	1000	1000	1000	1000	1000	1000	1000	1000	1000
2000	2000	2000	2000	2000	2000	2000	2000	2000	1100	1100	2100	2100	2100	2100	2100	2100	2100	0	0	0	0	0	0	0	0	0	1000	1000	1000	1000	1000	1000	1000	1000	1000
2000	2000	2000	2000	2000	2000	2000	2000	2000	1100	1100	1100	2100	2100	2100	2100	2100	2100	0	0	0	0	0	0	0	0	0	1000	1000	1000	1000	1000	1000	1000	1000	1000
2100	2000	2000	2000	2000	2000	2000	2000	2000	1100	1100	1100	1100	2100	2100	2100	2100	2100	0	0	0	0	0	0	0	0	0	1000	1000	1000	1000	1000	1000	1000	1000	1000
2100	2100	2000	2000	2000	2000	2000	2000	2000	1100	1100	1100	1100	1100	2100	2100	2100	2100	0	0	0	0	0	0	0	0	0	1000	1000	1000	1000	1000	1000	1000	1000	1000
2101	2101	2101	2101	1101	1101	1101	1101	1101	2001	2001	2001	2001	2001	2001	2001	2101	2101	1001	1001	1001	1001	1001	1000	1000	1000	1000	0	0	0	0	0	0	0	0	0
2101	2101	2101	2101	2101	1101	1101	1101	1101	2001	2001	2001	2001	2001	2001	2001	2001	2101	1001	1001	1001	1001	1001	1001	1000	1000	1000	0	0	0	0	0	0	0	0	0
2101	2101	2101	2101	2101	2101	1101	1101	1101	2001	2001	2001	2001	2001	2001	2001	2001	2001	1001	1001	1001	1001	1001	1001	1001	1000	1000	100	0	0	0	0	0	0	0	0
2101	2101	2101	2101	2101	2101	2101	1101	1101	2001	2001	2001	2001	2001	2001	2001	2001	2001	1001	1001	1001	1001	1001	1001	1001	1001	1000	100	100	0	0	0	0	0	0	0
1101	2101	2101	2101	2101	2101	2101	2101	1101	2001	2001	2001	2001	2001	2001	2001	2001	2001	1001	1001	1001	1001	1001	1001	1001	1001	1001	100	100	100	0	0	0	0	0	0
1101	1101	2101	2101	2101	2101	2101	2101	2101	2001	2001	2001	2001	2001	2001	2001	2001	2001	1001	1001	1001	1001	1001	1001	1001	1001	1001	100	100	100	100	0	0	0	0	0
1101	1101	1101	2101	2101	2101	2101	2101	2101	2001	2001	2001	2001	2001	2001	2001	2001	2001	1001	1001	1001	1001	1001	1001	1001	1001	1001	100	100	100	100	100	0	0	0	0
1101	1101	1101	1101	2101	2101	2101	2101	2101	2001	2001	2001	2001	2001	2001	2001	2001	2001	1001	1001	1001	1001	1001	1001	1001	1001	1001	100	100	100	100	100	100	0	0	0
1101	1101	1101	1101	1101	2101	2101	2101	2101	2001	2001	2001	2001	2001	2001	2001	2001	2001	1001	1001	1001	1001	1001	1001	1001	1001	1001	100	100	100	100	100	100	100	0	0
0	0	100	100	100	100	100	100	100	1001	1001	1001	1001	1001	1001	1001	1001	1001	2001	2001	2001	2001	2001	2001	2001	2001	2001	2101	2101	2101	2101	1101	1101	1101	1101	1101
0	0	0	100	100	100	100	100	100	1001	1001	1001	1001	1001	1001	1001	1001	1001	2001	2001	2001	2001	2001	2001	2001	2001	2001	2101	2101	2101	2101	2101	1101	1101	1101	1101
0	0	0	0	100	100	100	100	100	1001	1001	1001	1001	1001	1001	1001	1001	1001	2001	2001	2001	2001	2001	2001	2001	2001	2001	2101	2101	2101	2101	2101	2101	1101	1101	1101
0	0	0	0	0	100	100	100	100	1001	1001	1001	1001	1001	1001	1001	1001	1001	2001	2001	2001	2001	2001	2001	2001	2001	2001	2101	2101	2101	2101	2101	2101	2101	1101	1101
0	0	0	0	0	0	100	100	100	1001	1001	1001	1001	1001	1001	1001	1001	1001	2001	2001	2001	2001	2001	2001	2001	2001	2001	1101	2101	2101	2101	2101	2101	2101	2101	1101
0	0	0	0	0	0	0	100	100	1000	1001	1001	1001	1001	1001	1001	1001	1001	2001	2001	2001	2001	2001	2001	2001	2001	2001	1101	1101	2101	2101	2101	2101	2101	2101	2101
0	0	0	0	0	0	0	0	100	1000	1000	1001	1001	1001	1001	1001	1001	1001	2001	2001	2001	2001	2001	2001	2001	2001	2001	1101	1101	1101	2101	2101	2101	2101	2101	2101
0	0	0	0	0	0	0	0	0	1000	1000	1000	1001	1001	1001	1001	1001	1001	2101	2001	2001	2001	2001	2001	2001	2001	2001	1101	1101	1101	1101	2101	2101	2101	2101	2101
0	0	0	0	0	0	0	0	0	1000	1000	1000	1000	1001	1001	1001	1001	1001	2101	2101	2001	2001	2001	2001	2001	2001	2001	1101	1101	1101	1101	1101	2101	2101	2101	2101
1000	1000	1000	1000	1000	1000	1000	1000	1000	0	0	0	0	0	0	0	0	0	2100	2100	2100	2100	1100	1100	1100	1100	1100	2000	2000	2000	2000	2000	2000	2000	2100	2100
1000	1000	1000	1000	1000	1000	1000	1000	1000	0	0	0	0	0	0	0	0	0	2100	2100	2100	2100	2100	1100	1100	1100	1100	2000	2000	2000	2000	2000	2000	2000	2000	2100
1000	1000	1000	1000	1000	1000	1000	1000	1000	0	0	0	0	0	0	0	0	0	2100	2100	2100	2100	2100	2100	1100	1100	1100	2000	2000	2000	2000	2000	2000	2000	2000	2000
1000	1000	1000	1000	1000	1000	1000	1000	1000	0	0	0	0	0	0	0	0	0	2100	2100	2100	2100	2100	2100	2100	1100	1100	2000	2000	2000	2000	2000	2000	2000	2000	2000
1000	1000	1000	1000	1000	1000	1000	1000	1000	0	0	0	0	0	0	0	0	0	1100	2100	2100	2100	2100	2100	2100	2100	1100	2000	2000	2000	2000	2000	2000	2000	2000	2000
999	1000	1000	1000	1000	1000	1000	1000	1000	0	0	0	0	0	0	0	0	0	1100	1100	2100	2100	2100	2100	2100	2100	2100	2000	2000	2000	2000	2000	2000	2000	2000	2000
999	999	1000	1000	1000	1000	1000	1000	1000	0	0	0	0	0	0	0	0	0	1100	1100	1100	2100	2100	2100	2100	2100	2100	2000	2000	2000	2000	2000	2000	2000	2000	2000
999	999	999	1000	1000	1000	1000	1000	1000	0	0	0	0	0	0	0	0	0	1100	1100	1100	1100	2100	2100	2100	2100	2100	2000	2000	2000	2000	2000	2000	2000	2000	2000
999	999	999	999	1000	1000	1000	1000	1000	0	0	0	0	0	0	0	0	0	1100	1100	1100	1100	1100	2100	2100	2100	2100	2000	2000	2000	2000	2000	2000	2000	2000	2000

Abbildung 4.1: Penalty-Matrix der Kante $\{399, 400\}$ der Instanz CELAR06

Es ist die Idee, diesen Umstand zu verbessern, indem ausgenutzt wird, dass pro Kante nur wenige verschiedene Penalty-Werte zu beachten sind. Dadurch kann die Anzahl der Variablen gehörig verringert werden. Die folgende Erste Reformulierung modelliert diesen Gedanken genauer.

4.2 Erste Reformulierung

Pro Kante $\{v, w\} \in E$ bilden für $d_v \in D_v$ und $d_w \in D_w$ die Parameter $p(v, d_v, w, d_w)$ eine Penalty-Matrix. Mithilfe dieser Werte wird nun für eine Kante $\{v, w\} \in E$ folgende neue Parametermenge definiert, die Ausgangspunkt für die Definition neuer Variablen ist.

$$P_{\{v,w\}} := \Big\{ p : p(v, d_v, w, d_w) = p \text{ mit } d_v \in D_v, d_w \in D_w \text{ und } p \neq 0 \Big\} \tag{4.1}$$

Die Menge $P_{\{v,w\}}$ enthält somit alle möglichen Penalty-Werte der Kante $\{v, w\} \in E$ mit $0 \notin P_{\{v,w\}}$. Die durchschnittliche Mächtigkeit dieser Mengen ist für jede Instanz in der letzten Spalte von Tabelle 4.1 abzulesen. Es ist offensichtlich, dass keine dieser Mengen leer ist, denn sonst enthielte die zugehörige Penalty-Matrix nur Nullen und wäre so nicht zu beachten.
Zur Bequemlichkeit wird des Weiteren folgende Notation eingeführt:

$$(D_v \times D_w)_p := \Big\{ (d_v, d_w) \in (D_v \times D_w) : p(v, d_v, w, d_w) = p \Big\} \tag{4.2}$$

Die Erste Reformulierung für das MI-FAP wird modelliert mit den aus der PCS Formulierung bekannten $y(v, d_v)$-Variablen und hier neu einzuführenden binären $x(v, w, p)$-Variablen.
Für alle $\{v, w\} \in E$ und $p \in P_{\{v,w\}}$ seien

$$x(v, w, p) = \begin{cases} 1 & \text{falls } (d_v, d_w) \in D_v \times D_w \text{ mit } p(v, d_v, w, d_w) = p \text{ gewählt} \\ 0 & \text{sonst} \end{cases} \tag{4.3}$$

Das Programm der Ersten Reformulierung lautet dann wie folgt:

Erste Reformulierung (1st)

$$\textbf{min} \sum_{\{v,w\} \in E} \sum_{p \in P_{\{v,w\}}} p \cdot x(v, w, p) + \sum_{v \in V} \sum_{d_v \in D_v} q(v, dv) \cdot y(v, d_v) \tag{4.4}$$

$$\textbf{s.t.} \sum_{d_v \in D_v} y(v, d_v) = 1 \qquad \forall v \in V \tag{4.5}$$

$$y(v, d_v) + y(w, d_w) \leq 1 + x(v, w, p) \qquad \begin{array}{l} \forall \{v, w\} \in E, p \in P_{\{v,w\}}, \\ (d_v, d_w) \in (D_v \times D_w)_p \end{array} \tag{4.6}$$

$$y(v, d_v) \in \{0, 1\} \qquad \forall v \in V, d_v \in D_v \tag{4.7}$$

$$x(v, w, p) \in \{0, 1\} \qquad \forall \{v, w\} \in E, p \in P_{\{v,w\}} \tag{4.8}$$

Die Nebenbedingungen (4.5) modellieren wie bei der PCS Formulierung die Tatsache, dass pro Knoten genau eine Frequenz zugeordnet wird. Des Weiteren stellen die Nebenbedingungen (4.6) sicher, dass mit der Erfüllung von

$$y(v, d_v) = 1 \text{ und } y(v, d_w) = 1 \text{ für } (d_v, d_w) \in (D_v \times D_w)_p \Longrightarrow x(v, w, p) = 1 \tag{4.9}$$

die Variablen $x(v, w, p)$ modelliert sind. Dabei ist zu beachten, dass die $\Leftarrow$-Richtung von (4.9) nicht zu modellieren ist, da alle $p \in P_{\{v,w\}}$ als Zielfunktionsfaktoren von $x(v, w, p)$ positiv sind. Somit wird bei dem Minimierungsproblem nie grundlos ein $x(v, w, p) = 1$ gesetzt, es sei denn, dies wird wie in (4.9) beschrieben erzwungen. Für eine Implementierung dieses Modells ist noch zu erwähnen, dass die x-Variablen, aufgrund der binären y-Variablen, kontinuierlich zwischen 0 und 1 definiert werden können.

Ein erster Vergleich zwischen der Ersten Reformulierung und der PCS Formulierung zeigt, dass die Variablenanzahl stark verringert werden konnte. Jedoch hat sich dabei die Anzahl der Nebenbedingungen drastisch erhöht. Grund dafür ist, dass für jeden einzelnen Penalty-Matrix-Eintrag ungleich Null eine Nebenbedingung der Form (4.6) existiert. Tabelle 4.2 zeigt die genaue Anzahl an Variablen und Nebenbedingungen der einzelnen Instanzen.

Instanz		Anzahl Variablen			Anzahl Nebenbedingungen
		Summe	kontinuierliche	binäre	
CELAR06	PCS	585.914	581.904	4.010	28.628
	1st	6.234	2.224	4.010	379.607
CELAR07	PCS	1.331.048	1.323.072	7.976	65.928
	1st	12.916	4.940	7.976	837.270
CELAR08	PCS	2.518.664	2.500.564	18.100	128.886
	1st	26.247	8.147	18.100	1.568.643
CELAR09	PCS	1.793.484	1.780.056	13.428	89.970
	1st	20.487	7.059	13.428	1.148.459
CELAR10	PCS	1.793.484	1.780.056	13.428	89.970
	1st	19.299	5.871	13.428	1.144.440
GRAPH05	PCS	574.628	570.920	3.708	30.986
	1st	5.440	1.732	3.708	324.479
GRAPH06	PCS	1.201.638	1.194.096	7.542	63.846
	1st	10.875	3.333	7.542	671.049
GRAPH07	PCS	1.146.426	1.139.096	7.330	62.290
	1st	10.707	3.377	7.330	634.270
GRAPH11	PCS	2.032.792	2.019.972	12.820	107.688
	1st	18.700	5.880	12.820	1.169.944
GRAPH12	PCS	1.805.190	1.792.408	12.782	95.408
	1st	19.548	6.766	12.782	1.165.353
GRAPH13	PCS	2.798.400	2.780.812	17.588	145.034
	1st	25.755	8.167	17.588	1.693.257

Tabelle 4.2: Anzahl der Variablen und Nebenbedingungen der PCS Formulierung (PCS) und der Ersten Reformulierung (1st)

Wie sich in den weiteren Abschnitten dieses Kapitels herausstellen wird, lässt sich die Anzahl der Nebenbedingungen noch stark reduzieren. Die geschieht unter anderem dadurch, dass die schon angesprochene Blockstruktur der Penalty-Matrizen ausgenutzt wird.

4.2.1 Theoretischer Vergleich mit PCS Formulierung

Zunächst wird der Vergleich zwischen der Partial Constraint Satisfaction Formulierung aus Kapitel 3 und der vorgestellen Ersten Reformulierung aus theoretischer Sicht vertieft. Dazu wird im Folgenden das Konzept der guten Formulierung mithilfe von Definitionen und Beispielen eingeführt. Diese sind zum größten Teil [17] entnommen worden.

Definition 4.1.
$P = \{x \in \mathbb{R}^n : Ax \leq b\}$ ist als Teilmenge von $\mathbb{R}^n$ und beschrieben durch eine endliche Menge linearer Ungleichungen ein *Polyeder*.

Definition 4.2.
Ein Polyeder $P \subseteq \mathbb{R}^{n+p}$ ist eine *Formulierung* für eine Menge $X \subseteq \mathbb{N}^n \times \mathbb{R}^p$, wenn $X = P \cap (\mathbb{N}^n \times \mathbb{R}^p)$ gilt.

Um die Definition einer Formulierung schnell einzusehen, werden zwei Beispiele angeführt.

Beispiel.
Sei $c \in \mathbb{R}$. Dann ist $P = \{(x, y) \in \mathbb{R}^2 : x \leq cy, x \geq 0, 0 \leq y \leq 1\}$ eine Formulierung für die Menge $X = \{(0, 0), (x, 1)$ für $0 \leq x \leq c\}$.

Beispiel.
Nicht für jede Menge existiert eine Formulierung. Dies ist bei der Menge $X = \{(0, 0), (1, 0), (2, 0), (0, 1), (2, 1), (0, 2), (1, 2), (2, 2)\}$ der Fall, jedoch ist es leicht für die Menge $X \cup \{(1, 1)\}$ eine Formulierung zu finden.

Es ist offensichtlich, dass für eine Menge mehrere Formulierungen existieren können. Eine weitere ist die konvexe Hülle dieser Menge, die, wie sich herausstellen wird, eine besondere Formulierung ist.

Definition 4.3.
Für eine gegebene Menge $X \subseteq \mathbb{R}^n$ ist die *konvexe Hülle* von X definiert durch:

$$conv(X) = \left\{x : x = \sum_{i=1}^{t} \lambda_i x^i, \sum_{i=1}^{t} \lambda_i = 1, \lambda_i \geq 0 \text{ für alle endlichen } \{x^1, \ldots, x^t\} \subseteq X\right\}$$

Folgerung 4.4.
$conv(X)$ ist ein Polyeder und somit eine Formulierung für die Menge X. Außerdem liegen alle Extrempunkte von $conv(X)$ in X.

Die konvexe Hülle $conv(X)$ ist die beste (ideale) Formulierung für die Menge X, denn ein ganzzahliges lineares Programm $\max\{cx : x \in X\}$ kann äquivalent durch das lineare Programm $\max\{cx : x \in conv(X)\}$ ersetzt und somit "leicht" gelöst werden. Leider ist dies häufig nur eine theoretische Aussage, da in den meisten Fällen exponentiell viele Nebenbedingungen nötig sind, um $conv(X)$ zu beschreiben.

Bei zwei gegebenen Formulierungen P_1 und P_2 für eine Menge X eröffnet sich die Frage, welche der beiden die Bessere ist. Da $conv(X)$ als die beste Formulierung die Eigenschaft $X \subseteq conv(X) \subseteq P$ für alle Formulierungen P besitzt, liegt folgende Definition nahe:

Definition 4.5.
Seien P_1 und P_2 zwei Formulierungen für $X \subseteq \mathbb{R}^n$. $P_1 \subseteq \mathbb{R}^n$ ist eine *bessere Formulierung* als $P_2 \subseteq \mathbb{R}^n$, falls $P_1 \subset P_2$ gilt.

Anhand dieser Definition wird nun das folgende Beispiel betrachtet, in dem zwei Formulierungen für das gleiche Problem miteinander verglichen werden.

Beispiel.
Das *Uncapacitated Facility Location Problem* (UFL) besagt, dass eine Menge von Kunden $M = \{1, \cdots, m\}$ von einer Menge von Depots $N = \{1, \cdots, n\}$ mit minimalen Kosten beliefert werden sollen. Es entstehen Kosten für die Nutzung eines Depots $j \in N$ in Höhe von f_j und Kosten für den Warentransport zwischen Kunde $i \in M$ und Depot $j \in N$ in Höhe von c_{ij}. Jeder Kunde soll von genau einem Depot bedient werden.
Für die Modellierung des Problems werden folgende Variablen eingeführt:

$$x_{ij} = \begin{cases} 1 & \text{falls Depot } j \text{ Kunde } i \text{ beliefert} \\ 0 & \text{sonst} \end{cases} \qquad y_j = \begin{cases} 1 & \text{falls Depot } j \text{ genutzt} \\ 0 & \text{sonst} \end{cases}$$

Dann lautet ein ganzzahliges lineares Programm für das Uncapacitated Facility Location Problem:

$$\begin{aligned} \min \ & \sum_{i \in M} \sum_{j \in N} c_{ij} x_{ij} + \sum_{j \in N} f_j y_j & \\ \text{s.t.} \ & \sum_{j \in N} x_{ij} = 1 & \forall i \in M \\ & \sum_{i \in M} x_{ij} \leq |M| y_j & \forall j \in N \\ & x_{ij} \in \{0,1\}, y_j \in \{0,1\} & \forall i \in M, j \in N \end{aligned}$$

Somit ist

$$UFL_1 = \left\{ (x,y) : \sum_{j \in N} x_{ij} = 1, \sum_{i \in M} x_{ij} \leq |M| y_j, 0 \leq x_{ij} \leq 1, 0 \leq y_j \leq 1 \right\}$$

eine Formulierung für die Menge aller zulässigen und ganzzahligen Punkte für das Problem. Es ist leicht einzusehen, dass

$$UFL_2 = \left\{ (x,y) : \sum_{j \in N} x_{ij} = 1, x_{ij} \leq y_j, 0 \leq x_{ij} \leq 1, 0 \leq y_j \leq 1 \right\}$$

eine weitere Formulierung ist.

Satz 4.6.
UFL_2 ist eine bessere Formulierung als UFL_1.
Beweis:
Sei $(\tilde{x}, \tilde{y}) \in UFL_2$. Somit gilt $\sum_{i \in M} \tilde{x}_{ij} \leq \sum_{i \in M} \tilde{y}_j = |M| \tilde{y}_j$ für alle $j \in N$. Damit ist $(\tilde{x}, \tilde{y}) \in UFL_1$ erfüllt und $UFL_2 \subseteq UFL_1$ wurde nachgewiesen. Es bleibt zu zeigen, dass $UFL_2 \neq UFL_1$ gilt. Dafür wird ein $(\tilde{x}, \tilde{y}) \in UFL_1 \setminus UFL_2$ konstruiert. Es wird angenommen, dass $m = kn$ mit $k \geq 2$ ganzzahlig gilt. Definiere $(\tilde{x}, \tilde{y})$ wie folgt:

$$\tilde{x}_{ij} = \begin{cases} 1 & \text{für } i = k(j-1)+1, \cdots, k(j-1)+k \\ 0 & \text{sonst} \end{cases} \qquad \tilde{y}_j = \frac{k}{m} \qquad \forall j \in N$$

Es gilt $(\tilde{x}, \tilde{y}) \in UFL_1$, jedoch $(\tilde{x}, \tilde{y}) \notin UFL_2$ und es ist $UFL_2 \subset UFL_1$ gezeigt und somit die Behauptung bewiesen. □

Die zwei Formulierungen, die für das Uncapacitated Facility Location Problem betrachtet wurden, sind beide über dem gleichen Raum definiert. Das heißt, beide enthalten genau die gleichen Variablen und nur die Nebenbedingungen wurden modifiziert. Jedoch ist es oft sinnvoll, Formulierungen zu finden, die andere oder mehr Variablen enthalten, also über unterschiedlichen Räumen definiert sind. Dabei ist nicht sofort klar, wie zwei solche Formulierungen sinnvoll verglichen werden können. Dies ermöglicht zunächst der Begriff der Projektion.
Dazu sei angenommen, dass eine erste Formulierung $\min\{cx : x \in P \cap \mathbb{Z}^n\}$ mit $P \subseteq \mathbb{R}^n$ und eine zweite $\min\{cx : (x, w) \in Q \cap (\mathbb{Z}^n \times \mathbb{R}^p)\}$ mit $Q \subseteq \mathbb{R}^n \times \mathbb{R}^p$ existiert.

Definition 4.7.
Die *Projektion* von einem Polyeder $Q \subseteq (\mathbb{R}^n \times \mathbb{R}^p)$ auf den Unterraum $\mathbb{R}^n$ ist definiert durch

$$proj_x Q := \{x \in \mathbb{R}^n : (x, w) \in Q \text{ für ein } w \in \mathbb{R}^p\}.$$

Somit ist $proj_x Q$ durch Projektion von Q auf den Raum $\mathbb{R}^n$ der originalen x-Variablen entstanden und dies erlaubt es, Q mit anderen Formulierungen $P \subseteq \mathbb{R}^n$ zu vergleichen.

Unter diesem Gesichtspunkt wird nun die Erste Reformulierung mit der PCS Formulierung aus Kapitel 3 verglichen. Um diese im Einzelnen vor Augen zu haben, werden sie an dieser Stelle wiederholt.

Partial Constraint Satisfaction Formulierung

$$PCS := \Big\{(y, z) : \sum_{d_v \in D_v} y(v, d_v) = 1 \quad \forall v \in V \tag{4.10}$$

$$\sum_{d_w \in D_w} z(v, d_v, w, d_w) = y(v, d_v) \quad \forall \{v, w\} \in E, d_v \in D_v \tag{4.11}$$

$$0 \leq y(v, d_v) \leq 1 \tag{4.12}$$

$$0 \leq z(v, d_v, w, d_w) \leq 1\Big\} \tag{4.13}$$

Erste Reformulierung (1st)

$$R_1 := \Big\{(y, x) : \sum_{d_v \in D_v} y(v, d_v) = 1 \quad \forall v \in V \tag{4.14}$$

$$y(v, d_v) + y(w, d_w) \leq 1 + x(v, w, p) \quad \begin{matrix} \forall \{v, w\} \in E, p \in P_{\{v,w\}}, \\ (d_v, d_w) \in (D_v \times D_w)_p \end{matrix} \tag{4.15}$$

$$0 \leq y(v, d_v) \leq 1 \tag{4.16}$$

$$0 \leq x(v, w, p) \leq 1\Big\} \tag{4.17}$$

mit $P_{\{v,w\}} := \Big\{p : p(v, d_v, w, d_w) = p \text{ mit } d_v \in D_v, d_w \in D_w \text{ und } p \neq 0\Big\}$
und $(D_v \times D_w)_p := \Big\{(d_v, d_w) \in D_v \times D_w : p(v, d_v, w, d_w) = p\Big\}$.

Beide Formulierungen enthalten die gleichen y-Variablen. Somit werden jeweils die Projektionen auf diese Variablen miteinander verglichen.

Satz 4.8.
Es gilt $proj_y PCS = proj_y R_1$.
Beweis:
Es wird zuerst gezeigt, dass $proj_y PCS \subseteq proj_y R_1$ gilt. Sei dazu $(y, z) \in PCS$ und definiere $\tilde{y} := y$ sowie

$$\tilde{x}(v, w, p) := \sum_{(d_v, d_w) \in (D_v \times D_w)_p} z(v, d_v, w, d_w) \quad \forall \{v, w\} \in E, p \in P_{\{v,w\}}. \tag{4.18}$$

Im Folgenden wird gezeigt, dass $(\tilde{y}, \tilde{x}) \in R_1$ gilt. Dabei ist $\sum \tilde{y}(v, d_v) = 1$ offensichtlich erfüllt, da $\sum y(v, d_v) = 1$ gilt. Es bleibt zu zeigen, dass für alle $\{v, w\} \in E, p \in P_{\{v,w\}}$ und $(d_v, d_w) \in (D_v \times D_w)_p$ die Ungleichung

$$\tilde{y}(v, d_v) + \tilde{y}(v, d_v) \leq 1 + \tilde{x}(v, w, p)$$

erfüllt ist. Seien dazu $\{v, w\} \in E, p \in P_{\{v,w\}}$ und $(d_v, d_w) \in (D_v \times D_w)_p$ beliebig. Dann gilt:

$$\begin{aligned}
\tilde{y}(v, d_v) + \tilde{y}(w, d_w) &= y(v, d_v) + y(w, d_w) \\
&= \sum_{d_w \in D_w} z(v, d_v, w, d_w) + \sum_{d_v \in D_v} z(v, d_v, w, d_w) && (4.19) \\
&\leq 1 + z(v, d_v, w, d_w) && (4.20) \\
&\leq 1 + \sum_{(d_v, d_w) \in (D_v \times D_w)_p} z(v, d_v, w, d_w) \\
&= 1 + \tilde{x}(v, w, p)
\end{aligned}$$

Dabei wird im Ausdruck (4.19) die Variable $z(v, d_v, w, d_w)$ zweimal aufsummiert. Diese kann in der nächsten Zeile nicht ein zweites Mal mit gegen 1 abgeschätzt werden, somit verbleibt sie einmal.
Für die Abschätzung in (4.20) wird verwendet, dass $\sum_{d_v, d_w} z(v, d_v, w, d_w) = 1$ in PCS gilt. Dies ist schnell einzusehen, indem die Gleichungen (4.11) für jedes $d_v \in D_v$ aufsummiert werden und die Gleichung (4.10) ausgenutzt wird:

$$\sum_{d_v \in D_v} \sum_{d_w \in D_w} z(v, d_v, w, d_w) = \sum_{d_v \in D_v} y(v, d_v) = 1$$

Es wurde gezeigt, dass $proj_y PCS \subseteq proj_y R_1$ gilt. Um die Behauptung vollständig zu beweisen, wird noch die fehlende Inklusion

$$proj_y R_1 = \{y : (y, x) \in R_1 \text{ für ein } x\} \subseteq \{y : (y, z) \in PCS \text{ für ein } z\} = proj_y PCS$$

nachgewiesen. Dabei ist zu zeigen, dass zu einem beliebigen $(y, x) \in R_1$ ein passendes z existiert, sodass $(y, z) \in PCS$ erfüllt ist. Dieses Problem der z-Konstruktion lässt sich pro Kante $\{v, w\} \in E$ betrachten, da jedes $z(v, d_v, w, d_w)$ genau einer Kante zugeordnet ist.
Seien also eine Kante $\{v, w\} \in E$ mit $n := |D_v|$ und $m := |D_w|$ sowie sämtliche Belegungen der $0 \leq y(v, d_v), y(w, d_w) \leq 1$ mit der Eigenschaft

$$\sum_{d_v \in D_v} y(v, d_v) = 1 \quad \text{und} \quad \sum_{d_w \in D_w} y(w, d_w) = 1$$

vorgegeben. Es ist nun eine Zuweisung der $0 \leq z(v, d_v, w, d_w) \leq 1$ gesucht, sodass

$$\sum_{d_w \in D_w} z(v, d_v, w, d_w) = y(v, d_v) \quad \forall d_v \in D_v \tag{4.21}$$

$$\sum_{d_v \in D_v} z(v, d_v, w, d_w) = y(w, d_w) \quad \forall d_w \in D_w \tag{4.22}$$

erfüllt werden. (cont.)

Um diese Problemstellung besser zu verstehen und dadurch eine Lösung zu finden, wird der Beweis des Satzes 4.8 durch Überlegungen und Aussagen unterbrochen und im Anschluss daran zu Ende geführt.

Satz 4.9.
Das Problem der z-Konstruktion in dem Beweis von Satz 4.8 ist äquivalent dazu, eine Matrix $Z \in \mathbb{R}_{\geq 0}^{n \times m}$ mit Einträgen (z_{ij}) zu finden, sodass die einzelnen Zeilensummen gleich den Werten der $y(v, d_v^i)$ und die einzelnen Spaltensummen gleich den Werten der $y(w, d_w^j)$ sind. Mit der Zuweisung $z(v, d_v^i, w, d_w^j) := (z_{ij})$ ist eine zulässige Zuweisung der $z(v, d_v, w, d_w)$ für diese Kante gefunden.
Beweis:
Um die Aussage einzusehen wird folgende Abbildung betrachtet:

	$\sum_{\text{Spalte 1}} = y(w, d_w^1)$	$\sum_{\text{Spalte 2}} = y(w, d_w^2)$	$\cdots$	$\sum_{\text{Spalte } m} = y(w, d_w^m)$	$\sum_i y(w, d_w^i) = 1$
$\sum_{\text{Zeile 1}} = y(v, d_v^1)$	$z(v, d_v^1, w, d_w^1)$	$z(v, d_v^1, w, d_w^2)$	$\cdots$	$z(v, d_v^1, w, d_w^m)$	
$\sum_{\text{Zeile 2}} = y(v, d_v^2)$	$z(v, d_v^2, w, d_w^1)$	$z(v, d_v^2, w, d_w^2)$	$\cdots$	$z(v, d_v^2, w, d_w^m)$	
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	
$\sum_{\text{Zeile } n} = y(v, d_v^n)$	$z(v, d_v^n, w, d_w^1)$	$z(v, d_v^n, w, d_w^2)$	$\cdots$	$z(v, d_v^n, w, d_w^m)$	
$\sum_i y(v, d_v^i) = 1$					

Abbildung 4.2: Zusammenhang zwischen den y- und z-Variablen in der PCS Formulierung

Die Belegung der y-Variablen sind an der Kopfzeile und -spalte vorgegeben. Die hellgrau hinterlegte Matrix $Z \in \mathbb{R}_{\geq 0}^{n \times m}$ wird gesucht. Dabei wird gefordert, dass für alle $i = 1, \ldots, n$ die Summe der i-ten Zeile dem Wert von $y(v, d_v^i)$ entspricht. Dies verlangt genau die Gleichung (4.21). Analoges für die Spaltensummen fordert (4.22). Da die Summe aller Matrixeinträge der Summe aller Zeilen- oder Spaltensummen also 1 entspricht, ist auch $0 \leq z(v, d_v, w, d_w) \leq 1$ erfüllt. □

Die in Satz 4.9 beschriebene Problemstellung ist ein Spezialfall des in der diskreten Optimierung bekannten *Transportproblems* [2, 8, 18], welches wie folgt definiert ist:

Definition 4.10.
Ein Produkt soll von Ausgangsorten $A_1, \ldots, A_m$ zu Bestimmungsorten $B_1, \ldots, B_n$ transportiert werden. In den Ausgangsorten A_i ist ein bestimmter positiver Vorrat a_i vorhanden und in den Bestimmungsorten B_j besteht ein gewisser positiver Bedarf b_j an diesem Produkt. Dabei stimmen Gesamtvorrat und Gesamtbedarf überein. Eine Belieferung darf nur direkt von einem Ausgangsort an einen Bestimmungsort erfolgen. Von den $m \cdot n$ bestehenden Transportrouten sind

die Kosten pro transportierte Mengeneinheit c_{ij} für $i = 1, \ldots, m$ und $j = 1, \ldots, n$ bekannt und konstant. Das *Transportproblem* enthält das Ziel, einen Transportplan aufzustellen, charakterisiert durch die transportierten Mengen x_{ij} von A_i nach B_j, der alle Anforderungen erfüllt und dessen Gesamtkosten minimal sind.

Ein lineares Programm für das Transportproblem lautet wie folgt:

$$\begin{aligned}
\min \quad & \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} \\
\text{s.t.} \quad & \sum_{j=1}^{n} x_{ij} = a_i && i = 1, \ldots, m && (4.23)\\
& \sum_{i=1}^{m} x_{ij} = b_j && j = 1, \ldots, n && (4.24)\\
& x_{ij} \geq 0 && i = 1, \ldots, m,\ j = 1, \ldots, n \\
\text{wobei} \quad & \sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j \text{ und } a_i, b_j \geq 0
\end{aligned}$$

Übertragen auf Abbildung 4.2 stellen die Werte $y(w, d_w^i)$ der Kopfzeile den Vorrat a_i der Ausgangsorte A_i da. Analog geben die Werte $y(v, d_v^j)$ der Kopfspalte den Bedarf b_j der Zielorte B_j an. Die gesuchte Belegung von $z(v, d_v^j, w, d_w^i)$ gibt die transportierte Menge von A_i nach B_j eines zulässigen Transportplanes an. Passend dazu ist (4.23) mit (4.21) und (4.24) mit (4.22) vergleichbar. Für den Beweis von Satz 4.8 sind Transportkosten irrelevant, da zunächst nur eine zulässige Lösung für dieses Transportproblem gesucht wird. Diese liefert beispielsweise die Verwendung der sogenannten *Nordwesteckenregel* [18]. Dabei wird, ausgehend von einer Tabelle wie in Abbildung 4.2, rekursiv der obere linke Eintrag mit dem Minimum aus dem entsprechenden Vorrat und Bedarf gesetzt. Mit dem nachfolgenden Beispiel wird dieses Verfahren schnell klar.

Beispiel.
Seien $n := 3$ und $m := 2$ sowie $a := (1\ 2\ 3)^T$ und $b := (5\ 1)^T$.

	5	1
1		
2		
3		

→

	4	1
0	1	0
2		
3		

→

	2	1
0	1	0
0	2	0
3		

→

	0	1
0	1	0
0	2	0
1	2	

→

	0	0
0	1	0
0	2	0
0	2	1

Die letzte hellgrau hinterlegte Matrix ist die Ausgabe $Z \in \mathbb{R}^{3\times 2}$ dieses Verfahrens mit passenden Zeilen- und Spaltensummen und stellt somit einen zulässigen Transportplan für das zugehörige Transportproblem dar.

Satz 4.7. (cont.)
Es gilt $proj_y PCS = proj_y R_1$.
Beweis:
Wie in Satz 4.9 gezeigt wurde, ist das Problem der z-Konstruktion äquivalent dazu, eine Matrix mit bestimmten Zeilen- und Spaltensummen bzw. eine zulässige Lösung des zugehörigen Transportproblems zu finden. Die Nordwesteckenregel liefert beispielsweise die geforderte Matrix. Somit ist gezeigt, dass mit vorgegebener Belegung der y-Variablen für jede Kante $\{v, w\} \in E$

eine Zuweisung der $z(v, d_v, w, d_w)$ konstruiert werden kann, sodass insgesamt (4.10) und (4.11) erfüllt werden. Folglich gilt dann $(y, z) \in PCS$ und die Aussage von Satz 4.8 ist vollständig nachgewiesen. □

Es drängt sich der Gedanke auf, dass nicht aufgrund der Gültigkeit von $proj_y PCS = proj_y R_1$ zufriedenstellend gefolgert werden kann, dass beide Formulierungen gleich gut sind. Durch die Projektionen auf die gemeinsamen Variablen verlieren die beiden Formulierungen in Gestalt ihrer zusätzlichen Variablen eine wichtige Charakteristik. Die letztendlich im Vergleich größtenteils vernachlässigten Variablen repräsentieren gerade die unterschiedlichen Ideen der beiden Formulierungen.

4.2.2 Erweiterter theoretischer Vergleich mit PCS Formulierung

Um die vergleichende Analyse zwischen der PCS Formulierung und der Ersten Reformulierung zu intensivieren, wird im Folgenden das Konzept *der guten Formulierung unter erweiterter Ansicht* eingeführt. Dabei sollen nicht mehr nur die Polyeder der Formulierungen verglichen werden, sondern auch die Zielfunktionen miteinbezogen werden. Wenn beide Formulierungen neben gleichen Variablen noch zusätzlich eigene Variablen enthalten, unterscheiden sich meist auch die Zielfunktionen beider Formulierungen. Dieser neue Ansatz soll es nun ermöglichen, zwei derartig beschaffene Formulierungen angemessener zu vergleichen. Hierbei ist wichtig, eine Formulierung nicht nur als ein Polyeder anzusehen, sondern verstärkt als relaxierte Beschreibung einer Menge von (gemischt-)ganzzahligen Punkten. Bekanntlich liefert eine LP-Relaxierung im Falle eines Minimierungsproblems eine untere Schranke für den optimalen Zielfunktionswert für das eigentliche (gemischt-)ganzzahlige Problem. Im Sinne dieses Gedankens ist eine Formulierung die Bessere, wenn sie eine bessere untere Schranke für das Ausgangsproblem liefert.

Für die folgenden Definitionen seien P_1 und P_2 Formulierungen über unterschiedlichen Räumen für ein Optimierungsproblem, sodass

$$\min\Big\{cx + d_1 y : (x, y) \in P_1 \cap (\mathbb{Z}^n \times \mathbb{R}^p)\Big\} \text{ mit } P_1 \subseteq (\mathbb{R}^n \times \mathbb{R}^p)$$
$$\min\Big\{cx + d_2 z : (x, z) \in P_2 \cap (\mathbb{Z}^n \times \mathbb{R}^q)\Big\} \text{ mit } P_2 \subseteq (\mathbb{R}^n \times \mathbb{R}^q).$$

Definition 4.11.
P_1 und P_2 sind *unter erweiterter Ansicht gleich gute Formulierungen*, falls

$$proj_x P_1 = proj_x P_2$$
und für alle $(x, y) \in P_1$ ex. ein $(\tilde{x}, \tilde{z}) \in P_2$ mit $\tilde{x} = x$, sodass gilt: $cx + d_1 y \geq c\tilde{x} + d_2 \tilde{z}$ (4.25)
und für alle $(x, z) \in P_2$ ex. ein $(\tilde{x}, \tilde{y}) \in P_1$ mit $\tilde{x} = x$, sodass gilt: $cx + d_2 z \geq c\tilde{x} + d_1 \tilde{y}$ (4.26)

erfüllt sind.

Die Aussagen (4.25) und (4.26) fordern, dass zu jeder Variablenbelegung der einen Formulierung mindestens eine in den gemeinsamen Variablen übereinstimmende Belegung in der anderen Formulierung zu existieren hat, die einen nicht echt größeren Zielfunktionswert hat.
Falls nur eine der beiden Aussagen erfüllt ist, beispielsweise nur (4.25), dann ist P_1 unter erweiterter Ansicht eine nicht echt schlechtere Formulierung als P_2 oder anders gesagt, eine mindestens

gleich gute. Falls zusätzlich die Negation der anderen Aussage erfüllt ist, gibt es unter erweiterter Ansicht eine bessere Formulierung, wie folgende Definition aufzeigt.

Definition 4.12.
P_1 ist *unter erweiterter Ansicht* eine *bessere Formulierung* als P_2, falls

$$proj_x P_1 \subseteq proj_x P_2$$

und für alle $(x, y) \in P_1$ ex. ein $(\tilde{x}, \tilde{z}) \in P_2$ mit $\tilde{x} = x$, sodass gilt: $cx + d_1 y \geq c\tilde{x} + d_2\tilde{z}$ (4.27)

und es ex. ein $(\tilde{x}, \tilde{z}) \in P_2$, sodass für alle $(x, y) \in P_1$ mit $x = \tilde{x}$ gilt: $c\tilde{x} + d_2\tilde{z} < cx + d_1 y$ (4.28)

erfüllt sind.

Dabei verlangt Aussage (4.27), dass zu jeder Variablenbelegung der besseren Formulierung mindestens eine in den gemeinsamen Variablen übereinstimmende Belegung in der anderen Formulierung existiert, die einen nicht echt größeren Zielfunktionswert hat. Des Weiteren fordert Aussage (4.28), dass die schlechtere Formulierung eine Variablenbelegung zu besitzen hat, sodass jede in den gemeinsamen Variablen übereinstimmende Belegungen der besseren Formulierung einen echt größeren Zielfunktionswert hat.

Diese erweiterte Theorie macht es möglich, eine stärkere Aussage bei der Gegenüberstellung der PCS Formulierung und der Ersten Reformulierung zu erhalten.
Zur Erinnerung: Zu der PCS Formulierung (4.10) - (4.13) gehört die Zielfunktion

$$z_{PCS}(y, z) := \underbrace{\sum_{\{v,w\}\in E} \sum_{d_v \in D_v} \sum_{d_w \in D_w} p(v, d_v, w, d_w) \cdot z(v, d_v, w, d_w)}_{=: z^1_{PCS}(z)} + \sum_{v \in V} \sum_{d_v \in D_v} q(v, d_v) \cdot y(v, d_v)$$

sowie zu der Ersten Reformulierung (4.14) - (4.17) die Zielfunktion

$$z_{R_1}(y, x) := \underbrace{\sum_{\{v,w\}\in E} \sum_{p \in P_{\{v,w\}}} p \cdot x(v, w, p)}_{=: z^1_{R_1}(x)} + \sum_{v \in V} \sum_{d_v \in D_v} q(v, d_v) \cdot y(v, d_v).$$

Satz 4.13.
Die PCS Formulierung ist unter erweiterter Ansicht eine mindestens gleich gute Formulierung wie die Erste Reformulierung R_1. In Abhängigkeit der betrachteten Instanz ist die PCS Formulierung in vielen Fällen unter erweiterter Ansicht eine bessere Formulierung als die Erste Reformulierung R_1.
Beweis:
In Satz 4.8 ist nachgeweisen, dass $proj_y PCS = proj_y R_1$ gilt. Somit ist nach Definition 4.11 noch zu zeigen:

für alle $(y, z) \in PCS$ ex. ein $(\tilde{y}, \tilde{x}) \in R_1$ mit $\tilde{y} = y$, sodass gilt: $z_{PCS}(y, z) \geq z_{R_1}(\tilde{y}, \tilde{x})$ (4.29)

es ex. ein $(\tilde{y}, \tilde{x}) \in R_1$, sodass für alle $(y, z) \in PCS$ mit $y = \tilde{y}$ gilt: $z_{R_1}(\tilde{y}, \tilde{x}) < z_{PCS}(y, z)$ (4.30)

Die Behauptung sagt, dass (4.29) stets und (4.30) in Abhängigkeit der betrachteten Instanz gilt.
Zu jeder Variablenbelegung $(y, z) \in PCS$ erfüllt $(\tilde{y}, \tilde{x}) \in R_1$ mit $\tilde{y} := y$ und

$$\tilde{x}(v, w, p) := \sum_{(d_v, d_w) \in (D_v \times D_w)_p} z(v, d_v, w, d_w) \qquad \forall \{v, w\} \in W, p \in P_{\{v,w\}}$$

die Aussage (4.29). Im Beweis von Satz 4.8 wurde bereits gezeigt, dass diese Belegung für die Formulierung R_1 zulässig ist. Außerdem gilt für jedes dieser Paare Gleichheit bei den Zielfunktionen. Zunächst ist klar, dass der hintere Teil der beiden Zielfunktionen mit den y-Variablen gleich ist. Für den Rest der Zielfunktionen gilt

$$\begin{aligned}
z^1_{R_1}(\tilde{x}) &= \sum_{\{v,w\}\in E} \sum_{p\in P_{\{v,w\}}} p \cdot \tilde{x}(v,w,p) \\
&= \sum_{\{v,w\}\in E} \sum_{p\in P_{\{v,w\}}} p \sum_{(d_v,d_w)\in(D_v\times D_w)_p} z(v,d_v,w,d_w) \\
&= \sum_{\{v,w\}\in E} \sum_{p\in P_{\{v,w\}}} \sum_{(d_v,d_w)\in(D_v\times D_w)_p} p(v,d_v,w,d_w)\cdot z(v,d_v,w,d_w) \\
&= \sum_{\{v,w\}\in E} \sum_{d_v\in D_v} \sum_{d_w\in D_w} p(v,d_v,w,d_w)\cdot z(v,d_v,w,d_w) \\
&= z^1_{PCS}(z)
\end{aligned}$$

und somit auch Gleichheit. Dies impliziert, dass die PCS Formulierung unter erweiterter Ansicht eine mindestens gleich gute Formulierung wie die Erste Reformulierung ist.
Für den zweiten Teil der Behauptung, also (4.30), wird o.B.d.A. angenommen, dass $|D_v| \geq 2$ für alle $v \in V$ gilt. Ansonsten wäre für $|D_v| = 1$ eine Frequenzzuweisung trivial durchführbar. Es ist eine Variablenbelegung $(\tilde{y},\tilde{x}) \in R_1$ gesucht, dessen Zielfunktionswert $z_{R_1}(\tilde{y},\tilde{x})$ kleiner als der Zielfunktionswert $z_{PCS}(y,z)$ einer beliebigen Variablenbelegungen $(y,z) \in PCS$ mit $y = \tilde{y}$ ist. Dazu seien definiert:

$$\tilde{y}(v,d_v) := \frac{1}{|D_v|} \quad \forall v \in V, d_v \in D_v \tag{4.31}$$

$$\tilde{x}(v,w,p) := 0 \quad \forall \{v,w\} \in E, p \in P_{\{v,w\}} \tag{4.32}$$

Dass hierbei $(\tilde{y},\tilde{x}) \in R_1$ gilt ist schnell einzusehen: Die Gleichungen (4.14) sind offensichtlich erfüllt und die Ungleichungen (4.15) gelten, da jeweils $\frac{1}{|D_v|} + \frac{1}{|D_w|} \leq 1$ mit der obrigen Annahme erfüllt ist. Für diese Variablenbelegung ist der erste Teil der Zielfunktion $z^1_{R_1}(\tilde{x})$ offenbar gleich Null. Nun hängt es von der betrachteten Instanz ab, ob eine Variablenbelegung $(y,z) \in PCS$ mit $y = \tilde{y}$ existiert, sodass

$$z^1_{PCS}(z) = \sum_{\{v,w\}\in E} \sum_{d_v\in D_v} \sum_{d_w\in D_w} p(v,d_v,w,d_w)\cdot z(v,d_v,w,d_w) = 0$$

gilt. Dies ist nur der Fall, wenn bei $p(v,d_v,w,d_w) \neq 0$ für die Belegung $z(v,d_v,w,d_w) = 0$ gilt. Mit Blick auf die zugehöhrigen Transportprobleme (pro Kante eins) bedeutet dies, dass jedes Transportproblem mit Kosten von Null gelöst werden muss. Es wird deutlich, dass eine solche Variablenbelegung für eine beliebige Instanz nicht zwingend möglich ist. Nach dem Preprocessing der Daten aus Abschnitt 3.2.1 existiert in jeder Zeile und jeder Spalte jeder Penalty-Matix mindestens ein Eintrag 0. Das heißt, für Kanten $\{v,w\} \in E$ mit $|D_v| = |D_w|$ ist für die in (4.31) definierte Belegung der y-Variablen das zugehörige Transportproblem mit Kosten von Null zulässig lösbar. Für Kanten $\{v,w\} \in E$ mit $|D_v| \neq |D_w|$ wird hierfür meist mehr als ein Penalty-Wert von 0 pro Zeile und pro Spalte benötigt und dies ist im Allgemeinen nicht gewährleistet.
Somit ist in Abhängigkeit der betrachteten Instanz die PCS Formulierung in vielen Fällen unter erweiterter Ansicht eine bessere Formulierung als die Erste Reformulierung R_1. □

4.2.3 Gültige Ungleichungen

Wie schon kurz in Abschnitt 4.2.1 angesprochen, ist bekannt, dass ein ganzzahliges lineares Programm $\max\{cx : x \in X\}$ mit $X = \{x : Ax \leq b, x \in \mathbb{Z}_+^n\}$ in ein lineares Programm $\max\{cx : x \in conv(X)\}$ umformuliert werden kann. Dies ist leider meist nur in der Theorie möglich, da es zum Teil sehr schwer ist $conv(X)$ angemessen zu beschreiben. Das Konzept der *gültigen Ungleichungen* ist ein effektiver Ansatz, die konvexe Hülle $conv(X)$ einer Menge zu approximieren. Gültige Ungleichungen sind der erste Schritt zu facettendefinierenden Ungleichungen.

Definition 4.14.
Eine Ungleichung $\pi x \leq \pi_0$ ist eine *gültige Ungleichung* für $X \in \mathbb{R}^n$, falls $\pi x \leq \pi_0$ für alle $x \in X$ gilt.

Falls $X = \{x \in \mathbb{Z}^n : Ax \leq b\}$ und $conv(X) = \{x \in \mathbb{R}^n : \tilde{A}x \leq \tilde{b}\}$ sind $a^i x \leq b_i$ und $\tilde{a}^i x \leq \tilde{b}_i$ offensichtlich gültige Ungleichungen.

Es werden im Folgenden gültige Ungleichungen für die Erste Reformulierung entwickelt. Dazu wird die *Chvátal-Gomory Prozedur* verwendet.

Definition 4.15.
Mit der *Chvátal-Gomory Prozedur* lassen sich gültige Ungleichungen für die Menge $X = P \cap \mathbb{Z}^n$ mit $P = \{x \in \mathbb{R}_+^n : Ax \leq b\}$, A eine $m \times n$ Matrix mit Spalten $a_1, \ldots, a_n$ und $u \in \mathbb{R}_+^m$ konstruieren.

(i) Die Ungleichung
$$\sum_{j=1}^{n} ua_j x_j \leq ub$$
ist gültig für P mit $u \geq 0$ und $\sum_{j=1}^{n} a_j x_j \leq b$.

(ii) Die Ungleichung
$$\sum_{j=1}^{n} \lfloor ua_j \rfloor x_j \leq ub$$
ist gültig für P mit $x \geq 0$.

(iii) Die Ungleichung
$$\sum_{j=1}^{n} \lfloor ua_j \rfloor x_j \leq \lfloor ub \rfloor$$
ist gültig für P mit x ganzzahlig und somit $\sum_{j=1}^{n} \lfloor ua_j \rfloor x_j$ ganzzahlig.

Weitere Ausführungen zu diesem Thema sowie der Beweis für die nachfolgende Aussage ist [17] zu entnehmen.

Satz 4.16.
Jede gültige Ungleichung für X kann durch eine endliche Anzahl von Anwendungen der Chvátal-Gomory Prozedur konstruiert werden.

Mit Blick auf gültige Ungleichungen für die Erste Reformulierung seien $\{v, w, u\} \subseteq V$ eine Menge von Knoten, die in dem Interferenz-Graphen (V, E) eine Clique bilden. Abbildung 4.3 zeigt diese Situation, wobei aktuell den Knoten die Frequenzen f, g sowie h mit $z(v, f, w, g) = p_1$,

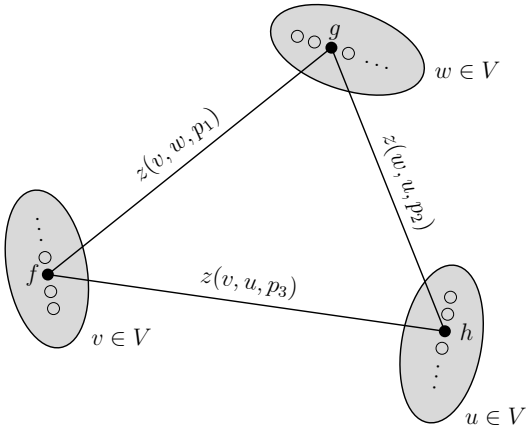

Abbildung 4.3: 3-Knoten-Clique in Interferenz-Graph (V, E)

$z(w, g, u, h) = p_2$ und $z(v, f, u, h) = p_3$ zugewiesen sind. Die zugehörigen z-Variablen sind in der Abbildung an den Kanten angegeben.
Zu dieser Situation enthält die Erste Reformulierung folgende drei Ungleichungen:

$$\begin{aligned} y(v, f) + y(w, g) &\leq 1 + z(v, w, p_1) \\ y(w, g) + y(u, h) &\leq 1 + z(w, u, p_2) \\ y(v, f) + y(u, h) &\leq 1 + z(v, u, p_3) \end{aligned}$$

Aufsummiert mit $u = \frac{1}{2}$ ergibt sich mit Definition 4.15 (i) die folgende gültige Ungleichung:

$$\begin{aligned} & y(v, f) + y(w, g) + y(u, h) \leq \frac{3}{2} + \frac{1}{2}z(v, w, p_1) + \frac{1}{2}z(w, u, p_2) + \frac{1}{2}z(v, u, p_3) \\ \iff & y(v, f) + y(w, g) + y(u, h) - \frac{1}{2}z(v, w, p_1) - \frac{1}{2}z(w, u, p_2) - \frac{1}{2}z(v, u, p_3) \leq \frac{3}{2} \end{aligned}$$

Da $y, z \geq 0$ gilt, ist mit Definition 4.15 (ii) folgende Ungleichung weiter gültig:

$$y(v, f) + y(w, g) + y(u, h) - z(v, w, p_1) - z(w, u, p_2) - z(v, u, p_3) \leq \frac{3}{2}$$

Schließlich ergibt sich mit Definition 4.15 (iii) und $y, z \in \{0, 1\}$ die folgende gültige Ungleichung für die Menge der zulässigen Lösungen der Ersten Reformulierung:

$$y(v, f) + y(w, g) + y(u, h) \leq 1 + z(v, w, p_1) + z(w, u, p_2) + z(v, u, p_3)$$

Folgerung 4.17.
Seien $\{v, w, u\} \subseteq V$ eine Clique in dem Interferenz-Graphen (V, E), dann ist

$$y(v, d_v) + y(w, d_w) + y(u, d_u) \leq 1 + z(v, w, p_1) + z(w, u, p_2) + z(v, u, p_3)$$

mit $d_v \in D_v, d_w \in D_w, d_u \in D_u$ sowie $z(v, d_v, w, d_w) = p_1$, $z(w, d_w, u, d_u) = p_2$ und $z(v, d_v, u, d_u) = p_3$ eine gültige Ungleichung für die Erste Reformulierung.

4.3 Zweite Reformulierung

Wie schon angesprochen lassen sich die Anzahl der Nebenbedingungen der Ersten Reformulierung noch deutlich verringern. Genauer gesagt, wird nun im Zuge der Zweiten Reformulierung die Modellierung der $x(v, w, p)$-Variablen modifiziert, indem die Nebenbedingungen (4.6) den Daten angepasst werden.
Die Nebenbedingungen (4.6) der Ersten Reformulierung modellieren, dass wenn zwei adjazente Knoten $v \in V$ und $w \in V$ jeweils eine Frequenz $d_v \in D_v$ bzw. $d_w \in D_w$ zugewiesen ist, also $y(v, d_v) = 1$ und $y(v, d_w) = 1$ gelten, die mit $p = p(v, d_v, w, d_w)$ zugehörige $x(v, w, p)$-Variable gesetzt wird. Dieser Zusammenhang wird in Nebenbedingung (4.6) pro Kante $\{v, w\} \in E$ für jede $(d_v, d_w) \in D_v \times D_w$ Kombination (außer mit $p(v, d_v, w, d_w) = 0$) modelliert.
Um die Idee der Zweiten Reformulierung nachzuvollziehen, soll erneut die Penalty-Matrix der Kante $\{399, 400\}$ der Instanz `CELAR06` in Abbildung 4.1 vor Augen geführt werden. Zur Erinnerung: In der Penalty-Matrix sind die Frequenzen des "ersten" Kantenknotens, hier 399, den Zeilen zugeordnet und die Frequenzen des "zweiten" Kantenknotens, hier 400, den Spalten.
Wie kann nun die Modellierung der $x(v, w, p)$-Variablen für jedes $p \in P_{\{v,w\}}$ verbessert werden, mit dem Ziel die Anzahl der nötigen Nebenbedingungen zu verkleinern? Es ist die Idee, für ein $p \in P_{\{v,w\}}$ die $x(v, w, p)$-Variable für jede nötige Penalty-Matrix-Zeile und nicht mehr für jeden einzelnen Eintrag zu modellieren.
Anhand der Penalty-Matrix in Abbildung 4.1 und dem Penalty-Wert 2000 sei dies beispielhaft erklärt. In der Ersten Reformulierung wurden für die neun 2000-Einträge in der ersten Zeile folgende neun Nebenbedingungen definiert:

$$\begin{aligned}
y(v, d_v^1) + y(w, d_w^1) &\leq 1 + x(v, w, 2000) \qquad & y(v, d_v^1) + y(w, d_w^2) &\leq 1 + x(v, w, 2000) \\
y(v, d_v^1) + y(w, d_w^3) &\leq 1 + x(v, w, 2000) \qquad & y(v, d_v^1) + y(w, d_w^4) &\leq 1 + x(v, w, 2000) \\
y(v, d_v^1) + y(w, d_w^5) &\leq 1 + x(v, w, 2000) \qquad & y(v, d_v^1) + y(w, d_w^6) &\leq 1 + x(v, w, 2000) \\
y(v, d_v^1) + y(w, d_w^7) &\leq 1 + x(v, w, 2000) \qquad & y(v, d_v^1) + y(w, d_w^8) &\leq 1 + x(v, w, 2000) \\
& & y(v, d_v^1) + y(w, d_w^9) &\leq 1 + x(v, w, 2000)
\end{aligned}$$

Da $\sum_{i=1}^{9} y(w, d_w^i) \leq 1$ gilt, können diese leicht zu nur einer Nebenbedingung zusammengefasst werden:

$$y(v, d_v^1) + \sum_{i=1}^{9} y(w, d_w^i) \leq 1 + x(v, w, 2000) \tag{4.33}$$

Falls dann dem Knoten v die Frequenz d_v^1 und dem Knoten w eine der Frequenzen d_w^1, …, d_w^9 zugeordnet wird, wird die Variable $x(v, w, 2000)$ gesetzt und so der Zielfunktionswert um den Penalty-Wert 2000 erhöht.
Es gilt (4.33) für eine gesamte Penalty-Matrix und für ein beliebiges $p \in P_{\{v,w\}}$ zu entwickeln. Dazu sind folgende zwei Definitionen sinnvoll:

$$\left(D_v\right)_{\exists D_w, p} := \left\{ d_v \in D_v : \exists\, d_w \in D_w \text{ mit } p(v, d_v, w, d_w) = p \right\} \tag{4.34}$$

$$\left(D_w\right)_{d_v, p} := \left\{ d_w \in D_w : p(v, d_v, w, d_w) = p \right\} \tag{4.35}$$

Die Menge in (4.34) wird dabei beschreiben, welche Zeilen einer Penalty-Matrix für ein p zu beachten sind. Des Weiteren ist (4.35) die Menge, über die in der Nebenbedingung summiert wird.

Die Nebenbedingungen (4.6) aus der Ersten Reformulierung können so durch folgende Ungleichungen ersetzt werden:

$$y(v,d_v) + \sum_{d_w \in (D_w)_{d_v,p}} y(w,d_w) \leq 1 + x(v,w,p) \qquad \begin{array}{c}\forall \{v,w\} \in E, p \in P_{\{v,w\}}, \\ d_v \in (D_v)_{\exists D_w,p}\end{array} \tag{4.36}$$

Wie schon beispielhaft angedeutet modelliert (4.36) folgende verallgemeinerte Aussage von (4.9):

$$y(v,d_v) = 1 \text{ und } \exists\, d_w \text{ mit } y(v,d_w) = 1 \text{ und } (d_v,d_w) \in (D_v \times D_w)_p \Longrightarrow x(v,w,p) = 1 \tag{4.37}$$

Es ist natürlich nicht nur möglich die $x(v,w,p)$-Variablen über die Zeilen der Penalty-Matrix zu modellieren, sondern stattdessen über die Spalten. Mit analogen Definitionen zu (4.34) und (4.35) ergeben sich folgende Ungleichungen:

$$\sum_{d_v \in (D_v)_{d_w,p}} y(v,d_v) + y(w,d_w) \leq 1 + x(v,w,p) \qquad \begin{array}{c}\forall \{v,w\} \in E, p \in P_{\{v,w\}}, \\ d_w \in (D_w)_{\exists D_v,p}\end{array} \tag{4.38}$$

Für eine Implementierung der Zweiten Reformulierung ist die beste Mischung aus (4.36) und (4.38) zu finden. Genauer bedeutet dies, dass pro Kante $\{v,w\} \in E$ und pro $p \in P_{\{v,w\}}$ abgezählt wird, ob es günstiger ist die $x(v,w,p)$-Variable über die Zeilen oder über die Spalten zu modellieren. Das Entscheidungsmerkmal ist hierbei schlicht die geringere Anzahl an benötigten Nebenbedingungen.

Konkret lautet das Programm der Zweiten Reformulierung wie folgt:

Zweite Reformulierung (2nd)

$$\textbf{min} \sum_{\{v,w\} \in E} \sum_{p \in P_{\{v,w\}}} p \cdot x(v,w,p) + \sum_{v \in V} \sum_{d_v \in D_v} q(v,d_v) \cdot y(v,d_v) \tag{4.39}$$

$$\textbf{s.t.} \sum_{d_v \in D_v} y(v,d_v) = 1 \qquad \forall v \in V \tag{4.40}$$

$$y(v,d_v) + \sum_{d_w \in (D_w)_{d_v,p}} y(w,d_w) \leq 1 + x(v,w,p) \qquad \begin{array}{c}\forall \{v,w\} \in E, p \in P_{\{v,w\}} \text{ mit} \\ |(D_v)_{\exists D_w,p}| \leq |(D_w)_{\exists D_v,p}|\end{array} \tag{4.41}$$

$$\sum_{d_v \in (D_v)_{d_w,p}} y(v,d_v) + y(w,d_w) \leq 1 + x(v,w,p) \qquad \begin{array}{c}\forall \{v,w\} \in E, p \in P_{\{v,w\}} \text{ mit} \\ |(D_v)_{\exists D_w,p}| > |(D_w)_{\exists D_v,p}|\end{array} \tag{4.42}$$

$$y(v,d_v) \in \{0,1\} \qquad \forall v \in V, d_v \in D_v \tag{4.43}$$

$$x(v,w,p) \in \{0,1\} \qquad \forall \{v,w\} \in E, p \in P_{\{v,w\}} \tag{4.44}$$

Wenn es günstiger ist $x(v,w,p)$ über die Spalten zu modellieren, greift (4.42). In allen anderen Fällen wird (4.41) verwendet.
Einen genauen Überblick über die Anzahl der Nebenbedingungen der bekannten Instanzen liefert Tabelle 4.3. Im Vergleich zu der Ersten Reformulierung konnte die Anzahl der Nebenbedingungen deutlich verringert werden. Ergänzend enthält Tabelle 4.3 in den letzten beiden Spalten

die Anzahl der Nebenbedingungen der Zweite Reformulierung bei ausschließlicher Modellierung der $x(v, w, p)$-Variablen über die Zeilen bzw. über die Spalten.

Instanz		Anzahl Variablen	Anzahl Nebenbedingungen		
				(über Zeilen)	(über Spalten)
CELAR06	PCS	585.914	28.628		
	1st	6.234	379.607		
	2nd	6.234	52.434	55.353	61.196
CELAR07	PCS	1.331.048	65.928		
	1st	12.916	837.270		
	2nd	12.916	113.516	120.229	131.137
CELAR08	PCS	2.518.664	128.886		
	1st	26.247	1.568.643		
	2nd	26.247	205.950	217.659	240.406
CELAR09	PCS	1.793.484	89.970		
	1st	20.487	1.148.459		
	2nd	20.487	160.204	169.048	185.075
CELAR10	PCS	1.793.484	89.970		
	1st	19.299	1.144.440		
	2nd	19.299	147.774	155.652	168.327
GRAPH05	PCS	574.628	30.986		
	1st	5.440	324.479		
	2nd	5.440	34.994	38.252	40.910
GRAPH06	PCS	1.201.638	63.846		
	1st	10.875	671.049		
	2nd	10.875	70.946	77.579	85.601
GRAPH07	PCS	1.146.426	62.290		
	1st	10.707	634.270		
	2nd	10.707	68.218	75.359	82.260
GRAPH11	PCS	2.032.792	107.688		
	1st	18.700	1.169.944		
	2nd	18.700	126.033	136.013	151.802
GRAPH12	PCS	1.805.190	95.408		
	1st	19.548	1.165.353		
	2nd	19.548	134.195	146.070	165.029
GRAPH13	PCS	2.798.400	145.034		
	1st	25.755	1.693.257		
	2nd	25.755	184.036	199.762	215.128

Tabelle 4.3: Anzahl der Variablen und Nebenbedingungen der PCS Formulierung (PCS), der Ersten Reformulierung (1st) und der Zweiten Reformulierung (2nd)

Schließlich wird für die letztendliche Penalty Block Formulierung zusätzlich gegenüber der Zweiten Reformulierung die schon angesprochene Blockstruktur der Penalty-Matrizen ausgenutzt.

4.4 Die Penalty Block Formulierung

Mit erneutem Blick auf die Penalty-Matrix in Abbildung 4.1 wird im Folgenden erläutert, wie die erkennbare Blockstruktur der Daten ausgenutzt werden kann. Dabei wird weiterhin das Ziel verfolgt, die Anzahl der Nebenbedingungen der Reformulierung zu verringern.

Die vorliegende Penalty-Matrix (Abbildung 4.4) enthält oben links einen fast vollständigen 9×9-Block mit Penalty-Wert 2000. Nur in der linken unteren Ecke dieses Blockes sind drei andere Penalty-Werte. Laut der Zweiten Reformulierung bilden diese 2000-Einträge zeilenweise folgende neun Nebenbedingungen:

```
2000 2000 2000 2000 2000 2000 2000 2000 2000 ...
2000 2000 2000 2000 2000 2000 2000 2000 2000
2000 2000 2000 2000 2000 2000 2000 2000 2000
2000 2000 2000 2000 2000 2000 2000 2000 2000
2000 2000 2000 2000 2000 2000 2000 2000 2000
2000 2000 2000 2000 2000 2000 2000 2000 2000
2000 2000 2000 2000 2000 2000 2000 2000 2000
2100 2000 2000 2000 2000 2000 2000 2000 2000
2100 2100 2000 2000 2000 2000 2000 2000 2000
...                                       ...
```

Abbildung 4.4: Ausschnitt aus Abb. 4.1

$$y(v,d_v^1)+\sum_{i=1}^{9} y(w,d_w^i) \le 1+x(v,w,2000) \qquad y(v,d_v^2)+\sum_{i=1}^{9} y(w,d_w^i) \le 1+x(v,w,2000)$$

$$y(v,d_v^3)+\sum_{i=1}^{9} y(w,d_w^i) \le 1+x(v,w,2000) \qquad y(v,d_v^4)+\sum_{i=1}^{9} y(w,d_w^i) \le 1+x(v,w,2000)$$

$$y(v,d_v^5)+\sum_{i=1}^{9} y(w,d_w^i) \le 1+x(v,w,2000) \qquad y(v,d_v^6)+\sum_{i=1}^{9} y(w,d_w^i) \le 1+x(v,w,2000)$$

$$y(v,d_v^7)+\sum_{i=1}^{9} y(w,d_w^i) \le 1+x(v,w,2000) \qquad y(v,d_v^8)+\sum_{i=2}^{9} y(w,d_w^i) \le 1+x(v,w,2000)$$

$$y(v,d_v^9)+\sum_{i=3}^{9} y(w,d_w^i) \le 1+x(v,w,2000)$$

Es ist zu beachten, dass bei den letzten beiden Nebenbedingungen die Summe über die $y(w,d_w^i)$-Variablen erst bei $i=2$ bzw. $i=3$ beginnt. Dies ist begründet in den drei 2100-Einträgen in der linken unteren Ecke des fast vollständigen 9×9-Blockes mit Penalty-Wert 2000.
Da $\sum_{i=1}^{9} y(w,d_w^i) \le 1$ und analog ebenso $\sum_{i=1}^{7} y(v,d_v^i) \le 1$ gelten, können diese neun zu nur drei Nebenbedingungen zusammengefasst werden. Eine für den vollständigen 7×9-Block der 2000-Einträge und zwei für die beiden übrigbleibenden Zeilen unterhalb dieses Blockes.

$$\sum_{i=1}^{7} y(v,d_v^i)+\sum_{j=1}^{9} y(w,d_w^j) \le 1+x(v,w,2000)$$

$$y(v,d_v^8)+\sum_{i=2}^{9} y(w,d_w^i) \le 1+x(v,w,2000) \qquad y(v,d_v^9)+\sum_{i=3}^{9} y(w,d_w^i) \le 1+x(v,w,2000)$$

Natürlich können die zwei Zeilen auch zu einem 2×7-Block zusammengefasst werden. Doch dann ist für den einen nichtüberdeckten 2000-Eintrag eine weitere Nebenbedingung nötig, sodass insgesamt die Anzahl der Nebenbedingungen gleich ist. Ein Block lohnt sich erst ab einer Breite und einer Höhe von mehr als zwei Einträgen. Um diese Blockidee zu verwirklichen ist ein Algorithmus zu definieren, der passende Blöcke innerhalb einer Penalty-Matrix findet. Dabei soll der Algorithmus nicht nur Blöcke liefern, sondern auch die Anzahl der insgesamt nötigen Nebenbedingungen beachten. Matrixeinträge, die nicht zu einem gefundenen Block gehören,

werden zeilen- bzw. spaltenweise in die Formulierung aufgenommen. Somit müssen auch die Nicht-Blockeinträge in dem Algorithmus berücksichtigt werden. Dementsprechend hat ein solcher Algorithmus die optimale Block-Zeilen-Spalten-Zerlegung einer Penalty-Matrix zu finden.

Definition 4.18.
Für eine Penalty-Matrix $M_{\{v,w\}}$ und einen Penalty-Wert $p \in P_{\{v,w\}}$ ist $(B_p, Z_p, S_p)_{\{v,w\}}$ mit B_p die Menge der Blöcke und Z_p sowie S_p die Menge der Zeilen bzw. Spalten eine zulässige *Block-Zeilen-Spalten-Zerlegung*, falls jeder Matrixeintrag m_{ij} mit $m_{ij} = p$ in einem Block aus B_p oder einer Zeile aus Z_p oder einer Spalte aus S_p enthalten ist. Eine Block-Zeilen-Spalten-Zerlegung bildet folglich eine Überdeckung aller Matrixeinträge mit Wert p.

Da pro Block, pro Zeile und pro Spalte nach den obrigen Überlegungen genau eine Nebenbedingung benötigt wird, ist folgende Definition schnell einzusehen.

Definition 4.19.
Eine *optimale Block-Zeilen-Spalten-Zerlegung* für eine Penalty-Matrix $M_{\{v,w\}}$ und einen Penalty-Wert $p \in P_{\{v,w\}}$ ist eine Block-Zeilen-Spalten-Zerlegung $(B_p, Z_p, S_p)_{\{v,w\}}$ mit $|B_p| + |Z_p| + |S_p|$ minimal.

Der für die Implementierung der Penalty Block Formulierung für diese Arbeit verwendete Algorithmus ist in Abschnitt 4.5.1 angegeben. Dieser ist recht einfach und intuitiv gehalten. Es kann noch beliebig viel Anstrengung investiert werden, den Algorithmus zu verbessern. Dies wurde zum Teil auch in dem genannten Abschnitt angesprochen.

Zu einer gegebenen Kollektion von Block-Zeilen-Spalten-Zerlegungen $\{(B_p, Z_p, S_p)_{\{v,w\}}\}$, die aus zulässigen Block-Zeilen-Spalten-Zerlegungen $(B_p, Z_p, S_p)_{\{v,w\}}$ für alle $\{v, w\} \in E$ und alle $p \in P_{\{v,w\}}$ besteht, werden so die Nebenbedingungen (4.41) und (4.42) aus der Zweiten Reformulierung durch Block-, Zeilen- und Spaltennebenbedingungen ersetzt. Für die Angabe derer sei folgendes vereinbart.

Definition 4.20.
Für einen Block $[b] \in B_p$ sei $[b]_v$ die Menge der zu Block $[b]$ gehörigen $d_v \in D_v$ und $[b]_w$ die Menge der zu Block $[b]$ gehörigen $d_w \in D_w$

Für einen Block $[b] \in B_p$ gilt für alle $d_v \in [b]_v$ und alle $d_w \in [b]_w$ die Gleichung $p(v, d_v, w, d_w) = p$.

Beispiel.
Sei $[b] \in B_{2000}$ der oben aufgezeigte 7×9-Block der 2000-Einträge in der Penalty-Matrix der Kante $\{399, 400\} \in E$. Dann ist $[b]_v = \{d_v^1, \ldots, d_v^7\}$ und $[b]_w = \{d_w^1, \ldots, d_w^9\}$.

Definition 4.21.
Für eine Zeile $z \in Z_p$ sei z_v die zu der Zeile z gehöhrige Frequenz $d_v \in D_v$. Für eine Spalte $s \in S_p$ sei s_w die zu der Spalte s gehöhrige Frequenz $d_w \in D_w$.

Beispiel.
Seien $z^1, z^2 \in Z_{2000}$ die beiden oben aufgezeigten Zeilen der 2000-Einträge unterhalb des 7×9-Blockes. Dann ist $z_v^1 = d_v^8$ und $z_v^2 = d_v^9$.

Die Blocknebenbedingungen lauten dann wie folgt:

$$\sum_{d_v \in [b]_v} y(v, d_v) + \sum_{d_w \in [b]_w} y(w, d_w) \leq 1 + x(v, w, p) \qquad \begin{matrix} \forall \{v, w\} \in E, \\ p \in P_{\{v,w\}}, [b] \in B_p \end{matrix} \tag{4.45}$$

Die Zeilen- und Spaltennebenbedingungen sind anzugeben als:

$$y(v, z_v) + \sum_{d_w \in (D_w)_{z_v, p}} y(w, d_w) \leq 1 + x(v, w, p) \qquad \begin{matrix} \forall \{v, w\} \in E, \\ p \in P_{\{v,w\}}, z \in Z_p \end{matrix} \tag{4.46}$$

$$\sum_{d_v \in (D_v)_{s_w, p}} y(v, d_v) + y(w, s_w) \leq 1 + x(v, w, p) \qquad \begin{matrix} \forall \{v, w\} \in E, \\ p \in P_{\{v,w\}}, s \in S_p \end{matrix} \tag{4.47}$$

Das konkrete Programm der Penalty Block Formulierung wird vollständig im nachfolgenden Abschnitt 4.5 angegeben.

Instanz		Anzahl Variablen			Anzahl
		Summe	kontinuierliche	binäre	Nebenbedingungen
CELAR06	PCS	585.914	581.904	4.010	28.628
	1st	6.234	2.224	4.010	379.607
	2nd	6.234	2.224	4.010	52.434
	PB	6.234	2.224	4.010	44.383
CELAR07	PCS	1.331.048	1.323.072	7.976	65.928
	1st	12.916	4.940	7.976	837.270
	2nd	12.916	4.940	7.976	113.516
	PB	12.916	4.940	7.976	98.029
CELAR08	PCS	2.518.664	2.500.564	18.100	128.886
	1st	26.247	8.147	18.100	1.568.643
	2nd	26.247	8.147	18.100	205.950
	PB	26.247	8.147	18.100	180.508
CELAR09	PCS	1.793.484	1.780.056	13.428	89.970
	1st	20.487	7.059	13.428	1.148.459
	2nd	20.487	7.059	13.428	160.204
	PB	20.487	7.059	13.428	137.432
CELAR10	PCS	1.793.484	1.780.056	13.428	89.970
	1st	19.299	5.871	13.428	1.144.440
	2nd	19.299	5.871	13.428	147.774
	PB	19.299	5.871	13.428	135.607

Tabelle 4.4: Anzahl der Variablen und Nebenbedingungen der PCS Formulierung (PCS) und der Reformulierungen, inkl. der Penalty Block Formulierung (PB)

Tabelle 4.4 und Tabelle 4.5 zeigen die Anzahl der benötigten Nebenbedingungen bei der Penalty Block Formulierung unter Verwendung von Algorithmus 4.22. Es wird deutlich, dass erneut die Anzahl der Nebenbedingungen reduziert werden konnte. Bei den drei Instanzen GRAPH05, GRAPH06 und GRAPH07 konnte sogar die Anzahl der Nebenbedingungen der PCS Formulierung unterschritten werden.

Instanz		Anzahl Variablen			Anzahl Nebenbedingungen
		Summe	kontinuierliche	binäre	
GRAPH05	PCS	574.628	570.920	3.708	30.986
	1st	5.440	1.732	3.708	324.479
	2nd	5.440	1.732	3.708	34.994
	PB	5.440	1.732	3.708	30.608
GRAPH06	PCS	1.201.638	1.194.096	7.542	63.846
	1st	10.875	3.333	7.542	671.049
	2nd	10.875	3.333	7.542	70.946
	PB	10.875	3.333	7.542	63.424
GRAPH07	PCS	1.146.426	1.139.096	7.330	62.290
	1st	10.707	3.377	7.330	634.270
	2nd	10.707	3.377	7.330	68.218
	PB	10.707	3.377	7.330	61.049
GRAPH11	PCS	2.032.792	2.019.972	12.820	107.688
	1st	18.700	5.880	12.820	1.169.944
	2nd	18.700	5.880	12.820	126.033
	PB	18.700	5.880	12.820	112.567
GRAPH12	PCS	1.805.190	1.792.408	12.782	95.408
	1st	19.548	6.766	12.782	1.165.353
	2nd	19.548	6.766	12.782	134.195
	PB	19.548	6.766	12.782	119.881
GRAPH13	PCS	2.798.400	2.780.812	17.588	145.034
	1st	25.755	8.167	17.588	1.693.257
	2nd	25.755	8.167	17.588	184.036
	PB	25.755	8.167	17.588	163.801

Tabelle 4.5: Anzahl der Variablen und Nebenbedingungen der PCS Formulierung (PCS) und der Reformulierungen, inkl. der Penalty Block Formulierung (PB)

In dem nächsten Abschnitt folgt nun das exakte Programm für die Penalty Block Formulierung. Da es sich hierbei um die neue Formulierung für das Frequenzzuweisungsproblem handelt und das Hauptergebnis dieser Arbeit darstellt, werden die Variablen und Parameter zum größten Teil wiederholt.

4.5 PB Integer Linear Programm für das MI-FAP

Das Penalty Block Integer Linear Programm für das Minimum Interference Frequency Assignment Problem ist für einen ungerichteten Penalty-Interferenz Graphen (V, E) definiert. Dabei repräsentiert ein Knoten $v \in V$ eine Verbindung, der eine Frequenz $d_v \in D_v$ zugeordnet werden soll. Für alle $v \in V$ und $d_v \in D_v$ sind folgende binäre Variablen definiert:

$$y(v, d_v) = \begin{cases} 1 & \text{falls Frequenz } d_v \in D_v \text{ gewählt} \\ 0 & \text{sonst} \end{cases}$$

Eine Kante $\{v, w\} \in E$ zwischen zwei Knoten $v, w \in V$ bedeutet, dass zu den Verbindungen v, w mindestens ein Frequenzpaar $d_v \in D_v, d_w \in D_w$ vorhanden ist, sodass diese interferieren können. Jeder Kante $\{v, w\} \in E$ ist eine Penalty-Matrix $M_{\{v,w\}}$ zugeordnet, die für jedes Frequenzenpaar den Penalty-Wert $p(v, d_v, w, d_w)$ für die dann auftretende Interferenz enthält. Die Menge $P_{\{v,w\}}$ enthält sämtliche, verschiedene Penalty-Werte der Matrix $M_{\{v,w\}}$ ungleich Null. Für alle $\{v, w\} \in E$ und alle $p \in P_{\{v,w\}}$ ist folgende binäre Variable definiert:

$$x(v, w, p) = \begin{cases} 1 & \text{falls } (d_v, d_w) \in D_v \times D_w \text{ mit } p(v, d_v, w, d_w) = p \text{ gewählt} \\ 0 & \text{sonst} \end{cases}$$

Des Weiteren enthält die Menge $(D_w)_{d_v,p} \subseteq D_w$ alle Frequenzen $d_w \in D_w$, die mit der Frequenz $d_v \in D_v$ einen Penalty-Wert p bilden. Für eine gegebene Kollektion von Block-Zeilen-Spalten-Zerlegungen $\{(B_p, Z_p, S_p)_{\{v,w\}}\}$, wie sie in Abschnitt 4.4 eingeführt wurde, lautet das Programm der Penalty Block Formulierung für das MI-FAP wie folgt:

Penalty Block Formulierung (PB)

$$\textbf{min} \sum_{\{v,w\}\in E} \sum_{p\in P_{\{v,w\}}} p \cdot x(v, w, p) + \sum_{v\in V} \sum_{d_v\in D_v} q(v, d_v) \cdot y(v, d_v) \tag{4.48}$$

$$\textbf{s.t.} \sum_{d_v\in D_v} y(v, d_v) = 1 \qquad \forall v \in V \tag{4.49}$$

$$\sum_{d_v\in [b]_v} y(v, d_v) + \sum_{d_w\in [b]_w} y(w, d_w) \leq 1 + x(v, w, p) \qquad \begin{array}{c} \forall \{v, w\} \in E, \\ p \in P_{\{v,w\}}, [b] \in B_p \end{array} \tag{4.50}$$

$$y(v, z_v) + \sum_{d_w\in (D_w)_{z_v,p}} y(w, d_w) \leq 1 + x(v, w, p) \qquad \begin{array}{c} \forall \{v, w\} \in E, \\ p \in P_{\{v,w\}}, z \in Z_p \end{array} \tag{4.51}$$

$$\sum_{d_v\in (D_v)_{s_w,p}} y(v, d_v) + y(w, s_w) \leq 1 + x(v, w, p) \qquad \begin{array}{c} \forall \{v, w\} \in E, \\ p \in P_{\{v,w\}}, s \in S_p \end{array} \tag{4.52}$$

$$y(v, d_v) \in \{0, 1\} \qquad \forall v \in V, d_v \in D_v \tag{4.53}$$

$$x(v, w, p) \in \{0, 1\} \qquad \forall \{v, w\} \in E, p \in P_{\{v,w\}} \tag{4.54}$$

Die Nebenbedingungen (4.49) modellieren die Tatsache, dass pro Knoten genau eine Frequenz zugeordnet wird. Die Ungleichungen (4.50) - (4.52) stellen die Modellierung der $x(v, w, p)$-Variablen sicher. Dabei wird ein $x(v, w, p)$ gesetzt, wenn $d_v \in D_v$ und $d_w \in D_w$ mit $p(v, d_v, w, d_w) = p$ und $y(v, d_v) = y(w, d_w) = 1$ existieren. Also, wenn zwei Frequenzen zugewiesen wurden, die einen Penalty-Wert p implizieren. Dabei bilden zu gegebenen Block-Zeilen-Spalten-Zerlegungen (4.50)

die Nebenbedingungen der Blöcke sowie (4.51) und (4.52) die der Zeilen und Spalten. Aufgrund der binären y-Variablen besteht die Möglichkeit die x-Variablen kontinuierlich zwischen 0 und 1 zu definieren.

4.5.1 Algorithmus für die Blocksuche

Für die Implementierung der Penalty Block Formulierung wurde in dieser Arbeit Algorithmus 4.22 für die Block-Zeilen-Spalten-Zerlegungen verwendet. Der wird nachfolgend erläutert.

Algorithmus 4.22.

Eingabe: Penalty-Matrix $M_{\{v,w\}} \in \mathbb{R}_+^{n \times m}$ mit Einträgen m_{ij}, Penalty-Wertemenge $P_{\{v,w\}}$
Ausgabe: für alle $p \in P_{\{v,w\}}$ eine zulässige Block-Zeilen-Spalten-Zerlegung (B_p, Z_p, S_p)

for $p \in P_{\{v,w\}}$ **do**
  initialize B_p, Z_p, S_p
  while passender Block im letzten Durchgang gefunden **or** erster Durchgang **do**
    initialize *bester Block*
    for $i = 1, \ldots, n$ **do**
      for $j = 1, \ldots, m$ **do**
        if $m_{ij} = p$ **and** m_{ij} nicht überdeckt durch B_p **then**
          initialize *aktueller Block*
          linke obere und *rechte untere Ecke aktueller Block* $\leftarrow (i, j)$
          while aktueller Block vergrößerbar um eine Zeile oder eine Spalte **do**
            passe *rechte untere Ecke aktueller Block* an
          end while
          if *aktuelle Blockbreite* ≥ 5 **and** *aktuelle Blockhöhe* ≥ 5
                and *aktueller Block* $>$ *bester Block* **then**
            bester Block $\leftarrow$ *aktueller Block*
          end if
        end if
      end for
    end for
    $B_p \leftarrow B_p \cup$ *bester Block*
  end while
  while es existiert ein von B_p, Z_p, S_p unüberdeckter p-Eintrag **do**
    $z :=$ Zeile mit meisten unüberdeckten p-Einträgen
    $s :=$ Spalte mit meisten unüberdeckten p-Einträgen
    if s enthält mehr unüberdeckte p-Einträge als z **then**
      $S_p \leftarrow S_p \cup s$
    else
      $Z_p \leftarrow Z_p \cup z$
    end if
  end while
end for

Der Algorithmus 4.22 für die Blocksuche teilt sich in zwei Abschnitte. Im ersten, Zeile 3 bis Zeile 20, werden die Blöcke gefunden und im zweiten Abschnitt, Zeile 21 bis Zeile 29, werden alle nicht von Blöcken überdeckten Penalty-Einträge mithilfe von Zeilen und Spalten überdeckt.
In dem Blockabschnitt wird die Penalty-Matrix zeilenweise durchlaufen bis ein potentieller Block in Form eines Penaly-Wertes p auftritt. Dieser wird dann abwechselnd und soweit möglich um einzelne Zeilen und Spalten vergrößert. Dabei wird stets darauf geachtet, dass der Block weiterhin nur Penalty-Werte p enthält. Kann ein Block nicht mehr vergrößert werden, wird überprüft, ob dieser die Mindesthöhe und -breite von 5 besitzt. In vorherigen Untersuchungen stellte sich heraus, dass diese Parameterwahl bei der nach Preprocessing größten Testinstanz `GRAPH13` zu der geringsten Anzahl an Nebenbedingungen führte. Falls dieser Block größer, also mehr Einträge umfasst als der in diesem Durchgang bisher gefundene *beste Block*, wird dieser aktualisiert. Nachdem die letzte Zeile der Penalty-Matrix durchlaufen wurde, wird der *beste Block* der Menge B_p hinzugefügt. Insgesamt liefert jeder Duchgang der Penalty-Matrix den größtmöglichen Block von noch nicht überdeckten p-Einträgen, bis kein passender Block mehr gefunden wurde.
Anschließend werden bisher nicht überdeckte Penalty-Einträge p mit Zeilen und Spalten überdeckt. Dabei wird durchweg die Zeile bzw. die Spalte der jeweiligen Menge Z_p bzw. S_p hinzugefügt, die die meisten nichtüberdeckten p-Einträge enthält.
Letztendlich liefert Algorithmus 4.22 für jedes $p \in P_{\{v,w\}}$ eine zulässige Block-Zeilen-Spalten-Zerlegung.

Der angegebene Algorithmus kann natürlich noch verbessert werden. Ein Ansatz wäre, für jedes $p \in P_{\{v,w\}}$ im Vorhinein die besten Parameter für die geforderte Mindestbreite und Mindesthöhe der Blöcke (siehe Zeile 13) zu finden. Dabei ist eine Parameterwahl besser als eine andere, wenn diese zu einer geringeren Anzahl an insgesamt für diesen Penalty-Wert in der Penalty-Matrix benötigten Nebenbedingungen führt. Es ist offensichtlich, dass sich ein Block erst ab einer Höhe und Breite von drei Einträgen lohnt, da dieser sonst mit der selben Anzahl an Nebenbedingungen mit Zeilen bzw. Spalten überdeckt werden kann. Versuche mit diesem Ansatz zeigten, dass die besten Parameter meist größer als drei sind. Anhand von Testinstanz `CELAR06` wurde die vorherige Suche nach der besten Parameterwahl ausprobiert. Der Algorithmus brauchte ein vielfaches länger, die Block-Zeilen-Spalten-Zerlegungen zu liefern. Dabei konnte die Anzahl der Nebenbedingungen jedoch nur um knapp $1{,}5\%$ verringert werden.
Trotz des intuitiven Charakters des Algorithmus 4.22, liefert dieser ein annehmbares, gutes Ergebnis. Das veranschaulicht auch das folgende Beispiel in Verbindung mit Abbildung 4.5.

Beispiel.
In Abbildung 4.5 ist erneut die Penalty-Matrix aus Abbildung 4.1 dargestellt. Zusätzlich sind die Blöcke markiert, die Algorithmus 4.22 zu dieser Matrix liefert. Neben diesen acht Blöcken werden noch zwölf Spalten und 114 Zeilen benötigt, um für die Kante $\{399, 400\}$ eine Kollektion von Block-Zeilen-Spalten-Zerlegungen zu bilden. Da in Algorithmus 4.22 ab Zeile 22 bei Gleichheit der Anzahl der unüberdeckten p-Einträge von Zeile und Spalte die Zeilennebenbedingung gewählt wird, ist die Mächtigkeit von Z_p soviel größer als die von S_p.

2000	2000	2000	2000	2000	2000	2000	2000	2000	2100	2100	2100	2100	1100	1100	1100	1100	1100	0	0	0	0	0	0	0	0	0	1000	1000	1000	1000	1000	999	999	999	999
2000	2000	2000	2000	2000	2000	2000	2000	2000	2100	2100	2100	2100	2100	1100	1100	1100	1100	0	0	0	0	0	0	0	0	0	1000	1000	1000	1000	1000	1000	999	999	999
2000	2000	2000	2000	2000	2000	2000	2000	2000	2100	2100	2100	2100	2100	2100	1100	1100	1100	0	0	0	0	0	0	0	0	0	1000	1000	1000	1000	1000	1000	1000	999	999
2000	2000	2000	2000	2000	2000	2000	2000	2000	2100	2100	2100	2100	2100	2100	2100	1100	1100	0	0	0	0	0	0	0	0	0	1000	1000	1000	1000	1000	1000	1000	1000	999
2000	2000	2000	2000	2000	2000	2000	2000	2000	1100	2100	2100	2100	2100	2100	2100	2100	1100	0	0	0	0	0	0	0	0	0	1000	1000	1000	1000	1000	1000	1000	1000	1000
2000	2000	2000	2000	2000	2000	2000	2000	2000	1100	1100	2100	2100	2100	2100	2100	2100	2100	0	0	0	0	0	0	0	0	0	1000	1000	1000	1000	1000	1000	1000	1000	1000
2000	2000	2000	2000	2000	2000	2000	2000	2000	1100	1100	1100	2100	2100	2100	2100	2100	2100	0	0	0	0	0	0	0	0	0	1000	1000	1000	1000	1000	1000	1000	1000	1000
2100	2000	2000	2000	2000	2000	2000	2000	2000	1100	1100	1100	1100	2100	2100	2100	2100	2100	0	0	0	0	0	0	0	0	0	1000	1000	1000	1000	1000	1000	1000	1000	1000
2100	2100	2000	2000	2000	2000	2000	2000	2000	1100	1100	1100	1100	1100	2100	2100	2100	2100	0	0	0	0	0	0	0	0	0	1000	1000	1000	1000	1000	1000	1000	1000	1000
2101	2101	2101	2101	1101	1101	1101	1101	1101	2001	2001	2001	2001	2001	2001	2001	2101	2101	1001	1001	1001	1001	1001	1000	1000	1000	1000	0	0	0	0	0	0	0	0	0
2101	2101	2101	2101	2101	1101	1101	1101	1101	2001	2001	2001	2001	2001	2001	2001	2001	2101	1001	1001	1001	1001	1001	1001	1000	1000	1000	0	0	0	0	0	0	0	0	0
2101	2101	2101	2101	2101	2101	1101	1101	1101	2001	2001	2001	2001	2001	2001	2001	2001	2001	1001	1001	1001	1001	1001	1001	1001	1000	1000	100	0	0	0	0	0	0	0	0
2101	2101	2101	2101	2101	2101	2101	1101	1101	2001	2001	2001	2001	2001	2001	2001	2001	2001	1001	1001	1001	1001	1001	1001	1001	1001	1000	100	100	0	0	0	0	0	0	0
1101	2101	2101	2101	2101	2101	2101	2101	1101	2001	2001	2001	2001	2001	2001	2001	2001	2001	1001	1001	1001	1001	1001	1001	1001	1001	1001	100	100	100	0	0	0	0	0	0
1101	1101	2101	2101	2101	2101	2101	2101	2101	2001	2001	2001	2001	2001	2001	2001	2001	2001	1001	1001	1001	1001	1001	1001	1001	1001	1001	100	100	100	100	0	0	0	0	0
1101	1101	1101	2101	2101	2101	2101	2101	2101	2001	2001	2001	2001	2001	2001	2001	2001	2001	1001	1001	1001	1001	1001	1001	1001	1001	1001	100	100	100	100	100	0	0	0	0
1101	1101	1101	1101	2101	2101	2101	2101	2101	2001	2001	2001	2001	2001	2001	2001	2001	2001	1001	1001	1001	1001	1001	1001	1001	1001	1001	100	100	100	100	100	100	0	0	0
1101	1101	1101	1101	1101	2101	2101	2101	2101	2001	2001	2001	2001	2001	2001	2001	2001	2001	1001	1001	1001	1001	1001	1001	1001	1001	1001	100	100	100	100	100	100	100	0	0
0	0	100	100	100	100	100	100	100	1001	1001	1001	1001	1001	1001	1001	1001	1001	2001	2001	2001	2001	2001	2001	2001	2001	2001	2101	2101	2101	2101	1101	1101	1101	1101	1101
0	0	0	100	100	100	100	100	100	1001	1001	1001	1001	1001	1001	1001	1001	1001	2001	2001	2001	2001	2001	2001	2001	2001	2001	2101	2101	2101	2101	2101	1101	1101	1101	1101
0	0	0	0	100	100	100	100	100	1001	1001	1001	1001	1001	1001	1001	1001	1001	2001	2001	2001	2001	2001	2001	2001	2001	2001	2101	2101	2101	2101	2101	2101	1101	1101	1101
0	0	0	0	0	100	100	100	100	1001	1001	1001	1001	1001	1001	1001	1001	1001	2001	2001	2001	2001	2001	2001	2001	2001	2001	2101	2101	2101	2101	2101	2101	2101	1101	1101
0	0	0	0	0	0	100	100	100	1001	1001	1001	1001	1001	1001	1001	1001	1001	2001	2001	2001	2001	2001	2001	2001	2001	2001	1101	2101	2101	2101	2101	2101	2101	2101	1101
0	0	0	0	0	0	0	100	100	1000	1001	1001	1001	1001	1001	1001	1001	1001	2001	2001	2001	2001	2001	2001	2001	2001	2001	1101	1101	2101	2101	2101	2101	2101	2101	2101
0	0	0	0	0	0	0	0	100	1000	1000	1001	1001	1001	1001	1001	1001	1001	2001	2001	2001	2001	2001	2001	2001	2001	2001	1101	1101	1101	2101	2101	2101	2101	2101	2101
0	0	0	0	0	0	0	0	0	1000	1000	1000	1001	1001	1001	1001	1001	1001	2101	2001	2001	2001	2001	2001	2001	2001	2001	1101	1101	1101	1101	2101	2101	2101	2101	2101
0	0	0	0	0	0	0	0	0	1000	1000	1000	1000	1001	1001	1001	1001	1001	2101	2101	2001	2001	2001	2001	2001	2001	2001	1101	1101	1101	1101	1101	2101	2101	2101	2101
1000	1000	1000	1000	1000	1000	1000	1000	1000	0	0	0	0	0	0	0	0	0	2100	2100	2100	2100	1100	1100	1100	1100	1100	2000	2000	2000	2000	2000	2000	2000	2100	2100
1000	1000	1000	1000	1000	1000	1000	1000	1000	0	0	0	0	0	0	0	0	0	2100	2100	2100	2100	2100	1100	1100	1100	1100	2000	2000	2000	2000	2000	2000	2000	2000	2100
1000	1000	1000	1000	1000	1000	1000	1000	1000	0	0	0	0	0	0	0	0	0	2100	2100	2100	2100	2100	2100	1100	1100	1100	2000	2000	2000	2000	2000	2000	2000	2000	2000
1000	1000	1000	1000	1000	1000	1000	1000	1000	0	0	0	0	0	0	0	0	0	2100	2100	2100	2100	2100	2100	2100	1100	1100	2000	2000	2000	2000	2000	2000	2000	2000	2000
1000	1000	1000	1000	1000	1000	1000	1000	1000	0	0	0	0	0	0	0	0	0	1100	2100	2100	2100	2100	2100	2100	2100	1100	2000	2000	2000	2000	2000	2000	2000	2000	2000
999	1000	1000	1000	1000	1000	1000	1000	1000	0	0	0	0	0	0	0	0	0	1100	1100	2100	2100	2100	2100	2100	2100	2100	2000	2000	2000	2000	2000	2000	2000	2000	2000
999	999	1000	1000	1000	1000	1000	1000	1000	0	0	0	0	0	0	0	0	0	1100	1100	1100	2100	2100	2100	2100	2100	2100	2000	2000	2000	2000	2000	2000	2000	2000	2000
999	999	999	1000	1000	1000	1000	1000	1000	0	0	0	0	0	0	0	0	0	1100	1100	1100	1100	2100	2100	2100	2100	2100	2000	2000	2000	2000	2000	2000	2000	2000	2000
999	999	999	999	1000	1000	1000	1000	1000	0	0	0	0	0	0	0	0	0	1100	1100	1100	1100	1100	2100	2100	2100	2100	2000	2000	2000	2000	2000	2000	2000	2000	2000

Abbildung 4.5: Penalty-Matrix der Kante $\{399, 400\}$ der Instanz CELAR06 inkl. der von Algorithmus 4.22 gefundenen Blöcken

Bei genauerer Betrachtung der Blöcke wird deutlich, dass Algorithmus 4.22 diese in der Penalty-Matrix perfekt gewählt hat. Überall, wo gleiche Penalty-Werte in vollständigen oder doppelseitigen Treppen auftreten lohnt sich kein Block. Dies wird im Folgenden näher erläutert.

Beispielsweise werden für die vollständigen 100er Treppe (Abb. 4.6) stets sieben Nebenbedingungen benötigt, unabhängig davon ob und wie ein Block gewählt wird. Der Grund liegt darin, dass kein Block mehr als zwei Zeilen oder Spalten ersetzen kann.

```
  0   0 100 100 100 100 100 100 100
  0   0   0 100 100 100 100 100 100
  0   0   0   0 100 100 100 100 100
  0   0   0   0   0 100 100 100 100
  0   0   0   0   0   0 100 100 100
  0   0   0   0   0   0   0 100 100
  0   0   0   0   0   0   0   0 100
  0   0   0   0   0   0   0   0   0
  0   0   0   0   0   0   0   0   0
```

Abbildung 4.6: Vollständige 100er Treppe

```
2100 2100 2100 2100 1100 1100 1100 1100 1100
2100 2100 2100 2100 2100 1100 1100 1100 1100
2100 2100 2100 2100 2100 2100 1100 1100 1100
2100 2100 2100 2100 2100 2100 2100 1100 1100
1100 2100 2100 2100 2100 2100 2100 2100 1100
1100 1100 2100 2100 2100 2100 2100 2100 2100
1100 1100 1100 2100 2100 2100 2100 2100 2100
1100 1100 1100 1100 2100 2100 2100 2100 2100
1100 1100 1100 1100 1100 2100 2100 2100 2100
```

Abbildung 4.7: doppelseitige 2100er Treppe

Für die vollständigen 1100er Treppen gilt das selbe. Diese schließen im Paar eine doppelseitige 2100er Treppe ein (Abb. 4.7). Hierbei ist zu beachten, dass noch drei weitere Penalty-Werte 2100 in gleichen Zeilen existieren. Dennoch gilt auch hier, dass kein Block mehr als zwei Zeilen oder Spalten ersetzten kann.

Die Bereiche, in denen der Algorithmus Blöcke liefert (Abb. 4.8 und Abb. 4.9), sind alle ähnlich. Sie enthalten jeweils in einer Ecke eine zweier- bzw. vierer-Treppe. Es wird auch deutlich,

```
2000 2000 2000 2000 2000 2000 2000 2000 2000
2000 2000 2000 2000 2000 2000 2000 2000 2000
2000 2000 2000 2000 2000 2000 2000 2000 2000
2000 2000 2000 2000 2000 2000 2000 2000 2000
2000 2000 2000 2000 2000 2000 2000 2000 2000
2000 2000 2000 2000 2000 2000 2000 2000 2000
2000 2000 2000 2000 2000 2000 2000 2000 2000
2100 2000 2000 2000 2000 2000 2000 2000 2000
2100 2100 2000 2000 2000 2000 2000 2000 2000
```

Abbildung 4.8: Bereich mit zweier-Treppe

```
1000 1000 1000 1000 1000 1000 1000 1000 1000
1000 1000 1000 1000 1000 1000 1000 1000 1000
1000 1000 1000 1000 1000 1000 1000 1000 1000
1000 1000 1000 1000 1000 1000 1000 1000 1000
1000 1000 1000 1000 1000 1000 1000 1000 1000
 999 1000 1000 1000 1000 1000 1000 1000 1000
 999  999 1000 1000 1000 1000 1000 1000 1000
 999  999  999 1000 1000 1000 1000 1000 1000
 999  999  999  999 1000 1000 1000 1000 1000
```

Abbildung 4.9: Bereich mit vierer-Treppe

dass die dort gefundenen Blöcke nicht besser festgelegt werden können. In dem Bereich mit der vierer-Treppe können auch zwei Blöcke platziert werden, die eine identische Anzahl an Nebenbedingungen implizieren.

Instanz		Optimum Wert	Optimum Zeit	Schranken nach 12 Std. duale	Schranken nach 12 Std. primale
CELAR06	Scip	–	–	2,96	7.141
	Cplex	–	–	0,00	9.473
CELAR07	Scip	–	–	0,00	11.466.360
	Cplex	–	–	0,00	35.108.463
CELAR08	Scip	–	–	0,00	1.762
	Cplex	–	–	*0,34	*2.210
CELAR09	Scip	15.571	1:16:48	–	–
	Cplex	15.571	0:00:44	–	–
CELAR10	Scip	31.516	0:04:32	–	–
	Cplex	31.516	0:00:04	–	–
GRAPH05	Scip	–	–	104,75	1.129
	Cplex	221	1:34:02	–	–
GRAPH06	Scip	–	–	0,07	38.828
	Cplex	–	–	0,00	42.774
GRAPH07	Scip	4.324	0:00:54	–	–
	Cplex	4.324	0:00:03	–	–
GRAPH11	Scip	–	–	1,00	290.300
	Cplex	–	–	14,65	268.908
GRAPH12	Scip	11.827	0:18:24	–	–
	Cplex	11.827	0:00:59	–	–
GRAPH13	Scip	–	–	0,00	132.508
	Cplex	–	–	0,00	370.768

Tabelle 4.6: Ergebnisse der Rechenstudie der Penalty Block Formulierung ohne erweitertem Preprocessing

Instanz		Optimum Wert	Optimum Zeit	Schranken nach 12 Std. duale	Schranken nach 12 Std. primale
CELAR06	Scip	–	–	160,26	7.645
	Cplex	–	–	45,58	4.101
CELAR07	Scip	–	–	0,00	565.035
	Cplex	–	–	*54,33	* 1.947.651
CELAR08	Scip	–	–	1,28	1.137
	Cplex	–	–	7,72	754
CELAR09	Scip	15.571	4:00:41	–	–
	Cplex	15.571	0:00:27	–	–
CELAR10	Scip	31.516	0:07:34	–	–
	Cplex	31.516	0:00:04	–	–
GRAPH05	Scip	221	0:00:40	–	–
	Cplex	221	0:00:13	–	–
GRAPH06	Scip	–	–	4.113,32	4.126
	Cplex	4.123	0:11:42	–	–
GRAPH07	Scip	4.324	0:00:42	–	–
	Cplex	4.324	0:00:03	–	–
GRAPH11	Scip	–	–	2.585,41	23.117
	Cplex	–	–	2.656,11	12.752
GRAPH12	Scip	11.827	0:06:44	–	–
	Cplex	11.827	0:00:05	–	–
GRAPH13	Scip	–	–	8.714,14	88.668
	Cplex	–	–	8.825,71	53.282

Tabelle 4.7: Ergebnisse der Rechenstudie der Penalty Block Formulierung mit erweitertem Preprocessing

4.5.2 Ergebnisse der Rechenstudie

Für jede der elf bekannten Probleminstanzen wurden mit Algorithmus 4.22 die Block-Zeilen-Spalten-Zerlegungen gefunden, jeweils auf den Daten mit einfachem Preprocessing sowie zusätzlich mit erweitertem Preprocessing. Im Durchschnitt entsteht durch das erweiterte Preprocessing pro Kante ein neuer, von den anderen verschiedener Penalty-Wert im Verglich zu den Daten mit nur einfachem Preprocessing. Insgesamt können aber trotzdem in jeder der genannten Reformulierung bis hin zur Penalty Block Formulierung durch die Anwendung des erweiterten Preprocessing die Anzahl der Nebenbedingungen verringert werden. Hierbei werden nur unmerklich mehr kontinuierliche Variablen benötig.

Tabelle 4.6 und Tabelle 4.7 enthalten die Ergebnisse der Rechenstudie mit der Penalty Block Formulierung. Ein * zeigt an, dass die Rechnung durch den Solver vorzeitig abgebrochen wurde. Es fällt auf, dass die Ergebnisse mit erweitertem Preprocessing nennenswert besser sind. Im Gegensatz zu lediglicher Anwendung des einfachen Preprocessing konnte die Instanz `GRAPH06` zusätzlich optimal gelöst werden. Außerdem konnten die unteren Schranken angehoben werden.

Im Vergleich zu den Ergebnissen mit der PCS Formulierung aus Kapitel 3 konnten genau die selben Instanzen optimal gelöst werden. Die grau markierten Werte sind Verbesserungen gegenüber der PCS Formulierung. In den nächsten beiden Abschnitten werden zwei Ansätze vorgestellt, um die Schranken der nicht gelösten Instanzen zu verbessern. Abschnitt 4.5.3 liegen hierbei die aus der Literatur bekannten Optimalwerte zugrunde und in Abschnitt führen Cliquen im Interferenz-Graphen zu höheren unteren Schranken.

4.5.3 Obere Schranken aus der Literatur verwenden

In der Literatur sind die optimalen Zielfunktionswerte für die `CELAR`- und `GRAPH`-Instanzen bekannt. Der Quelle [6] ist zusätzlich zu entnehmen, wann und mit welcher Methode die Probleminstanzen zuerst optimal gelöst wurden. Es ist im Folgenden die Idee, die Kenntnis des Optimalwertes ausnutzten.

Instanz	Optimum
CELAR06	3.389
CELAR07	343.592
CELAR08	262
CELAR09	15.571
CELAR10	31.516
GRAPH05	221
GRAPH06	4.123
GRAPH07	4.324
GRAPH11	3.080
GRAPH12	11.827
GRAPH13	10.110

Tabelle 4.8: Optimale Zielfunktionswerte

Genauer gesagt wird der optimale Zielfunktionswert als obere Schranke angesehen. Im Zuge dessen sollen zusätzliche Nebenbedingungen dabei gewisse Variablenbelegung verbieten, da diese zu einem größeren Zielfunktionswert als dem bekannten Optimum führen. Ziel ist es, den verwendeten Solver keine sehr schlechten oberen Schranken generieren zu lassen.

Angenommen, *upperbound* $\in \mathbb{Z}$ sei die im Vorhinein gesetzte obere Schranke des Zielfunktionswert. Es können sämtliche $x(v, w, p)$-Variablen auf Null gesetzt bzw. gelöscht werden, für die $p >$ *upperbound* gilt. Also:

$$x(v, w, p) = 0 \qquad \forall \{v, w\} \in E, p \in P_{\{v,w\}} \text{ mit } p > \textit{upperbound} \tag{4.55}$$

Verallgemeinert kann die Summe aller $x(v, w, p)$ für ein festes p eingeschränkt werden. Eine Auswahl von zu vielen $x(v, w, p)$-Variablen führt zu einem größeren Zielfunktionswert als die definierte obere Schranke. Es können nur maximal $\frac{\textit{upperbound}}{p}$ solcher Variablen gesetzt sein. Somit:

$$\sum_{\{v,w\} \in E} x(v, w, p) \leq \left\lfloor \frac{\textit{upperbound}}{p} \right\rfloor \qquad \forall p \in \bigcup_{\{v,w\} \in E} P_{\{v,w\}} \tag{4.56}$$

Da mit $p >$ *upperbound* für den abgerundete Quotient $\left\lfloor \frac{\textit{upperbound}}{p} \right\rfloor = 0$ gilt, ist (4.55) implizit in (4.56) enthalten. Darüberhinaus kann die Summe in der linken Seite der Ungleichungen (4.56) erweitert werden. Alle $x(v, w, q)$ mit $q \geq p$ können hinzugefügt werden, denn gesetzte $x(v, w, p)$-Variablen schränken die zusätzliche Auswahl von weiteren Variablen mit größerem Faktor in der Zielfunktion ein. Letztendlich modellieren folgende neue Nebenbedingungen die Definition der festgesetzten, oberen Schranke *upperbound*:

$$\sum_{\substack{\{v,w\} \in E \\ q \geq p}} x(v, w, q) \leq \left\lfloor \frac{\textit{upperbound}}{p} \right\rfloor \qquad \forall p \in \bigcup_{\{v,w\} \in E} P_{\{v,w\}} \tag{4.57}$$

Diese Idee wird an den fünf nicht mit der Penalty Block Formulierung optimal gelösten Instanzen `CELAR06`, `CELAR07`, `CELAR08` sowie `GRAPH11` und `GRAPH13` angewendet. Die Beobachtungen und Ergebnise mit dem Solver Scip werden im Folgenden erläutert und mit den Daten in der jeweiligen Scip-Zeile in Tabelle 4.7 verglichen.

`CELAR06` enthält keinen Penalty-Wert, der größer als der Optimalwert 3.389 ist. Somit kann keine $x(v, w, p)$-Variable im Vorfeld auf Null gesetzt bzw. gelöscht werden. Insgesamt werden der Penalty Block Formulierung in Form von (4.57) genau 111 neue Nebenbedingungen zugeführt. Es gilt für diese Instanz also $|\bigcup P_{\{v,w\}}| = 111$. Das Ergebnis des Solvers nach 12 Stunden ist jedoch ernüchternd: Es konnte erneut keine zulässige Lösung mit Optimalwert 3.389 gefunden werden und die ausgegebene untere Schranke ist mit $47{,}89$ schlechter als in Tabelle 4.7.

Bei der Instanz CELAR07 können mit der zuvor definierten, oberen Schranke von 343.592 ganze 1.539 der 4.940 x-Variablen vernachlässigt werden. Der Formulierung werden $123 = |\bigcup P_{\{v,w\}}|$ Nebenbedingungen der Form (4.57) hinzugefügt. Nach 12 Stunden konnte keine zulässige Lösung mit Optimalwert 343.592 gefunden werden und die ausgegebene untere Schranke ist mit $1{,}50$ nur minimal besser als in Tabelle 4.7.

Bei CELAR08 können wie bei CELAR06 keine der Variablen gelöscht werden. Dies liegt darin begründet, dass diese Instanz mit 262 nicht nur einen recht kleinen optimalen Zielfunktionswert hat, sondern mit $a = (4\ 3\ 2\ 1)^T$ auch sehr kleine Input Penalty-Werte hat (vgl. Abschnitt 2.4.1). Nur $14 = |\bigcup P_{\{v,w\}}|$ Nebenbedingungen (4.57) werden der Penalty Block Formulierung bei CELAR08 hinzugefügt. Nach 12 Stunden konnte keine zulässige Lösung mit Optimalwert 262 gefunden werden und die ausgegebene untere Schranke ist mit $3{,}49$ nur minimal besser als in Tabelle 4.7.

Bei den Instanzen GRAPH11 und GRAPH13 sind die in Tabelle 4.7 dokumentierten unteren Schranken im Verhältnis zu den zuvor betrachteten Instanzen gut annehmbar. In GRAPH11 kann mit der definierten, oberen Schranke eine Variable eliminiert werden. Insgesamt werden $148 = |\bigcup P_{\{v,w\}}|$ Nebenbedingungen der Form (4.57) hinzugefügt. Bei der Instanz GRAPH13 kann keine Variable vernachlässigt werden und es werden $166 = |\bigcup P_{\{v,w\}}|$ Nebenbedingungen hinzugefügt. Die Rechnungen zu GRAPH11 und GRAPH13 ergaben mit $2.578,74$ bzw. $8.699,40$ ähnliche untere Schranken wie in Tabelle 4.7.

Das Ziel, keine sehr schlechten oberen Schranken zu generieren, konnte natürlich mit diesem Vorgehen erreicht werden. Bei der Instanz CELAR07 wurde deutlich, dass viele Frequenzkombinationen bezüglich einer Optimallösung ausgeschlossen sind. Jedoch haben sich die restlichen Ergebnisse nicht nennenswert verbessert.

4.5.4 Disjunkte Cliquen in Interferenz-Graph

Besonders bei den Instanzen CELAR06, CELAR07 und CELAR08 sind die mit der Penalty Block Formulierung generierten unteren Schranken unbefriedigend. Indem der Interferenz-Graph auf knotendisjunkte Cliquen eingeschränkt wird, können bessere untere Schranken angegeben werden. Für diese Knoten kann es keine kostengünstigere Frequenzzuweisung als die optimale des reduzierten Problems geben, folglich bildet der optimale Zielfunktionswert eine untere Schranke für den ganzen Interferenz-Graphen. Auf einer einzelnen Clique, beispielsweise 6-Knoten-Clique, lässt sich das Problem schnell optimal lösen. Dies kann mit mehreren knotendisjunkten Cliquen durchgeführt werden und die Summe aller Optimalwerte ergeben eine untere Schranke für das gesamte Problem.

Im Falle der Instanz CELAR06 führen die disjunkten Cliquen

$$\{71, 72, 170, 171, 360, 361\}, \{137, 277, 303, 358, 359, 393\},$$
$$\{6, 7, 160, 161, 278, 299\}, \{134, 135, 136, 138, 291, 356\} \subset V$$

zu einer unteren Schranke von $816 + 222 + 113 + 13 = 1.164$ Zum Vergleich: Die bisherige untere Schranke lag nach 12 stündiger Rechenstudie bei $160{,}26$.

Der Interferenz-Graph der Instanz `CELAR07` enthält unter anderem die Cliquen

$$\{71, 72, 170, 171, 141, 140\}, \{136, 138, 277, 291, 303, 356\}, \{200, 281, 309, 333, 412, 414\} \subset V.$$

Diese implizieren eine verbesserte untere Schranke von $30.201 + 10.000 + 4 = 40.205$. Die bisherige lag bei nur $54{,}33$.

Im Falle der Instanz `CELAR08` führen die zwei disjunkten Cliquen

$$\{137, 358, 359, 360, 361, 393\}, \{200, 281, 309, 333, 412, 414\} \subset V$$

zu einer unteren Schranke von $10 + 7 = 17$. Zum Vergleich: Die bisherige untere Schranke lag bei $7{,}72$.

Es sei angemerkt, dass diese Methode noch zu sehr viel besseren unteren Schranken führt. Die hier angegebenen Cliquen wurden willkürlich ausgewählt und die drei Instanzen enthalten noch große Mengen weiterer Cliquen, die für diese Analyse verwendbar sind. Dieser Abschnitt soll nur aufzeigen, wie verbesserte untere Schranken leicht zu berechnen sind.

4.6 Zusammenfassung

Die grundlegende Idee der Penalty Block Formulierung lieferte die Struktur der Penalty-Matrizen, die sich aus dem Preprocessing ergeben. Es wurde ausgenutzt, dass diese Matrizen nur wenige verschiedene Penalty-Werte enthalten und diese eine gewisse Blockstruktur bilden.

Über die Erste Reformulierung und die Zweite Reformulierung wurde schließlich die Penalty Block Formulierung entwickelt. Hierbei wurde die Erste Reformulierung theoretisch mit der PCS Formulierung verglichen. Zusätzlich wurde die Theorie für den Vergleich zweier Formulierungen erweitert, sodass nun Formulierungen mit auch unterschiedlichen, ganzzahligen Variablen verglichen werden können. Es wurde bewiesen, dass die PCS Formulierung und die Erste Reformulierung für viele Instanzen gleich gut sind, bei Instanzen mit gewissen Gegebenheiten ist die PCS Formulierung die Bessere. Um die Penalty Block Formulierung umzusetzen wurde ein Algorithmus für die Blocksuche angegeben. Ergebnisse der Rechenstudien offenbarten, dass das erweiterte Preprocessing für die Penaly Block Formulierung vorteilhaft ist. Im Vergleich zu der PCS Formulierung konnten die gleichen Instanzen optimal gelöst werden und ein Teil der Schranken verbessert werden.

Es wurden anschließend zwei Ansätze angegeben, um die Schranken der nicht gelösten Instanzen zu verbessern. Indem der aus der Literatur bekannte Optimalwert als obere Schranke angesehen wurde, sollte verhindert werden, dass schlechte obere Schranken von dem Solver generiert

werden. Dies wurde durch neue Nebenbedingungen sichergestellt. Es zeigte sich, dass mit diesem Ansatz die unteren Schranken nicht angehoben werden können. Vielversprechender zeigte sich der zweite Ansatz. Hierbei wurde der Interferenz-Graph auf knotendisjunkte Cliquen engeschränkt. Die Summe der Optimalwerte auf jeder Clique bildete dann eine untere Schranke für das Gesamtproblem. Mit dieser Möglichkeit konnte die Schranke stark angehoben werden.

	Optimum		Schranken	
Instanz	Wert	Zeit	duale	primale
CELAR06	(3.389)	–	$\geq 1.164{,}00$	4.101
CELAR07	(343.592)	–	$\geq 40.205{,}00$	565.035
CELAR08	(262)	–	$\geq 17{,}00$	754
CELAR09	15.571	0:00:27	–	–
CELAR10	31.516	0:00:04	–	–
GRAPH05	221	0:00:13	–	–
GRAPH06	4.123	0:11:42	–	–
GRAPH07	4.324	0:00:03	–	–
GRAPH11	(3.080)	–	2.656,11	12.752
GRAPH12	11.827	0:00:05	–	–
GRAPH13	(10.110)	–	8.825,71	53.282

Tabelle 4.9: Ergebnisse der Penalty Block Formulierung

Die Ergebnisse der elf Probleminstanzen in Form der Penalty Block Formulierung ist in ihrer Gesamtheit Tabelle 4.9 zu entnehmen. Um die Qulität der Schranken einzuschätzen wurden die Optimalwerte in der zweiten Spalte angegeben. Abschnitt 4.5.4 zeigte, dass die unteren Schranken der ersten drei Instanzen noch angehoben werden können, dies ist mit dem Relationszeichen gekennzeichnet.

Insgesamt ist aus diesem Kapitel zu folgern, dass es gelungen ist, die Blockstruktur der Penalty-Matrizen ausnutzen und so eine neue Formulierung für das Minimum Interference Frequency Assignment anzugeben.

5 Die Frequency Assignment Formulierung

In [12] wurde im Kapitel für Vorschläge weiterführender Forschung mit der F*requency Assignment Formulierung* ein vielversprechendes, neues Modell für das MI-FAP angeführt. Diese Formulierung ist völlig losgelöst von den in den zwei vorherigen Kapiteln beschriebenen Modellen zu sehen. Der neue Ansatz sowie das ganzzahlige lineare Programm und die genaue Umsetzung ist in dem folgenden Kapitel erläutert. Insbesondere werden die Ergebnisse der Rechenstudie mit den bekannten Probleminstanzen dieses neuen Modells dargelegt.

Im Vorfeld ist anzumerken, dass im Zuge dieses neuen Modells von den eigentlichen Ausgangsdaten der Instanzen, wie sie in Abschnitt 2.4.1 erläutert wurden, ausgegangen wird und nicht von den Daten nach dem Preprocessing aus Abschnitt 3.2.1.

5.1 FA Integer Linear Programm für das MI-FAP

Mit Rückblick auf Abbildung 2.4, dem Ausschnitt von `ctr.txt` der Instanz `CELAR10`, kann in den meisten MI-FAPs zu einer gegebenen Kante $\{v, w\} \in E$ der Penalty-Wert von $d_v \in D_v$ und $d_w \in D_w$ angegeben werden als:

$$p(v, d_v, w, d_w) = \begin{cases} p_{vw} & \text{falls } |d_v - d_w| < \delta_{vw} \\ 0 & \text{sonst,} \end{cases}$$

wobei p_{vw} eine kantenabhängige Konstante und δ_{vw} der Mindestabstand der Frequenzen von v und w für interferenzfreie Datenübertragung ist. In `ctr.txt` ist p_{vw} durch die Zahl in der sechsten Spalte codiert und d_{vw} zumeist in der fünften Spalte angegeben. Somit ist die Strafe, abhängig von der Kante und der Distanz zwischen den zugewiesenen Frequenzen, entweder Null oder ein kostanter, positiver Wert.
Verallgemeinert können für jede Kante $\{v, w\} \in E$ jeweils n_{vw} Intervalle der Form $[l^i_{vw}, u^i_{vw}]$ für die Distanz zwischen den Frequenzen definiert werden. Diese Intervalle überdecken alle möglichen Differenzen, dabei sei $l^1_{vw} = -\infty$ sowie $u^{n_{vw}}_{vw} = \infty$ und $l^i_{vw} = u^{i-1}_{vw}$ für $i = 2, \ldots, n_{vw}$. Bei der Implementierung dieses Modells kann hier ∞ als der maximal mögliche Abstand $span_{\max}$ zwischen zwei Frequenzen einer Instanz gesetzt werden. Für jedes Intervall sei der Penalty-Wert fest:

$$p(v, d_v, w, d_w) = p^i_{vw} \qquad \text{falls } l^i_{vw} \leq d_v - d_w \leq u^i_{vw}, i = 1, \ldots, n_{vw} \tag{5.1}$$

In den `CELAR`- und `GRAPH`-Instanzen ist $span_{\max} = 800$ eine passende Wahl. In diesen Instanzen existieren zwei verschiedene Arten von Bedingungen und somit Kanten, wie sie in `ctr.txt` geführt werden. Zum Einen die $=$-Bedingungen, die unter allen Umständen genau eine Differenz zwischen zwei zugewiesenen Frequenzen fordern, und zum Anderen die $>$-Bedingungen,

die einen Mindestabstand zwischen den jeweiligen Frequenzen verlangt. Nichterfüllung einer $>$-Bedingung impliziert eine Strafe. Übertragen auf (5.1) wird eine $=$-Bedingung $|d_v - d_w| = \delta_{vw}$ formuliert als:

$$p(v, d_v, w, d_w) = \begin{cases} 0 & \text{falls } -\delta_{vw} \leq d_v - d_w \leq -\delta_{vw} \\ 0 & \text{falls } \delta_{vw} \leq d_v - d_w \leq \delta_{vw} \\ \mathit{BigM} & \text{sonst} \end{cases}$$

Hierbei ist *BigM* verhältnismäßig groß zu wählen. Bei einer Implementierung kann der *BigM*-Fall vernachlässigt werden. Eine Bedingung $|d_v - d_w| > \delta_{vw}$ wird formuliert als:

$$p(v, d_v, w, d_w) = \begin{cases} 0 & \text{falls } -\mathit{span}_{\text{max}} \leq d_v - d_w \leq -\delta_{vw} - 1 \\ p_{vw} & \text{falls } -\delta_{vw} \leq d_v - d_w \leq \delta_{vw} \\ 0 & \text{falls } \delta_{vw} + 1 \leq d_v - d_w \leq \mathit{span}_{\text{max}} \end{cases}$$

Das MI-FAP kann als ein ganzzahliges lineares Programm modelliert werden, welches die Vorzüge dieser Struktur ausnutzt. Dazu seien für jede Kante $\{v, w\} \in E$ und jedes $i = 1, \ldots, n_{vw}$ eine binäre Variable x^i_{vw} definiert.

$$x^i_{vw} = \begin{cases} 1 & \text{falls die Frequenzen } d_v \in D_v, d_w \in D_w \text{ mit } l^i_{vw} \leq d_v - d_w \leq u^i_{vw} \text{ zugewiesen} \\ 0 & \text{sonst} \end{cases}$$

Außerdem wird eine binäre Variable y^j_v für alle $v \in V$ und $j \in \{1, \ldots, |D_v|\}$ definiert.

$$y^j_v = \begin{cases} 1 & \text{falls } v \in V \text{ die } j\text{-te Frequenz aus } D_v \text{ zugewiesen} \\ 0 & \text{sonst} \end{cases}$$

Frequency Assignment Formulierung (FA)

$$\textbf{min} \sum_{\{v,w\} \in E} \sum_{i=1}^{n_{vw}} p^i_{vw} x^i_{vw} + \sum_{v \in V} \sum_{j=1}^{|D_v|} q^j_v y^j_v \tag{5.2}$$

$$\textbf{s.t.} \sum_{n_{vw}}^{i=1} x^i_{vw} = 1 \qquad \forall \{v, w\} \in E \tag{5.3}$$

$$\sum_{j=1}^{|D_v|} y^j_v = 1 \qquad \forall v \in V \tag{5.4}$$

$$d_v = \sum_{j=1}^{|D_v|} d^j_v y^j_v \qquad \forall v \in V \tag{5.5}$$

$$d_w \geq d_v - \sum_{i=1}^{n_{vw}} u^i_{vw} x^i_{vw} \qquad \forall \{v, w\} \in E \tag{5.6}$$

$$d_v \geq d_w + \sum_{i=1}^{n_{vw}} l^i_{vw} x^i_{vw} \qquad \forall \{v, w\} \in E \tag{5.7}$$

$$d_v \in \mathbb{Z} \qquad \forall v \in V \tag{5.8}$$

$$y^j_v \in \{0, 1\} \qquad \forall v \in V, j = 1, \ldots, |D_v| \tag{5.9}$$

$$x^j_{vw} \in \{0, 1\} \qquad \forall \{v, w\} \in E, i = 1, \ldots, n_{vw} \tag{5.10}$$

Schließlich werden ganzzahlige Variablen $d_v \in \mathbb{Z}$ verwendet, um die dem Knoten $v \in V$ zugewiesene Frequenz zu benennen. In dem Programm steht q_v^j für die Kosten bei Wahl der j-ten Frequenz für $v \in V$ und d_v^j für die eigentliche j-te Frequenz in D_v mit $1 \leq j \leq |D_v|$.

Die Zielfunktion (5.2) entspricht der Summe der Penalty-Werte für Frequenzkombinationen und einzelnen Frequenzen. Die Nebenbedingungen (5.3) modellieren, dass für jede Kante $\{v, w\} \in E$ der Abstand zwischen den beiden zugewiesenen Frequenzen in genau einem Intervall $[l_{vw}^i, u_{vw}^i]$ enthalten ist. Außerdem fordern die Nebenbedingungen (5.4), dass einem Knoten genau eine Frequenz zugewiesen wird, nämlich d_v in (5.5). Für eine nach (5.3) zulässige Belegung der Variablen x_{vw}^i muss für die Frequenzzuweisung $l_{vq}^i \leq d_v - d_w \leq u_{vw}^i$ im Falle von $x_{vw}^i = 1$ für alle $\{v, w\} \in E$ gelten. Die Bedingung kann für jede Kante $\{v, w\} \in E$ in $d_w \geq d_v - u_{vw}^i$ und $d_v \geq d_w + l_{vw}^i$ aufgeteilt werden. Genau diese beiden Bedingungen sind in Verbindung mit x_{vw}^i in (5.6) und (5.7) modelliert.

Sämtliche d_v-Variablen können auch kontinuierlich gewählt werden. Darüberhinaus ist es möglich auf diese ganz zu verzichten, indem d_v in (5.6) und (5.7) durch die Summe aus (5.5) ersetzt wird.

	Anzahl Variablen			Anzahl
Instanz	Summe	kontinuierliche	binäre	Nebenbedingungen
CELAR06	12.086	200	11.886	4.366
CELAR07	24.747	400	24.347	9.395
CELAR08	53.890	916	52.974	19.064
CELAR09	39.505	680	38.825	13.669
CELAR10	39.505	680	38.825	13.669
GRAPH05	10.918	200	10.718	3.802
GRAPH06	21.794	400	21.394	7.310
GRAPH07	21.370	400	20.970	7.310
GRAPH11	37.251	680	36.571	12.631
GRAPH12	37.955	680	37.275	13.411
GRAPH13	51.453	916	50.537	17.651
SMALL02	340	8	332	100

Tabelle 5.1: Anzahl Variablen und Nebenbedingungen der Frequency Assignment Formulierung

Die Tabelle 5.1 enthält für die bekannten Instanzen die Anzahl der Variablen und Nebenbedingungen für die Frequency Assignment Formulierung. Zusätzlich sind die Daten einer selbstkreierten, kleineren Instanz SMALL02 aufgetragen. Für den Interferenz-Graphen (V, E) der Instanz SMALL02 gilt $|V| = 8$ und $|E| = 28$. Die Größen der anderen Instanzen sind aus Tabelle 3.1 bekannt.

Ein genauer Vergleich der Anzahl der Variablen und Nebenbedingungen mit der PCS Formulierung aus Kapitel 3 sowie mit der in Kapitel 4 neu entwickelten Penalty Block Formulierung ist in dem letzten Kapitel 6 dieser Arbeit mit Tabelle 6.2 möglich.

5.1.1 Ergebnisse der Rechenstudie

Tabelle 5.2 enthält die Ergebnisse der Rechenstudie der Frequency Assignment Formulierung. Diese sind hauptsächlich mangelhaft, nur die selbstkreierte Instanz `SMALL02` konnte gelöst werden. Zum Vergleich: In Form der Penalty Block Formulierung (126 binäre, 61 kontinuierliche Variablen und 823 Nebenbedingungen) wurde `SMALL02` mit Scip in 1,56 Sekunden und mit Cplex in 0,84 Sekunden optimal gelöst. Zum größten Teil konnte für die anderen Instanzen gar keine zulässige, ganzzahlige Lösung gefunden werden.

		Optimum		Schranken nach 12 Std.	
Instanz		Wert	Zeit	duale	primale
CELAR06	Scip	–	–	0,00	31.598
	Cplex	–	–	0,00	∞
CELAR07	Scip	–	–	0,00	93.709.495
	Cplex	–	–	0,00	∞
CELAR08	Scip	–	–	0,00	4.207
	Cplex	–	–	0,00	∞
CELAR09	Scip	–	–	2.708,95	∞
	Cplex	–	–	5.479,98	∞
CELAR10	Scip	–	–	6.962,21	∞
	Cplex	–	–	10.586,38	∞
GRAPH05	Scip	–	–	1,17	29.995
	Cplex	–	–	*53,39	*∞
GRAPH06	Scip	–	–	0,00	113.910
	Cplex	–	–	*567,86	*∞
GRAPH07	Scip	–	–	313,82	255.376
	Cplex	–	–	*2.383,98	*∞
GRAPH11	Scip	–	–	0,00	237.566
	Cplex	–	–	*1001,00	*∞
GRAPH12	Scip	–	–	2.674,65	∞
	Cplex	–	–	*7.068,67	*∞
GRAPH13	Scip	–	–	0,00	365.954
	Cplex	–	–	*1.003,78	*∞
SMALL02	Scip	113	0:00:04	–	–
	Cplex	113	0:00:01	–	–

Tabelle 5.2: Ergebnisse der Rechenstudie der Frequency Assignment Formulierung

Eine Markierung mit * zeigt erneut an, dass diese Rechnung aufgrund eines Speicherfehlers vorzeitig durch den Solver abgebrochen wurde.

Ein Vergleich mit den Ergebnissen der Rechnerstudie der anderen in dieser Arbeit behandelten Formulierungen ist nicht angebracht. Hierfür sind diese oberen und unteren Schranken schlichtweg zu schlecht. Die gegenüber den anderen Formulierungen mehr als verdoppelte Anzahl an

binären Variablen scheint die Performance der Frequency Assigment Formulierung so sehr zu schwächen, dass die immens geringe Anzahl an Nebenbedingungen nicht viel bewirken kann.

Folglich ist die Implementierung dieses Modells in seiner rohen Form nicht viel wert. Wie auch im letzten Kapitel dieser Arbeit kurz angeschnitten, scheint es trotzdem vielversprechend zu sein, das Polyeder dieser Formulierung zu studieren, um etwaige Facetten zu finden. Dies würde den Ansatz der Frequency Assignment Formulierung auf jeden Fall verbessern.

6 Zusammenfassung und Ausblick

Nachdem das Problem der Frequenzzuweisung inklusive des in dieser Arbeit betrachteten Minimum Interference Frequency Assignment Problems eingeführt wurde, folgte die Vorstellung von drei Formulierungen. Die Partial Constraint Satisfaction Formulierung sowie die Frequency Assignment Formulierung wurden dabei [12] entnommen. Die Penalty Block Formulierung wurde in dieser Bachelorarbeit entwickelt. Es ist gelungen, die Blockstruktur der Penalty-Matrizen auszunutzen und so eine neue Formulierung für das MI-FAP anzugeben. Es zeigte sich, dass die neue Formulierung und die PCS Formulierung in der Theorie und auch in den Ergebnissen vergleichbar sind. Neben der detaillierten Dokumentation der Entwicklung dieser Formulierung wurden gültige Ungleichungen und zwei Ansätze zur Verbesserung der Schranken angegeben. Die Frequency Assignment Formulierung steht in der genannten Form in einem Vergleich mit den beiden anderen Formulierungen außen vor. Die gelieferten Ergebnisse hierfür sind schlichtweg zu schlecht.

Abschließend enthält Tabelle 6.1 die besten Ergebnisse der Rechenstudien und manuellen Analysen der elf Probleminstanzen sowie Tabelle 6.2 auf Seite 65 eine Übersicht der jeweiligen Anzahl an Variablen und Nebenbedingungen aller behandelten Formulierungen.

	Optimum		Schranken	
Instanz	Wert	Zeit	duale	primale
CELAR06	(3.389)	–	1.164,00	4.101
CELAR07	(343.592)	–	40.205,00	565.035
CELAR08	(262)	–	48,49	754
CELAR09	15.571	0:00:27	–	–
CELAR10	31.516	0:00:04	–	–
GRAPH05	221	0:00:13	–	–
GRAPH06	4.123	0:01:40	–	–
GRAPH07	4.324	0:00:03	–	–
GRAPH11	(3.080)	–	2.861,02	12.509
GRAPH12	11.827	0:00:05	–	–
GRAPH13	(10.110)	–	9.477,48	53.282

Tabelle 6.1: Ergebnisse aller Rechenstudien und manuellen Analysen

Die folgenden Ausführungen stellen einen Ausblick dar. Während dieser Bachelorarbeit ergaben sich weitere Arbeitsansätze, die aufgrund des eingeschränkten Umfangs und der begrenzten Zeit nicht weiter verfolgt wurden.

Wie schon angesprochen, liefert die Frequency Assignment Formulierung aus Kapitel 5 in ihrer dort genannten, rohen Form lediglich mangelhafte Schranken für die Instanzen. Diese Formulierung zeichnet sich durch die sehr geringe Anzahl an benötigten Nebenbedingungen aus. Die Analyse der polyedrischen Struktur sowie des zugrundeliegenden Polytops wird Möglichkeiten zur Verbesserung dieser Formulierung liefern.

Im Zusammenhang mit der Penalty Block Formulierung kann noch viel Anstrengung in den Algorithmus zur Blocksuche investiert werden. Der beste Algorithmus würde die optimale Block-Zeilen-Spalten-Zerlegung liefern. Da immer wiederkehrende Strukturen (Treppen, etc.) in den Matrizen vorkommen, ist es auch möglich diese Bereiche einzeln zu analysieren, wie dort die Blöcke optimal zu wählen sind. Es wäre sehr interessant zu sehen, ob und in welchem Ausmaß sich die Ergebnisse einer Rechenstudie mit optimaler oder zumindest verbesserter Block-Zeilen-Spalten-Zerlegung ändern würden.
Außerdem kann für verbesserte untere Schranken die Auswahl von knotendisjunkten Cliquen im Interferenz-Graph, wie es in Abschnitt 4.5.4 dokumentiert wurde, zielführender durchgeführt werden. Optimalerweise werden die Cliquen gewählt, die den größten Teil des Optimalwertes des gesamten Problems implizieren.
Nach dem einfachen Preprocessing enthalten die Penalty-Matrizen als Penalty-Werte lediglich Linearkombinationen der vier Einträge des a-Vektors, welcher im Abschnitt 2.4.1 erläutert wurde. Diese Gegebenheit führt zu folgender modifizierten Penalty Block Formulierung: Sei $a = (a_1 \;\; a_2 \;\; a_3 \;\; a_4)^T$ der Vektor der Kosten bei Nicht-Erfüllung einer Bedingung. Für eine Kante $\{v, w\} \in E$ in dem Graphen nach einfachem Preprocessing und $i = 1, \ldots, 4$ seien $n_i^{v,w}$ die Anzahl der mit a_i als Penalty beschrifteten Kanten in dem in Abbildung 3.3 beispielhaft dargestellten Multigraphen. Folglich ist jeder Eintrag der zugehörigen Penalty-Matrix kleiner oder gleich $\sum_{i=1}^{4} n_i^{v,w} \cdot a_i$. Für alle $i = 1, \ldots, 4$ und $j = 1, \ldots n_i^{v,w}$ werden Variablen

$$x(v, w, a_i, j) = \begin{cases} 1 & \text{falls } (d_v, d_w) \in D_v \times D_w \text{ gewählt,} \\ & \text{mit } p(v, d_v, w, d_w) \text{ enthält } j \text{ Summanden } a_i \\ 0 & \text{sonst} \end{cases}$$

eingeführt. Dabei enthält $p(v, d_v, w, d_w)$ in der Variablendefinition j Summanden a_i, falls im Sinne des Multigraphen in Abbildung 3.3 mindestens j Bedingungen mit Penalty a_i verletzt werden. Es gilt somit für $j > 0$ die Implikation

$$x(v, w, a_i, j + 1) = 1 \implies x(v, w, a_i, j) = 1.$$

Falls $n_i^{v,w} = 0$ gilt, wird für dieses i keine Variable benötigt. Die neuen $x(v, w, a_i, j)$-Variablen werden wieder durch Block-, Zeilen- und Spaltennebenbedingungen wie (4.45) - (4.47) modelliert. Dazu gibt es pro Kante $\{v, w\} \in E$ für jedes mögliche Vielfache des Summanden a_i eine Matrix, die nur Einträge aus $\{0, a_i\}$ enthält. In der Summe bilden diese Matrizen dann die eigentliche Penalty-Matrix. Pro Matrix und a_i werden dann Block-Zeilen-Spalten-Zerlegungen generiert, um die Nebenbedingungen anzugeben. Die Zielfunktion für diese Formulierung lautet:

$$\textbf{min} \quad \sum_{\{v,w\} \in E} \sum_{i=1}^{4} \sum_{j=1}^{n_i^{v,w}} a_i \cdot x(v, w, a_i, j) + \sum_{v \in V} \sum_{d_v \in D_v} q(v, d_v) \cdot y(v, d_v)$$

Die Partial Constraint Satisfaction Formulierung wurde in [12] nicht nur im Sinne des MI-FAP eingeführt, sondern auch detailliert analysiert. Dabei wurden für das Partial Constraint Satisfaction Polytop $X(PCSP) := conv\{(y,z) : (y,z) \text{ erfüllt (3.4)} - \text{(3.7)}\}$ nicht-triviale Klassen von Facetten gefunden: die Kreis-Ungleichungen und die Clique-Kreis-Ungleichungen. Diese sind jeweils charakterisiert durch induzierte Untergraphen $G[C] = (C, E[C])$ des Interferenz-Graphen $G = (V, E)$ mit $C \subseteq V$. Außerdem sei für jedes $v \in V$ weiter A_v, B_v eine Partition der Menge D_v und seien $y(v, D'_v) := \sum_{d_v \in D'_v} y(v, d_v)$ und $z(v, D'_v, w, D'_w) := \sum_{d_v \in D'_v} \sum_{d_w \in D'_w} z(v, d_v, w, d_w)$ definiert. Die Beweise der nun folgenden Aussagen ist [12] zu entnehmen.

Satz 6.1.
Kreis-Ungleichungen
Sei der induzierte Teilgraph $G[C] = (C, E[C])$ von $G = (V, E)$ ein sehnenloser k-Kreis. D.h. $C = \{v_1, \ldots, v_k\} \subseteq V$ und $E[C] = \{\{v_i, v_{i+1}\} : i = 1, \ldots, k-1\} \cup \{\{v_k, v_1\}\}$. Dann ist die k-Kreis-Ungleichung

$$\sum_{i=1}^{k-1} \Big(z(v_i, A_{v_i}, v_{i+1}, A_{v_{i+1}}) + z(v_i, B_{v_i}, v_{i+1}, B_{v_{i+1}}) \Big) + z(v_1, A_{v_1}, v_k, B_{v_k}) + z(v_1, B_{v_1}, v_k, A_{v_k} \leq k - 1$$

mit $k \geq 3$ gültig und facettendefinierend für $X(PCSP)$.

Satz 6.2.
Clique-Kreis-Ungleichungen
Sei der induzierte Teilgraph $G[C] = (C, E[C])$ von $G = (V, E)$ eine k-Clique. Dann ist die k-Clique-Kreis-Ungleichung

$$\sum_{v \in C} y(v, A_v) + \sum_{\{v,w\} \in E[C]} z(v, B_v, w, B_w) \geq k - 1$$

mit $k \geq 3$ und $k + 1 := 1$ gültig und facettendefinierend für $X(PCSP)$.

Diese k-Clique-Kreis-Ungleichungen können noch zu (γ, k)-Clique-Kreis-Ungleichungen erweitert werden [12].
Beide genannten Klassen enthalten exponentiell viele Facetten, sodass eine Überprüfung aller Ungleichungen zum Auffinden einer verletzten nicht möglich ist. Abhilfe leisten die zugehörigen Separierungsprobleme: Zu einem gegebenen Vektor $\tilde{x}$ soll entweder eine mit diesem Vektor verletzte Ungleichung gefunden werden oder gefolgert werden, dass alle Ungleichungen für diesen Vektor erfüllt sind. Neben exakten Lösungsmethoden für die Separierungsprobleme sind in [12] auch Heuristiken für diese Probleme angeführt.
Ein nächster Schritt im Zusammenhang mit der PCS-Formulierung ist es, die bekannten Facetten miteinzubeziehen. Die Heuristiken sowie exakten Methoden der Separierung der oben genannten Ungleichungen können implementiert werden und Rechenstudien an den bekannten Probleminstanzen durchführen werden. Die zusätzlichen, speziellen Cuts werden sicher zu verbesserten Ergebnissen führen.

Instanz		Anzahl Variablen			Anzahl Nebenbedingungen
		Summe	kontinuierliche	binäre	
CELAR06	PCS	585.914	581.904	4.010	28.628
	PB	6.234	2.224	4.010	44.383
	FA	12.086	200	11.886	4.366
CELAR07	PCS	1.331.048	1.323.072	7.976	65.928
	PB	12.916	4.940	7.976	98.029
	FA	24.747	400	24.347	9.395
CELAR08	PCS	2.518.664	2.500.564	18.100	128.886
	PB	26.247	8.147	18.100	180.508
	FA	53.890	916	52.974	19.064
CELAR09	PCS	1.793.484	1.780.056	13.428	89.970
	PB	20.487	7.059	13.428	137.432
	FA	39.505	680	38.825	13.669
CELAR10	PCS	1.793.484	1.780.056	13.428	89.970
	PB	19.299	5.871	13.428	135.607
	FA	39.505	680	38.825	13.669
GRAPH05	PCS	574.628	570.920	3.708	30.986
	PB	5.440	1.732	3.708	30.608
	FA	10.918	200	10.718	3.802
GRAPH06	PCS	1.201.638	1.194.096	7.542	63.846
	PB	10.875	3.333	7.542	63.424
	FA	21.794	400	21.394	7.310
GRAPH07	PCS	1.146.426	1.139.096	7.330	62.290
	PB	10.707	3.377	7.330	61.049
	FA	21.370	400	20.970	7.310
GRAPH11	PCS	2.032.792	2.019.972	12.820	107.688
	PB	18.700	5.880	12.820	112.567
	FA	37.251	680	36.571	12.631
GRAPH12	PCS	1.805.190	1.792.408	12.782	95.408
	PB	19.548	6.766	12.782	119.881
	FA	37.955	680	37.275	13.411
GRAPH13	PCS	2.798.400	2.780.812	17.588	145.034
	PB	25.755	8.167	17.588	163.801
	FA	51.453	916	50.537	17.651

Tabelle 6.2: Anzahl der Variablen und Nebenbedingungen aller behandelten Formulierungen

Quellen

[1] K. I. AARDAL, S. P. M. VAN HOESEL, A. M. C. A. KOSTER, C. MANNINO, A. SASSANO, *Models and Solution Techniques for Frequency Assignment Problems*, 2001

[2] R. K. AHUJA, T. L. MAGNANTI, J.B. ORLIN, *Network Flows: Theory, Algorithms, and Applications*, Prentice Hall, 1993

[3] T. CHRISTOF, A. LÖBEL, *PORTA POlyhedron Representation Transformation Algorithm*, Konrad-Zuse-Zentrum für Informationstechnik Berlin (ZIB), 1997-2009, `http://typo.zib.de/opt-long_projects/Software/Porta/index.html`

[4] G. CLASSEN, A. M. C. A. KOSTER, M. KUTSCHKA, C. RAACK, *MIP Interface*, Konrad-Zuse-Zentrum für Informationstechnik Berlin (ZIB), Lehrstuhl II für Mathematik RWTH Aachen

[5] S. A. COOK, *The complexity of theorem-proving procedures*, Proceedings of ACM STOC'71, Seite 151-158, 1971.

[6] A. EISENBLÄTTER, A. M. C. A. KOSTER, *FAP web - A website devoted to frequency assignment*, 2000-2011, `http://fap.zib.de`

[7] A. GAMST, W. RACE, *On frequency assignment in mobile automatic telephone systems*, Proceedings of GLOBECOM'82, 1982

[8] F. GLOVER, D. KLINGMAN, N. V. PHILLIPS, *Network Models in Optimization and Their Applications in Practice*, Wiley-Interscience, 1992

[9] E. GRÄDEL, *Mathematische Logik*, Vorlesungsskript 2010, `logic.rwth-aachen.de`, Mathematische Grundlagen der Informatik, RWTH Aachen

[10] *IBM ILOG CPLEX Optimization Studio*, IBM, 1988 - 2010, Version 12.2, `http://ibm.com/software/integration/optimization/cplex-optimization-studio`

[11] A. KOHL, *Aus der Forschung - Frequenzzuweisung in Mobilfunknetzen*, Freiberger Mathematische SemesterBlätter, 2002/2003, Seite 3-6

[12] A. M. C. A. KOSTER, *Frequency Assignment - Models and Algorithms*, Dissertation, Universität Maastricht, 1999

[13] S. O. KRUMKE, J. RAMBAU, *Online Optimierung*, Vorlesungsskript, Technische Universität Berlin, 2005

[14] *LEMON Library for Efficient Modeling and Optimization in Networks*, Eötvös Loránd University Budapest, Budapest University of Technology and Economics, 2003-2011, `http://lemon.cs.elte.hu`

[15] *SCIP Solving Constraint Integer Programs*, Konrad-Zuse-Zentrum für Informationstechnik Berlin (ZIB), 2002-2011, Version 2.0.1, `http://scip.zib.de`

[16] T. SCHIEX, *The CELAR Radio Link Frequency Assignment Problems*, INRA Toulouse, Département de Mathématiques et Informatique Appliquées, `http://www.inra.fr/mia/T/schiex/Doc/CELAR.shtml`, zuletzt aufgerufen: 05.09.2011

[17] L. A. WOLSEY, *Integer Programming*, Wiley-Interscience, 1998

[18] H-J. ZIMMERMANN, *Operations Research*, Vieweg Friedr. + Sohn Verlag, 2008