Brandenburg
University of Technology
Cottbus - Senftenberg

Eine Anwendung der ganzzahligen Optimierung auf die Stundenplanerstellung einer Unteroffiziersschule der Bundeswehr

Tabea Werger

Cottbus Mathematical Preprints
COMP # 2(2019)

Brandenburgische Technische Universität
Cottbus-Senftenberg
Fakultät 1 - MINT
Institut der Mathematik
Lehrstuhl Ingenieurmathematik und
Numerik der Optimierung
Prof. Dr. Armin Fügenschuh

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Bachelorarbeit

Eine Anwendung der ganzzahligen Optimierung auf die Stundenplanerstellung einer Unteroffiziersschule der Bundeswehr

An application of integer programming on a timetabling problem of a non-commissioned officer school of the federal armed forces

---

vorgelegt von: Tabea Werger
Matrikelnummer: 3436054
Studiengang: Wirtschaftsmathematik

Gutachter: Prof. Dr. Armin Fügenschuh
Zweitgutachter: Dr. Dietmar Kunde

Abgabedatum: 26. November 2018

Inhaltsverzeichnis

1 Einleitung

Das Timetabling-Problem ist ein sehr komplexes, diskretes Optimierungsproblem, das nicht nur im Bildungswesen Anwendung findet. Es spielt eine wichtige Rolle in unterschiedlichen Bereichen des Alltags, zum Beispiel beim öffentlichen Transportsystem, beim Erstellen eines Schichtplans oder beim Ablauf eines sportlichen Events. Der Grundgedanke ist hierbei jedoch immer der gleiche, eine möglichst gute Zuordnung der Ressourcen (Lehrer, Schüler, Räume, etc.) in einem kurzfristigen Planungszeitraum zu finden.
Stundenpläne werden auch in einer so digitalisierten Welt wie der heutigen oft noch manuell erstellt. Durch Probieren und einen geschulten Instinkt ist ein Stundenplan für ein kleine Einrichtung (z.B. für eine Grundschule) in wenigen Tagen entworfen. Handelt es sich jedoch um die Erstellung eines Stundenplans für ein Gymnasium oder eine Universität, ist es fast unmöglich, das Timetabling-Problem per Hand zu lösen. Aus diesem Grund beschäftigen sich viele Wissenschaftler mit dem Problem der Stundenplanoptimierung.
Bereits 1995 fand die erste PATAT-Konferenz (Practical and Theory of Automated Timetabling) statt. Hier wurden die unterschiedlichen Richtungen und Ergebnisse der Forschungen des Timetabling-Problems vorgestellt. Es wird jährlich eine Vielzahl von Paper veröffentlicht, die sich mit der praktischen und theoretischen Erstellung von Stundenplänen befassen. Ziel ist es, mit Hilfe eines Programms zulässige Pläne zu erstellen. Die Wissenschaft gerät hier jedoch immer wieder an ihre Grenzen.
Oft wird mit der Methode von Heuristiken gearbeitet, die aber nicht zwingend zu einer optimalen Lösung führt, da nicht der gesamte Lösungsraum betrachtet wird.
Die existierenden Programme unterstützen bei der Erstellung eines Stundenplans, sind aber meist unflexibel und schwer zu bedienen. Vor allem bei Änderungen von Gesetzen im Bildungswesen ist es für die Anwender der Optimierungspro-

gramme oft nicht möglich, die neuen Bedingungen in das Programm einzuarbeiten.
Zusätzlich muss das Timetabling-Programm in einer angemessenen Zeit einen zulässigen Stundenplan erstellen. Um die Laufzeiten der Programme zu minimieren, wird das Problem oft in mehrere Einzelprobleme unterteilt. Viele Wissenschaftler betrachten die Raumplanung als ein Extraproblem, das dann in einem zweiten separaten Durchlauf gelöst wird.
Für die Optimierung der Stundenpläne wird das Timetabling-Problem zum Beispiel als Constraint Satisfaction Problem oder als ganzzahliges-lineares Optimierungsproblem definiert. Es kommt auch zu Kombinationen mehrerer Modellierungen.

1.1 Problemstellung

Die vorliegende Bachelorarbeit beschäftigt sich mit der Erstellung eines möglichst guten, zulässigen Stundenplans für eine Unteroffiziersschule der Bundeswehr. Die Bildungseinrichtung kann vorhandene Optimierungsprogramme für Stundenpläne nur teilweise zu Rate ziehen, da hier im Vergleich zu normalen Schulen und Universitäten die Veranstaltungen nicht in einem Wochenzyklus stattfinden. Des Weiteren gibt es Spezialveranstaltungen, die nur an bestimmten Tagen besucht werden können. Zusätzlich haben die zu absolvierenden Kurse eine fest vorgegebene Reihenfolge, die beachtet werden muss.
Die bisherige Herangehensweise der Bundeswehr bei der Lösung des Timetabling-Problems nimmt viel Zeit in Anspruch. Die Erstellung des Stundenplans dauert in der Regel 7-8 Monate und wird mit Hilfe eines Stufenmodells bewerkstelligt. Das heißt, dass die Module unterschiedlich gewichtet und nach dieser Reihenfolge im Stundenplan berücksichtigt werden. Des Weiteren erfolgt die Erstellung durch eine Wasserfallanordnung, was bedeutet, dass die Kurse zeitversetzt in der gleichen Reihenfolge in den unterschiedlichen Klassen wiederholt werden.
Der Arbeitsaufwand für die Erstellung eines zulässigen Stundenplans soll in Zukunft minimiert werden. Ziel ist es daher, einen Stundenplan computergestützt, möglichst automatisch erstellen zu lassen.
Am Ende dieser Arbeit soll mit Hilfe der ganzzahligen Programmierung ein zulässiger Stundenplan für die Unteroffiziersschule entwickelt werden. Punkte, die

bei der Umsetzung beachtet werden müssen, sind die Akzeptanz der Lösung, die Handhabbarkeit des Programms sowie dessen Laufzeit.

1.2 Vorgehensweise

Die Bachelorarbeit wird sich im folgenden Kapitel mit dem derzeitigen Forschungsstand der Timetabling-Probleme auseinandersetzen. Hierfür werden unter anderem einige Paper, die bei den PATAT-Konferenzen veröffentlicht wurden, aufgearbeitet und analysiert. Ziel ist es, mögliche Lösungsansätze für das Timetabling-Problem der Unteroffiziersschule der Bundeswehr zu finden.
Im dritten Kapitel wird auf den mathematischen Hintergrund der verwendeten Verfahren der ganzzahligen Programmierung eingegangen. Es werden Techniken und Algorithmen zur Lösung des Timetabling-Problems vorgestellt.
Anschließend werden die Input-Daten der Bundeswehr aufgearbeitet, die dann im folgenden Kapitel in mathematische Formulierungen modelliert werden.
Zum Abschluss werden die Ergebnisse vorgestellt und diskutiert sowie Vorschläge für zukünftige Lösungsansätze gegeben.

2 Literaturrecherche

Es gibt viele bahnbrechende Erfindungen, Entdeckungen und Entwicklungen, die die Menschheit in ihrem Dasein fundamental beeinflussten. Eine der wichtigsten aus heutiger Sicht ist die lineare Optimierung. Mit der Begründung und Entwicklung dieses Gebiets ist der Name Georg B. Dantzig unlösbar verbunden [7]. In einem von ihm veröffentlichten Paper beschreibt Dantzig die Entwicklung der linearen Programmierung als revolutionär, da es mit ihr möglich ist, Probleme von großer Komplexität zu lösen [3].
Die Entwicklung des linearen Programmierens begann während des Zweiten Weltkriegs. Wie viele andere Entwicklungen auch, wurde sie maßgeblich durch die militärische Forschung vorangetrieben [3]. Im Jahr 1947 wurden die Untersuchungen der linearen Optimierung zusammengetragen und festgehalten. Mit Hilfe dieser Entdeckungen ist es nun möglich, Probleme explizit zu formulieren und ihre Lösungen wirkungsvoll zu bestimmen [7]. Wissenschaftler wie von Neumann, Kantorovich, Leontief und Koopmanns waren führende Köpfe auf diesem Gebiet [3].
Ein weiterer bahnbrechender Meilenstein auf dem Weg zur Optimierung komplexer Sachverhalte war die Entwicklung des Computers, die ebenfalls maßgeblich durch das Pentagon forciert wurde. Mathematiker und Wirtschaftswissenschaftler waren fasziniert von den Möglichkeiten, die sich ihnen von nun an boten [3]. Computer wurden ein schnell notwendiges Hilfsmittel bei der Lösung komplexer linearer Probleme [7].
Dantzig präsentierte sein Modell des linearen Programmierens erstmals auf einem Treffen der Economic Society in Wisconsin. Seine Erkenntnis zu jener Zeit beschreibt er folgendermaßen: „The world is highly nonlinear. Fortunately, systems of linear inequalities ... permit us to approximate most kinds of nonlinear relations encountered in practical planning.“ [3].
Das Jahr 1949 stellt einen weiteren Höhepunkt dar. Zahlreiche namhafte Forscher

stellten auf einer Konferenz – laut Dantzig das Zero Symposium – die Ergebnisse ihrer zweijährigen Forschungen auf dem Gebiet des linearen Programmierens vor [3]. Die betrachteten Probleme kamen aus unterschiedlichen Bereichen. Dazu gehörten zum Beispiel die Planung wechselnder Bodenbewirtschaftung, die Planung großräumiger militärischer Aktionen oder die Festlegung von Schiffsrouten [7]. Dantzig betont die immense Bedeutung dieser Resultate. „The research accomplished in exactly two years is, in my opinion, one of the remarkable events of history. The proceedings of the conference remain to this very day an important basic reference, a classic!" [3].
Der Grundstein der linearen Programmierung war somit gesetzt. Das Forschungsgebiet entwickelte sich in unterschiedliche Richtungen. Ein Teilgebiet ist die Optimierung der Planung.

2.1 Wissenschaftliche Veröffentlichungen

2.1.1 Überblick über den Forschungsstand

Das Paper „Automated university timetabling: The state of the art", das von Burke, Jackson, Kingston und Weare verfasst und 1997 veröffentlicht wurde, bietet einen guten Einstieg in das Problem der computergestützten Erstellung von Stundenplänen.
Die ursprüngliche Definition des Timetabling-Problems ermöglicht eine ersten Einordnung dessen. "A timetable is a placement of a set of meetings in time. A meeting is a combination of resources ..., some of which may be specified by the problem, and some of which must be allocated as part of the solution." [1]. In einem Stundenplan wird also festgehalten, zu welcher Zeit welches Treffen stattfindet. Ein Treffen ist eine Kombination von benötigten Ressourcen.
Bisher ist keine Lösungsmethode bekannt, die das Problem in angemessener Zeit lösen kann. An Universitäten unterscheidet man zwei Probleme bei der Planung. Auf der einen Seite die Anordnung der Vorlesungen und auf der anderen die Planung der Prüfungen. Prüfungen werden beispielsweise so geplant, dass Studenten nicht mehrere an einem Tag absolvieren müssen. Die Anordnung der Vorlesungen wird bereits festgelegt, bevor sich Studenten überhaupt in die Module einschreiben. Außerdem verlangt die Vorlesungsplanung eine andere Raumpla-

nung als die Prüfungsplanung [1].
Für die Planung ist es Grundvoraussetzung, die Termine über eine bestimmte Zeit so anzulegen, dass keine Konflikte entstehen und Nebenbedingungen berücksichtigt werden. Probleme entstehen beispielsweise dann, wenn die Planung es erfordert, dass Ressourcen an zwei Orten gleichzeitig vorhanden sein sollen. Die Nebenbedingungen unterscheiden sich an verschiedenen Institutionen. Dies ist für die Automatisierung der Erstellung von Stundenplänen eine besondere Herausforderung, da die Liste an Nebenbedingungen endlos scheint [2]. Auch Bloomfield und McSharry stellten fest, dass der Planungsprozess bei größerem und vielfältigerem Angebot steigt.
Romero beschreibt drei Gruppen, die Anforderungen an einen Stundenplan haben:

1. Verwaltungen, die Mindestanforderungen vorgeben,
2. Fachbereiche, die vorschreiben, dass der Stundenplan das Voranschreiten im Fach berücksichtigt, die Anforderungen an bestimmte Räume haben oder die die Prüfungszeiten individuell festsetzen wollen, um mehr Zeit für die Korrekturen zu haben,
3. Studenten, die ihre individuellen Vorstellungen umsetzten wollen

Häufige Nebenbedingungen sind:

1. Ressourcen (z.B. Räume für bestimmte Lehrer/Veranstaltungen),
2. Zeiten (z.B. Tage, an denen Lehrkräfte zur Verfügung stehen),
3. Zeitlicher Ablauf von Veranstaltungen,
4. Verteilung von Veranstaltungen,
5. Raumkapazitäten,
6. Kontinuität (Vorhersagbarkeit/Konstanz für Studenten).

Diese Bedingungen können in weiche und harte Nebenbedingungen gegliedert werden. Dabei ist die Verletzung harter Nebenbedingungen undurchführbar und verlangt eine entsprechende Korrektur des Plans. Es sind Konflikte ersten Ranges. Weiche Nebenbedingungen sind weniger wichtig. Hier ist es nicht immer möglich, alle Anforderungen einzuhalten. Jede Programmierung enthält eine Art Sperrfunktion, die die Schwere der Missachtung der weichen Nebenbedingungen

bewertet und entsprechend reagiert.
Um die Praktiken der Planung zu erforschen, wurde eine Umfrage unter britischen Universitäten gestartet. Dabei wurden unterschiedliche Voraussetzungen untersucht. Dazu gehörten die Anzahl der Studierenden, räumliche Gegebenheiten, der Aufbau des Campus etc [1].
Die Herangehensweise der Universitäten ist hierbei unterschiedlich. Einige benutzen den Plan vom Vorjahr, andere konstruieren jedes Jahr einen neuen Plan. Die Erstellung eines neuen Plans dauerte damals ohne Computerunterstützung bis zu 4 Wochen. Der Mangel an Automatisierung liegt heute aber vor allem bei den Planern selbst, die Stift und Tafel vorziehen, da sie so einen besseren Überblick haben und das lästige Scrollen und Abfragen am Computer entfällt.
Die Umfrage zeigte, dass nur 21% eine Computerunterstützung benutzen. Diese benötigen darüber hinaus aber trotzdem noch manuelle Arbeit oder Erfahrung aus vorherigen Planungen.
Die Studie zeigt, dass eine automatische Hilfe für die Erstellung von Plänen dringend erforderlich ist. Dabei sollten die Systeme eine große Anzahl von Nebenbedingungen berücksichtigen, kompatibel mit bestehenden Systemen sein und eine einfache Benutzeroberfläche aufweisen.
Zusammenfassend nutzen Universitäten unterschiedliche Herangehensweisen, die meist nur für einzelne Einrichtungen verwendbar sind. Das führt dazu, dass ein Vergleich von Methoden nicht möglich ist. Dieser ist jedoch wichtig auf der Suche nach der besten Methode der Planung. Je komplizierter und zahlreicher die Nebenbedingungen der Einrichtungen, desto schwieriger ist der Vergleich. Derzeit gibt es keine Maßstäbe, die einen echten Vergleich möglich machen. Der Datenaustausch zwischen Forschern ist daher kaum vorhanden. Aus diesem Grund finden nun Konferenzen zum Zweck des Forschungsaustauschs statt.
Bei der ersten internationalen Konferenz zur automatischen Erstellung von Plänen "Practice And Theory of Automated Timetabling" wurde eine große Vielfalt von Herangehensweisen an das Timetabling-Problem vorgestellt. Einige Forscher nähern sich der Lösung des Problems mit Hilfe des Genetic oder Memetic Algorithmus, andere nutzen Simulated Annealing, Tabu Search oder Constraint Logic Programming.
Schlussendlich weiß man, dass der schwierigste Teil der Stundenplanerstellung durch ein entsprechendes Programm den Planungsprozess erleichtern und verkürzen würde [1].

Das Paper „Automated university timetabling: The state of the art“ [1] bot einen umfangreichen Einblick in den Forschungsstand der Planung an Universitäten. Im Vergleich dazu wurde einige Jahre später an der University of KwaZulu-Natal von Pillay ein Überblick über den Forschungsstand zu der Erstellung von Stundenplänen an Schulen veröffentlicht [16]. Im Gegensatz zu der nun im Laufe der Jahre gut kommunizierten Forschung in Richtung der computergestützten Erstellung von Stundenplänen an Universitäten und den zugehörigen Prüfungsstundenplänen, erfolgte die Entwicklung der Erstellung von Stundenplänen für Schulen immer noch separat. Das Paper „An overview of school timetabling research“ [16] zeigt, in wie viele verschiedene Richtungen die Untersuchungen gehen. Probleme, die hierbei aufgeführt werden, sind nicht nur die voneinander abweichenden Definitionen von Mengen, Parametern und Variablen, sondern vor allem die Implementierung der Probleme, die so speziell sind, dass sie nicht auf die Allgemeinheit angewandt werden können. Schlussendlich sind zum großen Teil die unterschiedlichen Bildungssysteme in den Ländern ausschlaggebend für eine individuelle Planung.
Wie auch bei der Erstellung von Stundenplänen an Universitäten werden hier Tupel von Klassen, Lehrern und Räumen entsprechenden Timeslots zugeordnet. Außerdem werden die Nebenbedingungen ebenfalls in harte und weiche Bedingungen unterteilt. Die Methoden zur Lösung des Problems sind sehr unterschiedlich (Simulated Annealing, Evolutionary Algorithms, Tabu Search, Integer Programming, Constraint Programming, GRASP (Greedy Randomized Search Procedure), Tiling Algorithms).
In der Zukunft sollte mehr Zusammenarbeit angestrebt werden und die entwickelten Programme müssen in ihrer Bedienerfreundlichkeit überarbeitet werden. Es muss für die Nutzer der Programme leichter sein Nebenbedingungen anzupassen, da sich die Bildungssysteme rasch verändern können [16].

Aus beiden Veröffentlichungen lässt sich schlussfolgern, dass die Forschung auf dem Gebiet des Timetabling-Problems noch vor vielen Problemen steht. Die automatische Erstellung von Stundenplänen, sei es für Universitäten, Schulen oder andere Bereiche, bietet noch viele Forschungsmöglichkeiten. Die Herausforderung, Programme mit entsprechenden Laufzeiten und der gewünschten Flexibilität zu erstellen, ist Aufgabe zukünftiger Wissenschaft.

2.1.2 Herangehensweisen zur Lösung des Timetabling-Problems

Anfangs wurde sich dem Timetabling-Problem mit der Methode der Graphenfärbung und anderer Heuristiken genähert. Schon das im Jahr 1967 veröffentlichte Paper „An upper bound for the chromatic number of a graph and its application to timetabling problems“ von Welsh und Powell zeigt, dass es eine Verbindung zwischen Problemen in der Graphentheorie und dem Timetabling-Problem gibt [21].
Im Gegensatz zu der klassischen Annäherung von lediglich einer mathematischen Methode, werden heute mehrere neue und effiziente Techniken kombiniert. Unter den Techniken befinden sich Methoden wie Tabu Search, Constraint Logic Programming und genetische Algorithmen [4].

Memetic Algorithmus

Jat und Yang [8] von der University of Leicester entwickelten einen Algorithmus zur Erstellung von Stundenplänen. Diesen präsentieren sie in ihrem Paper „A memetic algorithm for the university course timetabling problem“. Der Memetic Algorithmus, der zwei lokale Suchmethoden in einen generischen Algorithmus integriert, gehört ihrer Meinung nach zu den damalig besten Algorithmen zur Lösung des Stundenplanproblems von Universitäten. Experimentelle Ergebnisse untermauern diese Aussage und zeigen, dass es sich um einen effizienten Algorithmus handelt. Durch die Kombination von Nachbarschaftsuntersuchungen ist es möglich, gute Resultate bei unterschiedlichen Arten von Nebenbedingungen in kurzer Zeit zu erzielen.
In der Zukunft sollen mehr Methoden für die Nachbarschaftssuche analysiert und angewendet werden. Die Wechselbeziehung von generischen Operationen, Heuristiken und die mit der Zeit gesammelten Erfahrungen sowie das richtige Stellen dieser Techniken in einem Algorithmus können schlussendlich zu besseren Ergebnissen führen [8].

Constraint Programming

Mit dem Constraint Programming näherten sich Valouxis und Housos [20]dem Timetabling-Problem an. Sie testeten ihre Untersuchungen an griechischen Schu-

len. In dem Paper „Constraint programming approach for school timetabling“ wird präsentiert, dass mit der Kombination aus Constraint Programming und herkömmlichen Operation-Research-Methoden mehrere große Timetabling-Probleme gelöst werden können.
Das Modell definiert eine Zielfunktion, die die tägliche Belastung der Lehrkräfte aufteilt und berücksichtigt, welche Vorlieben bezüglich der Arbeitszeiten bestehen. Auf Seiten der Schüler muss beachtet werden, dass keine Freiblöcke im Stundenplan entstehen dürfen. Die eine Hälfte des Modells, die sich mit der Datenverwaltung und den Einschränkungsoptionen auseinandersetzt, wird mit Hilfe des Constraint Programming definiert. Die andere Hälfte, die die Wahl der Suchwege beinhaltet, wird mit entsprechenden Techniken des Operation Research modelliert. Die Implementierung fand in C++ statt und wurde mit einem ILOG-Solver gelöst.
Durch Anpassung der Suchstrategie soll das entwickelte Modell leicht auf andere Timetabling-Probleme übertragen werden können. Die guten Rechenlaufzeiten sind durch die Eingrenzung des Suchraums zu begründen. Weitere Forschungen in der Zukunft, die eine Kombination des Constraint Programming und anderen Methoden des Operation Research nutzen, scheinen vielversprechend und interessant für die Lösung von Timetabling-Problemen zu sein [20].

Integer Programming

In dem Paper „A computational approach to enhancing course timetabling with integer programming“, das die Herangehensweise einer iranischen Universität veranschaulicht, wird gezeigt, dass die ganzzahlige Optimierung eine gute Möglichkeit bietet, das Timetabling-Problem zu lösen. Das Modell der ganzzahligen Programmierung ist ein sehr einfaches. Auch die Implementierung solcher Probleme kann mit Hilfe einfacher mathematischer Programmiersprachen vollbracht und im Anschluss mit einem Integer-Programming-Solver optimiert werden. Durch die Annäherung der ganzzahligen Optimierung ist es möglich, das Modell in verschiedene interessante Richtungen zu erweitern. Das entwickelte Modell stellt Einschränkungen für Regeln und akademische Voraussetzungen zur Verfügung. Die genauen Restriktionen können [13] entnommen werden.
An der Shahrood University of Technology gibt es eine separate Abteilung, die nur für die Stundenplanerstellung zuständig ist. Die ganzzahlige Programmie-

rung ist hier ein klassisches Werkzeug, das bei vielen Problemen angewendet werden kann. Die Techniken der ganzzahligen Optimierung sind sehr gut erforscht und bieten zuverlässige Lösungsmethoden. Jedoch ist der entstehende Rechenaufwand in Relation zu dem Timetabling-Problem zu setzen. Es kann passieren, dass die Laufzeiten der Algorithmen bei einer großen Anzahl an Variablen schnell ansteigen. Aus diesem Grund kann man erst sicher sein, ob der implementierte Algorithmus eine Lösung in angemessener Zeit findet, wenn man die Lösungsmethode an einem realistischen Beispiel testet.
In dem beschriebenen Modell werden die Entscheidungsvariablen als Binärvariablen definiert, die von der Wahl des Lehrenden, dem Unterrichtskurs, dem Tag und der Zeit abhängen. Zusätzlich werden die Unterrichtszeiten durch einen Parameter unterschiedlich gewichtet. Es werden Variablen eingeführt, die die Nichteinhaltung von Nebenbedingungen bestrafen sowie aufzeichnen. In der Zielfunktion werden die Variablen der weichen Nebenbedingungen und die Bestrafung von nicht bevorzugten Unterrichtszeiten minimiert.
Ziel des Modells ist es, einen konfliktfreien Stundenplan zu entwerfen. Für die Implementierung wurde die Programmiersprache AIMMS genutzt. Der IP-Solver XA (GAMS/XA linear and mixed-integer programming solver) wurde zur Lösung des Algorithmus verwendet.
Das vorgestellte Modell arbeitet eine Stufe höher als die bereits von der iranischen Universität genutzten Modelle. Hauptinhalte waren die Raumzuteilung und die Berücksichtigung der Voraussetzungen an die Studierenden. Mögliche Probleme, die durch das getestete Experiment nicht widergespiegelt wurden, sind die Laufzeit und die Einschränkung des Suchprozesses. Wenn mehr Zeit für den Suchprozess zur Verfügung steht, kann ein größerer Suchraum betrachtet werden. Das Modell wäre so auch auf größere Probleme anwendbar. Außerdem kann durch Änderungen der Größenordnungen, der Bestrafung der Nebenbedingungen der Optimierungsprozess beschleunigt werden und schneller zu einer optimalen Lösung führen [13].

Ein weiteres Beispiel für die Option der Lösung des Timetabling-Problems durch die ganzzahlige Programmierung bietet das Paper „An integer programming formulation for a case study in university timetabling“. Wie in dem vorherigen Modell stellt das Verfahren in dem Paper von Daskalaki, Birbas und Housos [4] viele Einschränkungen zur Verfügung, die es ermöglichen, Gesetze und akademische

Voraussetzungen zu berücksichtigen. Ziel ist es ebenfalls, die lineare Kostenfunktion zu minimieren. Durch die Zielfunktion ist es möglich, Befriedigungen bezüglich lehrender Perioden, Tage oder Unterrichtsräume auszudrücken. Durch eine passende Definition der Kostenkoeffizienten kann eine Einschränkung des Lösungsraums erreicht werden. Das Problem kann dadurch lenkbarer gemacht werden. Die vorhandenen IP-Solver ermöglichen es Problemen mit vielen Variablen zu lösen. Somit ist es auch für große Einrichtungen machbar einen Stundenplan zu erstellen. In dem Paper wird für eine technische Fakultät mit vielen Kursen und Lehrern für fünf Jahre geplant.
An vielen Einrichtungen ist die Verwaltung für die Erstellung eines neuen Plans zuständig. Diese passt den alten Stundenplan an aktuelle Änderungen an. Jedoch ist die Überarbeitung alter Pläne nicht immer die beste Lösung. Durch den Fortschritt bei der Hardware- und Softwaretechnologie und dem Wissensstand der Forschung, könnten die Stundenpläne durch automatische Verfahren effizienter Pläne entwickeln.
Das Timetabling-Problem ist ein sehr komplizierter und langwieriger Prozess. Jedoch ist es nun mit Hilfe der ganzzahligen Programmierung möglich, ein flexibles Modell aufzustellen, das das Problem automatisch löst. Die Flexibilität resultiert aus der Definition mehrdimensionaler Variablen [4].

In diesem Absatz wird eine Arbeit von Schimmelpfeng und Helber genauer betrachtet. Das Paper „Application of a real-world university-course timetabling model solved by integer programming“, das 2007 veröffentlicht wurde, setzt sich ebenfalls mit der Lösung des Timetabling-Problems mit Hilfe der ganzzahligen Optimierung auseinander. Das Paper veranschaulicht eine Fallstudie eines Fachbereichs der Universität Hannover.
Geplant wird für 150 verschiedene, wöchentlich stattfindende Vorlesungen, Tutorien und Seminare, die von Gruppen bestehend aus 5 bis 650 Studenten besucht werden. Unterrichtet wird von ca. 100 Lehrern. Es muss eine entsprechende Zuteilung von Unterrichtszeiten und Räumen erfolgen. In dem beschriebenen Modell werden ungefähr 100.000 Variablen und zahlreiche Typen von Nebenbedingungen definiert.
Allgemein wurde das Timetabling-Problem an Universitäten als Zuweisung von sich regelmäßig wiederholenden lehrenden Gruppen (Kurs, Studenten und Lehrer) zu passenden Räumen und Zeiteinheiten beschrieben. Es ist nicht überra-

schend, dass es trotz der zahlreich vorhandenen Literatur, keinen allgemein verwendeten Algorithmus für den Planungsprozess gibt, da die jeweiligen Ziele und Einschränkungen der Bildungseinrichtungen hoch speziell sind.
Das Modell berücksichtigt im Planungsprozess vier Gruppen: die Zeit, den Raum, den Kurs und die Lehrkräftepräferenzen. Neben der Zielfunktion wurden 20 Arten von Nebenbedingungen definiert, die wiederum in vier Kategorien unterteilt sind. Dazu gehören die grundlegenden Anweisungsbeschränkungen von Lehrkräften, die schulspezifischen Voraussetzungen, die Einschränkungen, die sich aus Sicht des Instituts ergeben, und die Einschränkungen aufgrund der Lehrkräftepräferenzen.
Aus der Fallstudie ergab sich, dass die Stundenpläne besser waren als alle bisherigen. Durch die Akzeptanz der neuen Herangehensweise wird nun nicht mehr mit Hilfe von GAMS programmiert, sondern in FlopC++, einer Klasse der Programmiersprache C++. Es ist nun möglich, das Timetabling-Problem mit einer hohen Lösungsqualität zu beheben, das vorher viele Jahre lang betrachtet worden war [18].

Zum Abschluss wird das erfolgreiche Programm *Moses*, dass von der Technischen Universität Berlin entwickelt wurde, vorgestellt. Seit 2002 forscht die Abteilung *innoCampus* in verschiedene Richtungen, um das Timetabling-Problem effizienter zu lösen. Im Frühling 2003 wurde *Moses* zum ersten Mal an der TU Berlin verwendet. Zu diesem Zeitpunkt wurden mehr als 32.000 Studenten den jeweiligen Tutorien des Mathematikfachbereichs zugeordnet. Heute kann das Programm die Studenten aus mehr als 80 großen Kursen den mehr als 1.000 Tutorien zuteilen. Wegen des großen Erfolgs ist *Moses* erweitert worden. Seit 2015 wird *Moses* auch an der Technischen Universität München mit mehr als 39.000 Studenten angewandt. Auch die RWTH Aachener Universität nutzte *Moses* zur Planung. Die TU Berlin erstellt ihre Stundenpläne mittlerweile komplett automatisch und kopiert nicht mehr die Stundenpläne aus vorherigen Semestern.
Das Optimierungsmodell hinter *Moses* ist die gemischt-ganzzahlige Programmierung. Die Lösung erfolgt mit einem kommerziellen MIP-Solver. Innerhalb von Stunden kann das Modell eine annehmbare Lösung (weniger als 20 % vom Optimum) finden. Harte Nebenbedingungen werden vermieden, da diese dem Planer keine Informationen liefern, an welchen Stellen das Programm bei der Erstellung eines optimalen Plans scheitert. Im Gegensatz zu vielen anderen Planungslösun-

gen ist man dadurch im Stande zu schätzen, wie weit der erstellte Plan vom Optimum entfernt ist.
Die Implementierung erfolgte mit Java EE. Das System *Moses* hat verschiedene Schnittstellen zu anderen Systemen, um Daten automatisch, halbautomatisch, oder manuell zu importieren und zu exportieren.
Je größer die Timetabling-Probleme sind, desto offensichtlicher ist es geworden, dass eine Standardisierung notwendig ist. Für die Zukunft wäre es wichtig, Bedingungen unabhängig voneinander zu definieren, um sie dann abhängig von den Vorgaben der jeweiligen Bildungseinrichtung zusammenstecken zu können [12].

2.2 Schlussfolgerung

Schon bei den vorgestellten Herangehensweisen an die Lösung der Timetabling-Problems ist zu erkennen, wie unterschiedlich die Anforderungen an die Stundenpläne der einzelnen Einrichtungen sind. Die voneinander abweichenden Bildungssysteme und Präferenzen der Schulen und Universitäten haben große Auswirkungen auf den Planungsprozess.
Mehrere Autoren teilen die Voraussetzungen in zwei Gruppen. Zum einen die Bedingungen, die fest sind, die dann als harte Nebenbedingungen im Modell berücksichtigt werden, zum anderen die weichen Nebenbedingungen, die in der Zielfunktion widergespiegelt werden. Oft wird auch die Strategie des Teilens des Timetabling-Problems in kleinere Teilprobleme verwendet [4].

Die Planung mit Unterstützung der ganzzahligen Programmierung zeigte vielversprechende Resultate, sowohl in der einfachen Aufstellung der Modelle als auch in den Laufzeiten der erarbeiteten Algorithmen. Aus diesem Grund wird sich dem Problem der Erstellung eines Stundenplans für eine Unteroffiziersschule der Bundeswehr über die ganzzahlige Optimierung genähert.
Bereits vorhandene Entwicklungen können hier nicht genutzt werden, da die Planung bei der Bundeswehr nicht wochen-zyklisch stattfindet. Bei der Unteroffiziersschule wird ein Semester geplant, in dem keine Wiederholungen von Kursen berücksichtigt werden müssen. Des Weiteren gibt es auch keine Einschränkungen, die den Lehrkörper betreffen. Es bestätigt sich die Aussage von vielen Wissen-

schaftlern, die sich mit dem Thema befassen, dass eine allgemeine automatische Unterstützung bei der Erstellung von Stundenplänen durch die unterschiedlichen Nebenbedingungen nicht möglich ist.

3 Mathematischer Hintergrund

Die Methode der linearen Programmierung, die hier zur Lösung des Timetabling-Problems genutzt wird, gehört zu den gängigsten Lösungsverfahren des Operation Research (mathematische Optimalplanung). Das Operation Research ist ein Teilgebiet der Wissenschaft, das sich mit der Nachbildung realer Entscheidungsprobleme durch ein mathematisches Optimierungsmodell und der Lösung dessen befasst [6]. Bei der Planung von militärischen Operationen im Laufe des Zweiten Weltkriegs wurde durch die USA und Großbritannien bereits eine große Anzahl an Lösungsmethoden entwickelt. Diese Verfahren wurden mit der Zeit weiter ausgebaut und werden heute größtenteils für wirtschaftliche Probleme verwendet. In Deutschland wird das Operation Research daher auch oft als Unternehmensforschung bezeichnet. Alle Probleme haben die Gemeinsamkeit, dass eine optimale Entscheidung zu treffen ist. Das heißt, dass eine strukturelle Ähnlichkeit vorliegt, die es ermöglicht, universelle Lösungsmöglichkeiten zu entwickeln [10].
Grundlegende Planungsprozesse, so auch die des Operation Research, sind wie folgt aufgebaut:

1. Problem erkennen und analysieren,
2. Ziele und Handlungsmöglichkeiten aufzeigen,
3. Aufstellen des mathematischen Modells,
4. Datenbeschaffung und Datenaufbereitung,
5. Ermittlung der Lösung,
6. Bewertung der Lösung.

Es wird also ein reales Problem durch ein mathematisches Modell abgebildet, welches dann mit Hilfe eines Algorithmus gelöst wird [5]. Der letzte Schritt, auch Validierung der Ergebnisse genannt, ist sehr wichtig und sollte nicht vernachlässigt werden. Hier wird unter anderem analysiert, wie aussagekräftig das Ergebnis ist

[10].
Die Lösungsmethoden aller Probleme des Operation Research gehen nach dem beschriebenen Muster vor. Unterschiede ergeben sich aber in den verschiedenen Verfahren, die genutzt werden können und in Rechenvorschriften festgehalten werden [10]. Bei wirtschaftlichen Problemen, die nicht mit Hilfe von Algorithmen gelöst werden können, gibt es die Möglichkeit der Simulation oder der Anwendung von Heuristiken [6].
Abbildung 3.1 zeigt einen Überblick über die Teilgebiete des Operation Research [5].

Abbildung 3.1: Teilgebiete des Operation Research.

3.1 Lineare Optimierung

In der Praxis findet das Teilgebiet der linearen Optimierung etliche Anwendungen. Viele praktische Aufgabenstellungen können mit einem linearen Ansatz modelliert werden [14]. Die lineare Programmierung zeichnet sich durch eine einfache Modellbildung aus. Durch effiziente Lösungsverfahren lassen sich auch komplexe Probleme mit einer Vielzahl von Variablen und Nebenbedingungen lösen [15]. Mit Hilfe der linearen Programmierung können zum Beispiel Transport-, Produktions-, Mischungs- oder Zuordnungsprobleme gelöst werden [9]. Die Problemstellungen beinhalten entweder eine Nutzen- bzw. Gewinnmaximierung oder Kostenminimierung [14].
Die Optimierung der Verfahren lässt sich durch ein System mit einer linearen Zielfunktion und linearen Nebenbedingungen formulieren. Die Nebenbedingungen werden in Form von Gleichungen und Ungleichungen dargestellt. Die Zielfunktion wird unter den gegebenen Nebenbedingungen maximiert oder minimiert. Das Modell der linearen Optimierung ist damit universell einsetzbar und wird auch bei der ganzzahligen, nichtlinearen oder dynamischen Optimierung als Grundlage verwendet [15].

Die **Zielfunktion** wird im Allgemeinen als eine lineare Funktion $f(x)$ bezeichnet und besitzt folgende Gestalt

$$f(x) = f(x_1, ..., x_n) = c_1x_1 + c_2x_2 + ... + c_nx_n.$$

Die Variablen $x_1, ..., x_n$ werden dabei **Entscheidungsvariablen** genannt. Die Konstanten $c_1, ..., c_n$ stellen in der Regel Kosten-, Ertrags- oder Gewinnfaktoren dar [10]. Auf die Zielfunktion wirken bestimmte **Nebenbedingungen** (Restriktionen)

$$a_{i1}x_1 + a_{i2}x_2 + ... + a_{in}x_n \geq b_i.$$

Diese können als Mindestanforderungen formuliert werden. Sie können aber auch in der $\leq$- oder $=$- Beziehung vorliegen. Für eine einheitliche Darstellung werden die Nebenbedingungen oft in Mindestanforderungen umgeformt [15].
Für die Entscheidungsvariablen gilt stets die **Nichtnegativitätsbedingung** [5]

$$x_i \geq 0.$$

Das lineare Optimierungsproblem mit m Nebenbedingungen liegt also in folgender Form vor [10]:

$$\begin{aligned} \max(\min) \quad & c_1x_1 + c_2x_2 + ... + c_nx_n \\ \text{unter den Nebenbedingungen} \quad & a_{i1}x_1 + a_{i2}x_2 + ... + a_{in}x_n \geq b_i, \qquad i = 1, ..., m, \\ & x_i \geq 0, \qquad i = 1, ..., n. \end{aligned}$$

Als **zulässige Lösung** bezeichnet man ein $x \in \mathbb{R}^n$, welches alle Nebenbedingungen der Optimierungsaufgabe erfüllt. Die Menge aller zulässigen Punkte nennt man **zulässige Menge** [14]

$$\mathbb{M} = \{x \in \mathbb{R}^n | \, a_i x \geq b_i, \, i = 1, ..., m, \, x \geq 0\}.$$

Ein Punkt x^* der Menge $\mathbb{M}$, der das Maximierungsproblem (Minimierungsproblem) am besten löst, wird als **optimale Lösung** bezeichnet. Dieser Punkt erfüllt also die folgende Bedingung

$$f(x) \leq f(x^*) \quad (f(x) \geq f(x^*)) \quad \forall x \in \mathbb{M}.$$

Der Wert $f(x^*)$ heißt **optimaler Wert** [15].

3.2 Ganzzahlige lineare Optimierung

Entscheidungsprobleme, die mit Hilfe der ganzzahligen linearen Optimierung gelöst werden, sind in der heutigen Wirtschaft ebenso vertreten wie die der linearen Optimierung. Hierbei sind die Entscheidungsvariablen auf den ganzzahligen, natürlichen oder binären Wertebereich beschränkt [15]. Die Entwicklung der Lösungsmethoden der ganzzahligen Programmierung startete kurz nach der linearen Programmierung. Bereits 1958 entwickelte Gomory ein erstes endliches Verfahren zur Lösung von ganzzahligen Problemen [19].
Die Ermittlung einer optimalen ganzzahligen Lösung ist erheblich aufwendiger als die Suche nach einer reellen Lösung [14]. Umso mehr Variablen definiert werden, desto größer ist der Lösungsaufwand. Das heißt, dass mehr Speicher benötigt wird und die Laufzeiten der Algorithmen exponentiell ansteigen [11]. Aus diesem Grund wird bei Problemen mit einer großen Anzahl von Entscheidungs-

variablen oft die lineare Optimierung zur Ermittlung einer Lösung genutzt und die Ergebnisse werden dann entsprechend gerundet [14]. Aber gerade bei kleineren Problemen kann eine solche Herangehensweise zu sehr ungenauen und vor allem nicht zwingend zu optimalen Lösungen führen [6].

Durch die Verwendung von ganzzahligen Variablen ergeben sich neue Modellierungsmöglichkeiten. Es lassen sich logische Verknüpfungen mit Hilfe von binären Variablen ausdrücken:

$$x_i = \begin{cases} 1, & \text{wenn } A_i \text{ wahr ist} \\ 0, & \text{sonst} \end{cases} \quad i = 1, ...n$$

Das heißt, dass die Variable x_i auf 1 gesetzt wird, wenn die Aussage A_i wahr ist, anderenfalls auf 0.
Des Weiteren können Mengenbeziehungen formuliert werden:

$$y \in \{0, 1\},$$
$$x \leq M \cdot y.$$

Die Variable y ist hierbei eine Entscheidungsvariable, die zum Beispiel aussagt, ob produziert wird oder nicht. Die Variable x gibt an, wie viele Einheiten hergestellt werden sollen. Wenn $y = 0$, ist auch $x = 0$. Wenn aber $y = 1$, ist $x \in \{0, ..., M\}$. Das bedeutet, dass maximal M Stück produziert werden können aber x genauso auch 0 sein kann.
Die dritte Möglichkeit ist, "Entweder-Oder"-Zusammenhänge bzw. alternative Nebenbedingungen zu modellieren. Dafür wird wieder eine Binärvariable y eingeführt, die eine Vorentscheidung trifft, womit andere Nebenbedingungen eingeschränkt werden. Nähere Erläuterungen findet man in [15].

3.2.1 Komplexitätstheorie

Der Ressourcenbedarf für Verfahren zur Lösung von ganzzahligen Problemen ist höher als der bei Verfahren der linearen Programmierung. Es ist in der Praxis also nicht nur die Ermittlung einer optimalen Lösung von Interesse, sondern auch der benötigte Aufwand. Diesen Aufwand misst man an der Zeit, die das Verfahren zur Lösung des Problems benötigt und dem verwendeten Speicherplatz.

Die formulierten Probleme sind Zusammensetzungen von Teilproblemen. Ein Algorithmus ist das Verfahren zur Lösung eines Problems, das eine Folge von Rechenschritten, die für jede Probleminstanz die gesuchte Antwort liefert, vereint. Die Eigenschaften, die einen Algorithmus auszeichnen, sind dabei Endlichkeit, Beschreib- und Durchführbarkeit sowie die Eindeutigkeit in der Abfolge der Rechenoperationen.
Die Rechenschritte, die im Algorithmus ausgeführt werden, bestimmen den Rechenaufwand dessen. Die Grundrechenarten, Vergleiche und Sprünge sowie Speicheroperationen zählen dabei zu Rechenschritten eines Algorithmus [6].
Ein Algorithmus besitzt einen Rechenaufwand. Dies nennt man auch Komplexität eines Algorithmus [5]. Ist $g(n)$ ein Polynom, ist der Algorithmus polynomial oder effektiv [6]. Ist das nicht der Fall, das heißt, dass kein Polynom gefunden werden kann, spricht man von exponentiellem Rechenaufwand [5].
Alle Probleme, die mit einem Verfahren, das in polynomialer Zeit eine optimale Lösung findet, modelliert werden können, gehören zu der Klasse P. Die Probleme, deren Lösungsalgorithmen keine optimale Lösung in polynomialer Zeit ermitteln, teilt man der Klasse NP zu [6].
Algorithmen, die eine optimale Lösung des Problems finden, nennt man exakte Lösungsverfahren. Sie garantieren, die optimale Lösung in endlich vielen Schritten zu ermitteln [5].

3.2.2 Verfahren zur Lösung von ganzzahligen Problemen

Zu den Lösungsverfahren der ganzzahligen Optimierung gehören unter anderem das Lösen durch Runden, das Schnittebenen-Verfahren von Gomory und das Branch-and-Bound-Verfahren. Allgemein werden ganzzahlige Probleme nach dem Algorithmus, der in der Abbildung 3.2 dargestellt ist, gelöst.

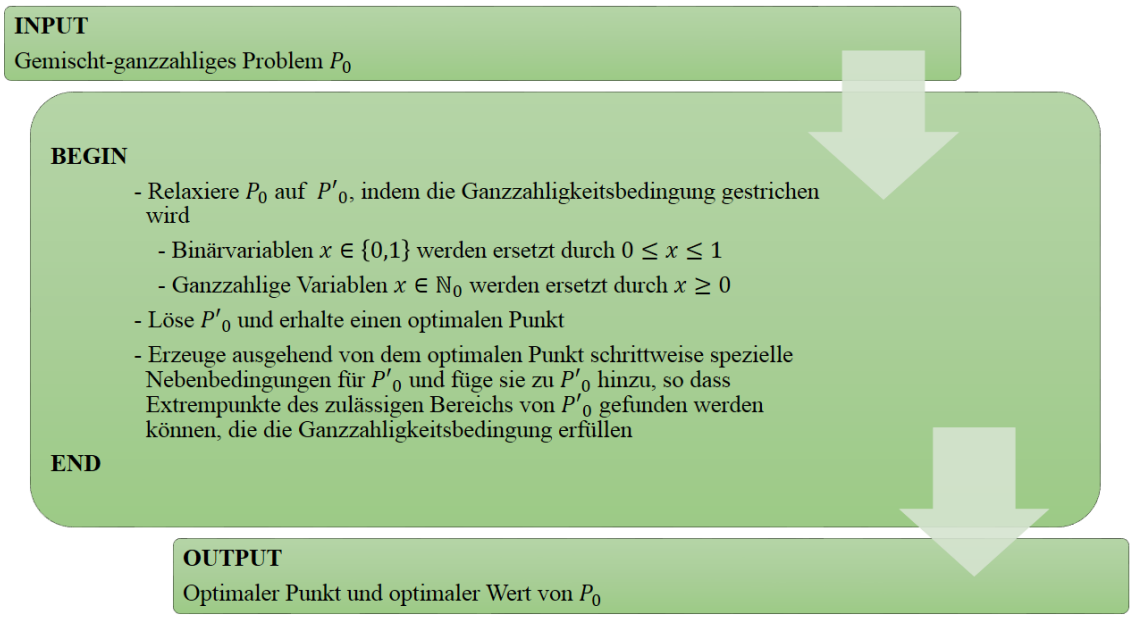


Abbildung 3.2: Generischer Algorithmus zur Lösung eines gemischt-ganzzahligen Problems [6].

Grundlage der Lösung des Stundenplanproblems bildet das Branch-and-Bound-Verfahren. Weitere Ausführungen zu anderen Verfahren zur Lösung von ganzzahligen Problemen findet man zum Beispiel in den Büchern [15], [14] und [5].
Das Branch-and-Bound-Prinzip ist eine spezielle Form des Suchverfahrens. Grundlagen des Verfahrens stammen aus den Forschungen von Nickel, Stein und Waldmann [15]. Bei der Untersuchung des Lösungsraums wird lediglich ein Bruchteil der zulässigen Menge beleuchtet. Das heißt, dass nicht alle Lösungen des Problems explizit berechnet werden [14]. Das Ausgangsproblem wird in immer kleinere Teilprobleme zerlegt (branch=verzweigen). Diese Teilprobleme werden dann durch obere und untere Schranken des Zielfunktionswerts ausgelotet und abgegrenzt (bound=abgrenzen) [5]. Das Verfahren führt durch die Eingrenzung des zu untersuchenden Raums zu einer Lösung des Problems. Zur Verdeutlichung des Arbeitens des Branch-and-Bound-Verfahrens wird mit gerichteten Bäumen gearbeitet (siehe Abbildung 3.3) [14].

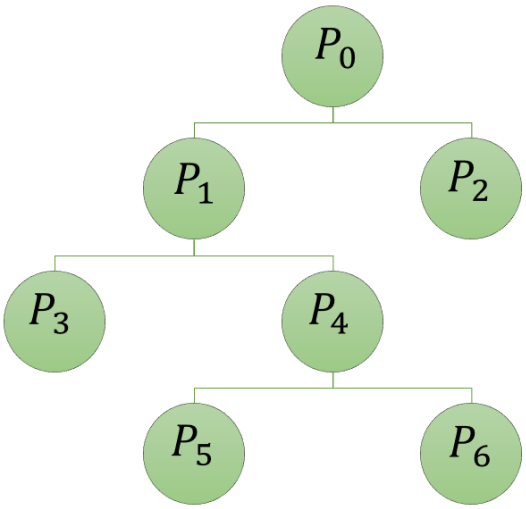


Abbildung 3.3: Lösungsbaum im Branch-and-Bound-Verfahren [10].

Die Untersuchung des Lösungsraums besitzt eine Verzweigungsstruktur, die sich bei binären Entscheidungsvariablen automatisch ergibt, da durch Ja-Nein-Entscheidungen bzw. Wahr-Falsch-Aussagen das Problem in Unterprobleme geteilt wird [10]. Schlussfolgernd ist das Verfahren zur Lösung von Optimierungsproblemen mit binären Entscheidungsvariablen besonders geeignet [14].
In Abbildung 3.4 wird schrittweise das Vorgehen des Branch-and-Bound-Verfahrens beschrieben.

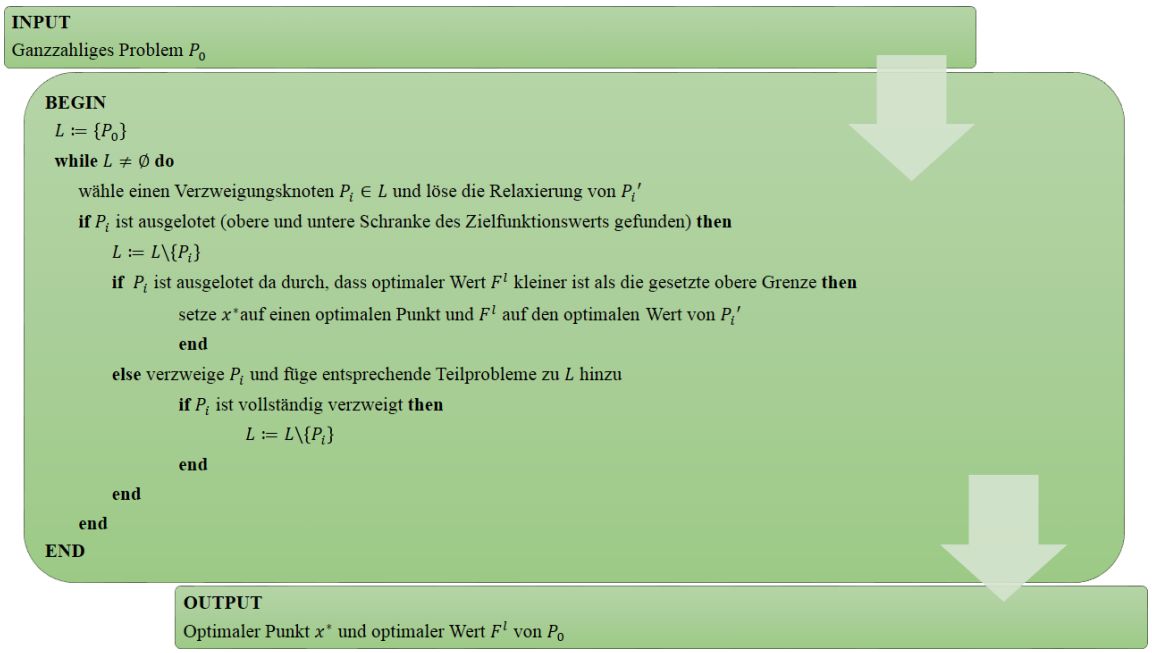


Abbildung 3.4: Allgemeines Branch-and-Bound-Verfahren [15].

3.2.3 Optimierung von Stundenplänen

Bei der Erstellung eines Stundenplans handelt es sich um ein Zuordnungsproblem. Zuordnungen erfolgen mit Hilfe von binären Entscheidungsvariablen [5]. Die Lösung des Problems besitzt also ein Verzweigungsmuster und gehört somit zu den Optimierungsproblemen, die mit dem Branch-and-Bound-Verfahren gelöst werden können [10].
Bei der Erstellung eines Stundenplans werden den Veranstaltungen der aktive und passive Handlungsträger (Lehrer und Schüler) sowie der Raum/der Ort und die Zeit zugeordnet. Zur Verdeutlichung der fünf Dimensionen betrachte man die Abbildung 3.5.

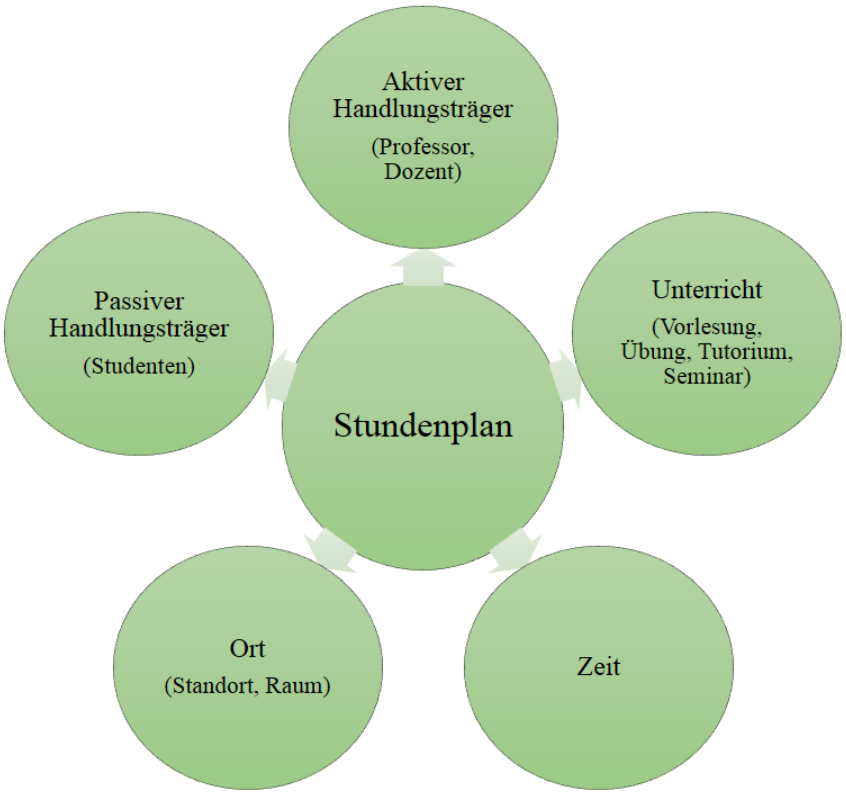


Abbildung 3.5: Dimensionen der Stundenplanerstellung.

Ziel ist es, einen zulässigen Stundenplan zu erstellen, der die Vorschriften und Ressourcen der jeweiligen Dimension berücksichtigt. Die Einschränkungen der Dimensionen können hierbei in harte und weiche Nebenbedingungen getrennt werden. Wenn der erstellte Stundenplan eine harte Nebenbedingung verletzt, ist der Plan unzulässig, wird hingegen eine weiche Nebenbedingung missachtet, führt das zur Minderung der Lösungsqualität des Plans. Aus dieser Unterteilung lässt sich leicht schlussfolgern, dass es sich nur um eine Optimierungsaufgabe handelt, wenn weiche Nebenbedingungen modelliert werden. Gibt es bei der Formulierung eines Stundenplanproblems lediglich harte Nebenbedingungen, spricht man von einem Zulässigkeitsproblem [17].

4 Datenbeschaffung

In der Unteroffiziersschule der Bundeswehr werden sechs verschiedene Lehrgänge für rund 1.200 Teilnehmer geplant. Ein Lehrgang kann hierbei in einer oder mehreren Klassen gleichzeitig stattfinden. Einfluss auf den Stundenplan haben gewisse Bedingungen. Ausschlaggebend sind die Inhalte einer Lektion oder eines Themas sowie die benötigten Ressourcen, wie zum Beispiel Unterrichtsräume und Lehrpersonal. Außerdem muss der zeitliche Ablauf beachtet werden. Die Ausbildung muss vor der Prüfung stattfinden. Weiterhin spielt auch die zeitliche Limitierung eine Rolle. Es muss die Dauer eines Unterrichtstags oder Blocks berücksichtigt werden. Zusätzlich gibt es auch feste Vorgaben der Bildungseinrichtung, beispielsweise thematische Schwerpunkte, die eingehalten werden müssen. Letztendlich muss auch sichergestellt werden, dass die externe Rahmenordnung befolgt wird. Hierunter versteht man zum Beispiel die Einhaltung des Arbeitszeitgesetzes oder die von gesetzlichen Feiertagen. Bei den Bedingungen kann es sich um Muss-Bedingungen, deren Berücksichtigung zwingend erforderlich ist, oder Kann-Bedingungen handeln.

4.1 Fachbegriffe

Das **Training** ist gleichbedeutend mit der Klasse in einer Schule. Es wird der konkrete Durchführungszeitraum festgelegt. Die Planung muss die tatsächlichen Gegebenheiten und die benötigten Ressourcen berücksichtigen.
Bei der Ausbildung werden verschiedene **Trainingstypen** unterschieden, ähnlich wie Klassenstufen. Der Begriff des Trainingstyps ist gleichbedeutend mit dem eines Lehrgangs. In einem Trainingstyp müssen die Auszubildenden in einer festgelegten Zeit bestimmte Qualifikationen erlangen. Die Eigenschaften eines Trainingstyps gelten für alle Trainings.

Die Unterrichtsstunden werden mit **Unterrichtseinheiten** bezeichnet, der jeweilige Unterricht mit **Unterrichtseinheitstyp**. Die Unterrichtseinheit ist die Umsetzung des Unterrichtseinheitstyps. Durch sie wird festgelegt, wie viel Zeit und welche Ressourcen die Ausbildungsstunde benötigt. Somit ist die Unterrichtseinheit eine maßgebliche Größe für die Erstellung eines Stundenplans. Der Unterrichtseinheitstyp beschreibt die Unterrichtseinheiten auf Typenebene. Hierbei unterscheidet man die Typen Unterricht, Praktischer Dienst, Prüfung und Fernausbildung. Der Unterrichtseinheitstyp ist vergleichbar mit dem Lehrplan eines bestimmten Unterrichts. Es werden alle nötigen Informationen festgehalten. Dazu zählt beispielsweise die Definition des groben Lernziels sowie eine Gliederung von Zieletappen, das Festhalten von Ausbildungsformen- und verfahren, der Ressourcenbedarf oder Lernzielbereiche. Allgemein sind die Unterrichtseinheitstypen eine Spezifizierung der Ziele eines Wissensgebietes.
Alle Arbeits- und Hilfsmittel, die zur Durchführung einer Unterrichtseinheit erforderlich sind, werden **Ressource** genannt. Sie wird einem oder mehreren **Ressourcentypen** zugeordnet und mit einem Trainingsort verknüpft. Die Ressourcen werden gegliedert in Räume, Referenten, Materialien und sonstige Ressourcen. Innerhalb der Planung erfolgt eine Zuordnung der Ressourcen an die Unterrichtseinheiten. Der Ressourcentyp ist eine Zusammenfassung der einzelnen Ressourcen.

4.2 Zielsetzung und Bedingungen

Es gibt eine große Anzahl an Rahmenbedingungen, die bei der Planung beachtet werden müssen. Ziel ist es, eine Lösung zu finden, die alle zwingenden Voraussetzungen erfüllt.

4.2.1 Unterrichtseinheiten

Unterrichtseinheiten finden an einem konkreten Veranstaltungsort zu einem festgelegten Zeitpunkt mit fest zugeordneten Ressourcen satt. Eine Unterrichtseinheit dauert in der Regel 45 Minuten. Der jeweilige Unterrichtseinheitstyp legt genau

fest, wie viele Einheiten unterrichtet werden müssen. Eine vollständige Ausplanung ist zwingend erforderlich.
Die Unterrichtseinheiten eines Typs müssen innerhalb eines festgelegten Zeitintervalls (Planungshorizont) geplant werden. Es darf keine Unterrichtseinheit außerhalb des Planungshorizonts liegen. Des Weiteren muss der Abstand zwischen den Einheiten beachtet werden. Hierbei kann der Unterricht lückenlos geplant werden oder es werden Pausen zwischen den Einheiten vorausgesetzt. Leerlaufzeiten zwischen den Unterrichtseinheitstypen sollen vermieden werden. Außerdem ist die vordefinierte Reihenfolge der Unterrichtseinheiten zu berücksichtigen.
Weiterhin gibt es Unterrichtseinheiten, die eine Alternative besitzen. Eine der beiden Einheiten muss geplant werden. Alle Unterrichtseinheiten eines Typs müssen am gleichen Veranstaltungsort stattfinden.

4.2.2 Ressourcen

Einige Ressourcen stehen nur in definierten Zeitperioden zu Verfügung und können vom Veranstaltungsort abhängen. Eine Ressource muss den Anforderungen entsprechen. Zum Beispiel muss die entsprechende Fachkraft eine Ausbildung für den Unterricht absolviert haben. Ferner darf eine Ressource nicht mehr als der zulässigen Anzahl von Unterrichtseinheiten zugeordnet sein. Ein Referent kann nicht mehrere Einheiten gleichzeitig unterrichten. Das maximale Arbeitszeitfenster von personellen Ressourcen muss berücksichtigt werden.
Abgesehen davon gibt es auch Ressourcen, die voneinander abhängen. Das kann zum Beispiel bei der Informationstechnik und dem zugehörigem Lehrsaal der Fall sein.
Außerdem können einige Ressourcen nur über eine bestimmte Zeit aktiviert werden. Bei der Einplanung von beweglichen Ressourcen muss die Reisezeit berücksichtigt werden. Es kann Beschränkungen bezüglich des minimalen oder maximalen Abstands vom Einsatz von Ressourcen geben. Es gibt Ressourcen, die konsequent für alle Unterrichtseinheiten eines Typs eingesetzt werden müssen und Ressourcen, die nicht mehrfach für den gleichen Typ eingesetzt werden dürfen.

Bei der Vorstellung der Lehrgänge werden die allgemein aufgezählten Bedingungen explizit formuliert und zugeordnet.

4.3 Lehrgänge

Die Ausbildung der Einrichtung der Bundeswehr unterteilt sich in sechs Lehrgänge. Hiervon bilden Lehrgang 0001 und Lehrgang 0004 den Schwerpunkt. Diese Schulungen müssen von jedem Auszubildenden absolviert werden. Die vier anderen Lehrgänge können stattfinden, müssen aber nicht. Die Lehrgänge 0002, 0003 und 0006 werden nur geplant, wenn die entsprechenden Ressourcen zur Verfügung stehen. Die Entscheidung des Durchführens eines Lehrgangs trifft der Schulleiter. Der Umfang der Lehrgänge unterscheidet sich nicht nur in der Dauer sondern auch in der Teilnehmerzahl und der Anzahl der parallel stattfindenden Trainings.

4.3.1 Lehrgang 0001

Der Lehrgang 0001 umfasst einen Planungshorizont bestehend aus 3 Kalendermonaten. Die Inhalte des Planungshorizonts werden in Unterrichtseinheiten unterteilt. Der Lehrgang 0001 ist sehr zeit- und ressourcenintensiv, weshalb eine manuelle Planung sehr langwierig ist. Es müssen 52 Unterrichtstage mit 427 Unterrichtseinheiten verplant werden. Außerdem sollen gleichzeitig 26 Trainings mit je 25 Teilnehmern diesen Trainingstyp absolvieren.
Im Allgemeinen kann jeder Unterricht an jedem Tag in jedem Training gehalten werden. Die Tabelle 4.1 bietet einen Überblick über alle im Lehrgang 0001 enthaltenen Unterrichte mit der jeweiligen Anzahl an Unterrichtseinheiten.

Tabelle 4.1: Unterrichtsblöcke des Lehrgang 0001

Unterricht	Bezeichnung	Einheiten	Inhalt	Alternative
01	U	10 UE	Unterricht	-
02	U	11 UE	Unterricht	-
03	U	11 UE	Unterricht	-
04	Ve	5 UE	Unterricht	-
05	A	10 UE	Praktische Ausbildung	-
06	PA	10 UE	Praktische Ausbildung	-
07	PA	10 UE	Praktische Ausbildung	-
08	PP	11 UE	Praktische Prüfung	-
09	PP	11 UE	Praktische Prüfung	-
10	PP	11 UE	Praktische Prüfung	-
11	PT1	5 UE	Theoretische Prüfung	-
12	B	9 UE	Praktische Ausbildung	BA
13	B	9 UE	Praktische Ausbildung	BA
14	B	9 UE	Praktische Ausbildung	BA
15	B	9 UE	Praktische Ausbildung	BA
16	BA	9 UE	Praktische Ausbildung	B
17	BA	9 UE	Praktische Ausbildung	B
18	BA	9 UE	Praktische Ausbildung	B
19	BA	9 UE	Praktische Ausbildung	B
20	PT2	3 UE	Theoretische Prüfung	-
21	VV	9 UE	Praktische Ausbildung	VVA
22	VVA	9 UE	Praktische Ausbildung	VV
23	S1	9 UE	Praktische Ausbildung	SA1
24	S1	9 UE	Praktische Ausbildung	SA1
25	S2	9 UE	Praktische Ausbildung	SA2
26	SA1	9 UE	Praktische Ausbildung	S1
27	SA1	9 UE	Praktische Ausbildung	S1
28	SA2	9 UE	Praktische Ausbildung	S2
29	S3	9 UE	Praktische Ausbildung	SA3
30	SA3	9 UE	Praktische Ausbildung	S3
31	T	9 UE	Praktische Ausbildung	VA
32	TA	9 UE	Praktische Ausbildung	V
33	H1	2 UE	Unterricht	-
34	H2	8 UE	Unterricht	-
35	PU	9 UE	Praktische Prüfung	-
36	PU	9 UE	Praktische Prüfung	-
37	PU	9 UE	Praktische Prüfung	-
38	G2	4 UE	Unterricht	-
39	G3	4 UE	Unterricht	-
40	G1	6 UE	Unterricht	-
41	PG	2 UE	Theoretische Prüfung	-
42	DD	10 UE	Praktische Ausbildung	-

Bei der Erstellung des Stundenplans sind einige Nebenbedingungen zu beachten. Als Erstes unterteilt sich die Gruppe der Trainings in zwei Hälften. Die ersten 13 Trainings beginnen ihre Ausbildung ab dem 2. Tag des Planungshorizonts und enden spätestens am 48. Tag. Die zweite Hälfte kann ab dem 4. Tag starten und beendet am 52. Tag des Planungszeitraums ihre Ausbildung.
Zusätzlich gilt für alle Unterrichte, dass nur ein Unterrichtseinheitstyp an einem Tag stattfinden darf unabhängig von der Anzahl der Unterrichtseinheiten, die dieser umfasst. Außerdem darf ein Unterricht nicht gesplittet werden. Es ist also nicht möglich, 5 Unterrichtseinheiten von einem Unterricht an dem einen Tag und die restlichen Unterrichtseinheiten an einem anderen Tag zu planen. Des Weiteren gibt es Einschränkungen der Bundeswehr bezüglich der Reihenfolge und der zeitlichen Beschränkung einiger Unterrichte sowie der Bereitstellung der benötigten Ressourcen. Die folgenden Absätze stellen die Bedingungen an die jeweiligen Unterrichtsblöcke vor. Hierbei gelten die Formulierungen müssen und dürfen als feste Vorgabe. Die Formulierungen sollen und können stellen hingegen keine feste Einschränkung dar.

Unterricht 01-03 (U)

- Die Unterrichte 01-03 müssen vor den Unterrichten 04-11 stattfinden.
- Die chronologische Reihenfolge der Unterrichte 01-03 ist frei wählbar.
- Jeder der Unterrichte 01-03 sollte entweder an einem ganzen Unterrichtstag (Montag - Donnerstag) oder an zwei halben Unterrichtstagen liegen. Falls der Unterricht auf zwei Tage gesplittet wird, darf an diesen Tagen kein weiterer Unterricht stattfinden.
- Der früheste Unterricht von 01-03 aller Trainings muss vor dem 4. Tag des Planungshorizonts durchgeführt werden.
- Der früheste Unterricht von 01-03 aller Trainings darf nicht vor dem 2. Tag des Planungshorizonts stattfinden.

Unterricht 04 (Ve)

- Der Unterricht 04 muss nach den Unterrichten 01-03 und 05 gehalten werden.
- Der Unterricht 04 muss spätestens nach einem Unterrichtstag von 08-10 stattfinden.
- Der Unterricht 04 muss vor dem Unterricht 11 liegen.

Unterricht 01-04

- Zeitgleich dürfen maximal 3 Unterrichte von 01-03 in allen Trainings geplant werden.

Unterricht 05 (A)

- Der Unterricht 05 muss nach den Unterrichten 01-03 durchgeführt werden.
- Der Unterricht 05 muss vor den Unterrichten 06-11 geplant werden.
- Zeitgleich dürfen maximal 2 Unterrichte von 05 in allen Trainings gehalten werden.

Unterricht 06-07 (PA)

- Die Unterrichte 06-07 müssen nach den Unterrichten 01-03 und 05 stattfinden.
- Die Unterrichte 06-07 müssen vor den Unterrichten 08-11 in den Stundenplan aufgenommen werden.

Unterricht 08-10 (PP)

- Die Unterrichte 08-10 müssen nach den Unterrichten 01-03 und 05-07 gehalten werden.
- Einer der Unterrichte von 08-10 darf vor dem Unterricht 04 stattfinden.

- Die Unterrichte 08-10 müssen vor dem Unterricht 11 liegen.
- Zeitgleich dürfen maximal 3 Unterrichte von 08-10 in allen Trainings geplant werden.

Unterricht 05-10

- Die Unterrichte 05-10 müssen an nicht mehr als 7 aufeinanderfolgenden ganzen Unterrichtstagen (Montag - Donnerstag) geplant werden.
- Es dürfen zeitgleich maximal 5 Unterrichte von 05-10 in allen Trainings stattfinden.

Unterricht 11 (PT1)

- Der Unterricht 11 muss nach den Unterrichten 01-10 im Stundenplan berücksichtigt werden.

Unterricht 12-15 (B)

- Jeder Unterricht 12-15 hat einen Alternativunterricht 16-19.
- Wenn einer der Unterrichte 12-15 in den Stundenplan aufgenommen wird, müssen auch die restlichen Unterrichte berücksichtigt werden. Ihre Alternativen (16-19) kommen dann nicht im Stundenplan vor.
- Die Unterrichte 12-15 müssen an maximal 5 zusammenhängenden Unterrichtstagen geplant werden.
- Die Unterrichte 12-15 müssen vor den Unterrichten 20 und 35-37 stattfinden.

Unterricht 16-19 (BA)

- Jeder Unterricht 16-19 hat einen Alternativunterricht 12-15.
- Wenn einer der Unterrichte 16-19 in den Stundenplan aufgenommen wird, müssen auch die restlichen Unterrichte berücksichtigt werden. Ihre Alternativen (12-15) kommen dann nicht im Stundenplan vor.

- Die Unterrichte 16-19 müssen an maximal 5 zusammenhängenden Unterrichtstagen geplant werden.
- Die Unterrichte 16-19 müssen vor den Unterrichten 20 und 35-37 stattfinden.

Unterricht 20 (PT2)

- Der Unterricht 20 muss nach den Unterrichten 12-15 bzw. 16-19 im Stundenplan berücksichtigt werden.
- Der Unterricht 20 muss spätestens nach einem Unterricht von 23-25 und 29 bzw. 26-28 und 30 stattfinden.
- Der Unterricht 20 sollte an einem Freitag gehalten werden.
- Der Unterricht 20 muss vor den Unterrichten 35-37 liegen.

Unterricht 21 (VV)

- Der Unterricht 21 hat einen alternativen Unterricht 22.
- Der Unterricht 21 muss vor den Unterrichten 23-25, 29 bzw. 26-28, 30 und 35-37 liegen.

Unterricht 22 (VVA)

- Der Unterricht 22 hat einen alternativen Unterricht 21.
- Der Unterricht 22 muss vor den Unterrichten 23-25, 29 bzw. 26-28, 30 und 35-37 liegen.

Unterricht 23-25 und 29 (S1, S2, S3)

- Jeder Unterricht 23-25 und 29 hat einen Alternativunterricht 26-28 und 30.
- Wenn einer der Unterrichte 23-25 und 29 in den Stundenplan aufgenommen wird, müssen auch die restlichen Unterrichte berücksichtigt werden. Ihre Alternativen 26-28 und 30 kommen dann nicht im Stundenplan vor.

- Einer der Unterrichte 23-25 und 29 darf vor dem Unterricht 20 stattfinden.
- Die Unterrichte 23-25 und 29 müssen nach dem Unterricht 21 bzw. 22 gehalten werden.
- Die Unterrichte 23-25 und 29 müssen vor den Unterrichten 35-37 liegen.
- Die Unterrichte 23-25 und 29 müssen an maximal 6 aufeinanderfolgenden Unterrichtstagen geplant werden.

Unterricht 26-28 und 30 (SA1, SA2, SA3)

- Jeder Unterricht 26-28 und 30 hat einen Alternativunterricht 23-25 und 29.
- Wenn einer der Unterrichte 26-28 und 30 in den Stundenplan aufgenommen wird, müssen auch die restlichen Unterrichte berücksichtigt werden. Ihre Alternativen 23-25 und 29 kommen dann nicht im Stundenplan vor.
- Einer der Unterrichte 26-28 und 30 darf vor dem Unterricht 20 stattfinden.
- Die Unterrichte 26-28 und 30 müssen nach dem Unterricht 21 bzw. 22 gehalten werden.
- Die Unterrichte 26-28 und 30 müssen vor den Unterrichten 35-37 liegen.

Unterricht 31 (T)

- Der Unterricht 31 hat einen Alternativunterricht 32.
- Wenn der Unterricht 31 in den Stundenplan aufgenommen wird, darf 32 nicht im Stundenplan vorkommen.
- Der Unterricht 31 muss vor den Unterrichten 35-37 geplant werden.

Unterricht 32 (TA)

- Der Unterricht 32 hat einen Alternativunterricht 31.
- Wenn der Unterricht 32 in den Stundenplan aufgenommen wird, darf 31 nicht im Stundenplan vorkommen.

- Der Unterricht 32 muss vor den Unterrichten 35-37 geplant werden.

Unterricht 33 (H1)

- Der Unterricht 33 muss vor dem Unterricht 34 stattfinden.
- Der Unterricht 33 sollte an einem Freitag gehalten werden.
- Der Unterricht 33 muss nach einem Unterrichtstag von 12-32 geplant werden.

Unterricht 34 (H2)

- Der Unterricht 34 muss nach dem Unterricht 33 stattfinden.
- Der Unterricht 34 muss vor dem neunten Unterrichtstag von 12-32 geplant werden.

Unterricht 33-34

- Die Unterrichte 33-34 müssen an maximal 4 aufeinanderfolgenden Unterrichtstagen liegen.

Unterricht 35-37 (PU)

- Die Unterrichte 35-37 müssen nach den Unterrichten 12-31 geplant werden.

Unterricht 12-37

- Die Unterrichte 12-37 müssen an maximal 16 aufeinanderfolgenden Unterrichtstagen geplant werden.

Unterricht 38-40 (G1, G2, G3)

- Die Unterrichte 38-40 müssen vor dem Unterricht 41 geplant werden.

- Wenn der letzte Unterricht von 38-40 an einem Freitag unterrichtet wird, muss der Unterricht 41 am darauffolgenden Montag liegen. Sonst muss zwischen den Unterrichten 38-40 und dem Unterricht 41 ein Tag Lernpause berücksichtigt werden.
- Die Unterrichte 38-40 müssen an maximal 4 zusammenhängenden Unterrichtstagen stattfinden.

Unterricht 41 (PG)

- Der Unterricht 41 muss nach den Unterrichten 38-40 geplant werden.
- Wenn der letzte Unterricht von 38-40 an einem Freitag stattfindet, muss der Unterricht 41 am darauffolgenden Montag liegen. Sonst muss zwischen den Unterrichten 38-40 und dem Unterricht 41 ein Tag Lernpause berücksichtigt werden.

Unterricht 38-41

- Zeitgleich dürfen maximal 3 Unterrichte von 38-41 in allen Trainings stattfinden.

Unterricht 42 (DD)

- Der Unterricht 42 darf nicht an einem Mittwoch geplant werden.
- Zeitgleich dürfen maximal 2 Unterrichte von 42 in allen Trainings stattfinden.

Bedingung Ressourcentyp

- Die Unterrichte 21, 22, 31, 32 und 35-37 benötigen jeweils einen Ressourcentyp.
- Die Unterrichte 23-30 benötigen den gleichen Ressourcentyp jeweils zweimal.

- Am Unterrichtsort von den Unterrichten 12-15, 21, 23-25 und 31 steht der Ressourcentyp elfmal zur Verfügung.
- Am Unterrichtsort von den Unterrichten 16-19, 22, 26, 27, 30 und 32 steht der Ressourcentyp viermal zur Verfügung.
- Am Unterrichtsort von Unterricht 28 steht der Ressourcentyp zweimal zur Verfügung.
- An jedem Unterrichtsort steht der Ressourcentyp für die Unterrichte 35-37 bereit.

4.3.2 Andere Lehrgänge

Nach dem Lehrgang 0001 ist der **Lehrgang 0004** der Wichtigste in der Ausbildung an der Unteroffiziersschule. Der Planungshorizont beträgt hier 49 Unterrichtstage und kann viermal pro Jahr unterrichtet werden. In einem Quartal kann der Lehrgang 0004 von 21 Trainings gleichzeitig absolviert werden. Ein Training kann aus bis zu 18 Schülern bestehen. Von den Teilnehmern werden 402 Unterrichtseinheiten visitiert.
Die nächstfolgende Schulung ist der **Lehrgang 0006**. Er wird innerhalb von 29 Unterrichtstagen absolviert und kann ebenfalls quartalsweise unterrichtet werden. Im Gegensatz zu Lehrgang 0001 und Lehrgang 0004 kann dieser lediglich von einem Training mit 25 Teilnehmern besucht werden.
Der **Lehrgang 0002** und der **Lehrgang 0003** umfassen jeweils 9 Unterrichtstage und können ganzjährig stattfinden. Diese Lehrgänge können wie der Lehrgang 0006 nur von einem Training mit 25 Teilnehmern gleichzeitig besucht werden.
Auch der **Lehrgang 0005** umfasst 9 Unterrichtstage und kann ganzjährig geplant werden. Hier können gleichzeitig 2 Trainings mit je 25 Teilnehmern pro Training unterrichtet werden. Der Lehrgang 0005 beinhaltet 73 Unterrichtseinheiten und wird je nach Bedarf gelehrt.

4.4 Bewertung der Daten

Es scheint so, als würden sich die Einschränkungen der Bundeswehr sehr von den Bedingungen an Schulen oder Universitäten unterscheiden. Jedoch ist die grundlegende Planung die gleiche. Es müssen Unterrichtseinheiten den entsprechenden Zeiten, Trainings und Ressourcen zugeteilt werden ebenso wie den Kursen an einer Universität Studenten, Professoren und Räume zugeordnet werden müssen oder Schulfächern Schulstunden, Klassen, Lehrer und Räume zugewiesen werden.
Der Unterschied zur Planung an anderen Bildungseinrichtungen ist der zeitliche Rahmen, in dem Unterrichte stattfinden müssen. Bei der Bundeswehr wird nicht im Wochen-Zyklus geplant. Die Unterrichtseinheiten eines Unterrichtseinheitstyps müssen in der Regel an zusammenhängenden Tagen unterrichtet werden und tauchen danach nicht noch einmal im Planungshorizont auf. Die Prüfungen zu den einzelnen Unterrichtsblöcken finden ebenfalls gleich am Ende der Phase, in der ein Unterrichtseinheitstyp gelehrt wurde, statt und nicht erst in einer Prüfungszeit im hinteren Viertel des Planungshorizonts.
Zusammenfassend kann sich ein Teil der Modellierung an der automatischen Erstellung von Plänen an Schulen und Universitäten orientieren. Sobald die Einschränkungen aber spezifisch werden, müssen die Bedingungen neu hergeleitet und modelliert werden.

Durch die genaue Formulierung der Bedingungen ist es möglich, ein Programm zur Unterstützung der Stundenplanerstellung zu erarbeiten. Bei den Einschränkungen im Lehrgang 0001 handelt es sich größtenteils um harte Bedingungen. Das hat zur Folge, dass der Lösungsraum sehr eingeschränkt ist, da die Anzahl der Anordnungen abhängig von der Anzahl der harten Nebenbedingungen ist. Daraus resultiert, dass eine automatische Erstellung eines Plans sehr schwierig werden kann.

5 Mathematisches Modell

Aus dem angesammelten Wissen der vorangegangenen Kapitel kann nun das mathematische Modell, das die Grundlage der Berechnung der Lösung darstellt, aufgebaut werden. Modelle sind vereinfachte Abbildungen von realen Systemen oder Problemen. Ein mathematisches Modell zeichnet Handlungsmöglichkeiten mit der Berücksichtigung von Restriktionen auf und bewertet diese [5].
Ziel des Modells ist es, eine konfliktfreie Zuweisung der Dimensionen zu finden. Das heißt, dass alle harten Nebenbedingungen erfüllt sein müssen und die weichen Nebenbedingungen so gut wie möglich berücksichtigt werden. Bei Dateninput aus der realen Welt ist es fast unmöglich, die Existenz einer Lösung zu sichern, da sich harte Nebenbedingungen gegenseitig ausschließen können. Deshalb definiert man oft ein Unterziel, in dem so viele Kurse wie möglich konfliktfrei geplant werden.
Das Ziel der Arbeit ist es, einen Stundenplan, der alle Anforderungen der Bundeswehr (Kapitel 4) berücksichtigt, für so viele Trainings wie möglich zu erstellen. Geplant wird ausschließlich der Lehrgang 0001 (Kapitel 4.3.1).

Es handelt sich bei der Stundenplanerstellung um ein gemischt-ganzzahliges Optimierungsproblem, wie es im Kapitel 3.2 vorgestellt wurde. Die Implementierung des Modells erfolgte in der Programmiersprache zimpl und wurde mit dem MIP-Solver IBM ILOG CPLEX gelöst. Weitere Ausführungen zur Implementierung findet man im Anhang im Kapitel A. Der MIP-Solver geht beim Lösen des Problems nach dem vorgestellten Branch-and-Bound-Verfahren (Kapitel 3.2.2, Seite 26ff.) vor.

5.1 Eingabedaten

Es wird ein vierdimensionales Zuordnungsmodell betrachtet. Dabei wird einer gewissen Anzahl von Trainings die zu absolvierenden Unterrichtseinheiten in Abhängigkeit von dem Tag und der Zeit zugeordnet.
Sei K die Menge der Trainings.

Training (Klassen)

$$K = \{1, ..., 26\}.$$

Der Stundenplan muss für 26 Trainings, die den Lehrgang 0001 im gleichen Planungshorizont absolvieren, erstellt werden.
Sei U die Menge alle Unterrichte des Lehrgangs.

Unterricht

$$U = \{1, ..., 42\}.$$

Die Unterrichte haben gewisse Einschränkungen, die im Vorfeld definiert werden müssen. Dazu zählen die Bedingungen, dass einige Unterrichte nur am Freitag bzw. nicht am Freitag stattfinden dürfen. Sei F die Menge der Unterrichte, die an einem Freitag stattfinden und NF die Mengen der Unterrichte, die nicht an einem Freitag unterrichtet werden.

Unterrichte am Freitag

$$F = \{20, 33\}$$

Unterrichte nicht am Freitag

$$NF = \{5, ..., 10, 42\}.$$

Des Weiteren sei HT die Menge der Unterrichte, die nur einen halben Tag in Anspruch nehmen.

Halbtagsunterrichte

$$HT = \{4, 11, 20, 33, 39, 40, 41\}.$$

Nicht selten muss bei den Unterrichten die Bereitstellung eines Ressourcentyps beachtet werden. Bei der Stundenplanung für die Bundeswehr gibt es nur einige Unterrichte, die alle mit einem gemeinsamen Ressourcentyp in Verbindung stehen. Sei R die Menge, die diese Unterrichte beinhaltet.

Unterrichte mit Ressourcentyp

$$R = \{12, ..., 19, 21, ..., 32, 35, ..., 37\}.$$

In den Mengen R_{11}, R_4 und R_2 ist die Kombination von Unterrichten enthalten, die den gemeinsamen Ressourcentyp in unterschiedlicher Anzahl zur Verfügung stellt.

11xRessource

$$R_{11} = \{12, ..., 15, 21, 31, 23, ..., 25, 29\}.$$

4xRessource

$$R_4 = \{16, ..., 19, 22, 26, 27, 30, 32\}.$$

2xRessource

$$R_2 = \{28\}.$$

Die Menge R_{11} stellt dabei die Ressource 11-mal bereit, wohingegen die Menge R_4 den Ressourcentyp nur 4-mal vorweisen kann und die Menge R_2 lediglich 2-mal. Sei T die Menge der Tage des Planungshorizonts.

Tage

$$T = \{1, ...51\}.$$

Unter 4.3.1 sind 52 Tage aufgeführt, dazu zählt aber der Anreisetag, an dem kein Unterricht stattfinden kann. Es können also 51 Tage geplant werden.
Der Unterschied zu einem Stundenplan in einer Schule ist hier, dass nicht jeder Tag geplant wird. An Tagen, an denen kein Unterricht vom Lehrgang 0001 stattfindet, kann ein anderer Lehrgang besucht werden oder das Training wird von dem zugeteilten Betreuer unterrichtet. Es können auch andere Ereignisse wie zum Beispiel Tagesausflüge auf dem Plan stehen.
Da keine explizite Unterteilung in die einzelnen Wochentage erfolgt, werden diese in den folgenden Mengen zugeordnet.

Montag

$$Mo = \{3, 8, 13, 21, 26, 31, 33, 38, 43, 48\}.$$

Dienstag

$$Di = \{4, 9, 14, 22, 27, 32, 34, 39, 44, 49\}.$$

Mittwoch

$$Mi = \{5, 10, 15, 18, 23, 28, 35, 40, 45, 50\}.$$

Donnerstag

$$Do = \{1, 6, 11, 16, 19, 24, 29, 36, 41, 46, 51\}.$$

Freitag

$$Fr = \{2, 7, 12, 17, 20, 25, 30, 37, 42, 47\}.$$

Die Dimension der Zeit muss ebenfalls mit einer Menge definiert werden. Hierfür sei Z die Menge der Unterrichtszeiten.

Zeiten

$$Z = \{v, n\}.$$

Die Stundenplanung erfolgt auf Datenbasis von Ganztags- und Halbtagsunterrichten. Deswegen wird ein Tag ausschließlich in Vormittag und Nachmittag unterteilt. Auf eine detailliertere Zeiteinteilung wird verzichtet.

Sei d_u der Parameter, der dem Unterricht u die jeweilige Dauer zuordnet.

$$\begin{aligned} d_u &= 2, \quad \forall u \in U/HT, \\ d_u &= 1, \quad \forall u \in HT. \end{aligned}$$

Für eine übersichtlichere Darstellung werden die Unterrichtseinheiten zusammengefasst. Ist $d_u = 1$ bedeutet das, dass u ein Halbtagsunterricht ist. Ist hingegen $d_u = 2$, umfasst die Planungszeit des Unterrichts u einen ganzen Tag.
Durch verschiedene Einschränkungen kann nicht jeder Unterricht beliebig oft am Tag geplant werden. Der Parameter l_u limitiert, in wie vielen Trainings parallel der gleiche Unterricht stattfinden kann.

$$\begin{aligned} l_u &= 3, \quad \forall u \in \{1, ..., 4, 8, ..., 10\}, \\ l_u &= 2, \quad \forall u \in \{5, 42\}, \\ l_u &= 26, \quad \forall u \in U \backslash \{1, ..., 5, 8, ..., 10, 42\}. \end{aligned}$$

Des Weiteren gibt es Unterrichte, die einen Ressourcentyp benötigen. Sei b_u die Anzahl der benötigten Ressourcen in den einzelnen Unterrichten.

$$\begin{aligned} b_u &= 0, \quad \forall u \in \{12, ..., 19\}, \\ b_u &= 1, \quad \forall u \in \{21, 22, 31, 32, 35, ..., 37\}, \\ b_u &= 2, \quad \forall u \in \{23, ..., 30\}. \end{aligned}$$

Unter Kapitel 4.3.1 wird zusätzlich beschrieben, dass es Unterrichte mit einer Alternative gibt. Der Parameter a_u ordnet diesen Unterrichten ihre Alternativen zu.

$$\begin{aligned} a_u &= u, \quad \forall u \in \{1, ..., 11, 20, 33, ..., 42\}, \\ a_{21} &= 22, \quad a_{22} = 21, \\ a_{25} &= 28, \quad a_{28} = 25, \\ a_{29} &= 30, \quad a_{30} = 29, \\ a_{31} &= 32, \quad a_{32} = 31, \\ a_{\{12,...,15\}} &= \{16, ..., 19\}, \quad a_{\{16,...,19\}} = \{12, ..., 15\}, \\ a_{\{23,24\}} &= \{26, 27\}, \quad a_{\{26,27\}} = \{23, 24\}. \end{aligned}$$

Für das beschriebene Problem der Stundenplanerstellung für die Unteroffiziersschule ergeben sich schlussendlich 16 Mengen und 4 Parameter, die als Eingabedaten übergeben werden und im Vorfeld deklariert werden müssen.

5.2 Nebenbedingungen

Die Einschränkungen an den Stundenplan werden mit Hilfe von Nebenbedingungen formuliert. Die Bedingungen untergliedern sich in harte und weiche Nebenbedingungen. Das heißt, dass es Bedingungen gibt, die nicht missachtet werden dürfen und solche, die bei Nichteinhaltung die Lösungsqualität mindern. Im Kapitel 4.3.1 werden alle Bedingungen an den Lehrgang 0001 aufgeführt.

5.2.1 Harte Nebenbedingungen

Die Einschränkungen der Bundeswehr sind in der Mehrzahl feste Vorgaben. Diese müssen unter allen Umständen eingehalten werden.
Der Planungshorizont des Stundenplans ist beschränkt. Das heißt, dass die Unterrichtseinheiten von Lehrgang 0001 nur in einem bestimmten Zeitraum stattfinden dürfen. Um die Bedingung zu modellieren, benötigt man eine Variable, die den Unterrichtseinheiten die Zeiten im Stundenplan zuordnet. Dafür definiert man

eine Binärvariable $x_{k,t,z,u}$, die von den Mengen Training K, Tag T, Zeit Z und Unterricht U abhängt:

$$x_{k,t,z,u} = \begin{cases} 1 & \text{Training } k \text{ hat am Tag } t \text{ zur Zeit } z \text{ den Unterricht } u \\ & \forall\, k \in K, t \in T, z \in Z, u \in U, \\ 0 & \text{sonst.} \end{cases}$$

Alle Variablen $x_{k,t,z,u}$, die außerhalb des Planungshorizonts liegen, müssen 0 sein. Der Planungszeitraum umfasst 51 Tage, wobei die Hälfte der Lehrgänge drei Tage früher anfängt und drei Tage früher endet. Die Mengen K und T werden hierfür unterteilt:

$$\begin{aligned} K_1 &= \{1, ..., 13\}, \\ K_2 &= \{14, ..., 26\}, \\ T_1 &= \{1, ..., 48\}, \\ T_2 &= \{4, ..., 51\}. \end{aligned}$$

Für die Einhaltung des Zeitraums werden die Binärvariablen $x_{k,t,z,u}$ über alle Tage t, die nicht im Planungshorizont liegen, addiert und 0 gesetzt. Es ergeben sich die beiden folgenden Nebenbedingungen:

$$\begin{aligned} \sum_{t \notin T_1,\, z \in Z,\, u \in U} x_{k,t,z,u} &= 0, \quad \forall k \in K_1, \\ \sum_{t \notin T_2,\, z \in Z,\, u \in U} x_{k,t,z,u} &= 0, \quad \forall k \in K_2. \end{aligned}$$

Des Weiteren darf eine Zeiteinheit maximal mit einem Unterricht belegt werden. Dafür definiert man, dass die Summe über die Unterrichte in jedem Training zu jedem Zeitpunkt maximal 1 sein darf:

$$\sum_{u \in U} x_{k,t,z,u} \leq 1, \quad \forall k \in K, t \in T, z \in Z.$$

In der Unteroffiziersschule der Bundeswehr wird am Freitagnachmittag nicht unterrichtet. Das bedeutet, dass der Zeitslot mit keinem Unterricht belegt werden darf. Daraus ergibt sich, dass die Addition der Variablen $x_{k,t,z,u}$ am Freitagnach-

mittag über die Unterrichte in U für alle Trainings gleich 0 sein muss:

$$\sum_{u \in U} x_{k,t,n,u} = 0, \quad \forall k \in K, t \in Fr.$$

Eine weitere Voraussetzung ist, dass alle Unterrichte gelehrt werden müssen. Im Vorherigen wurde dafür ein Parameter d_u eingeführt, der dem jeweiligen Unterricht die Anzahl der Unterrichtseinheiten zuordnet, wobei die Unterrichtseinheiten nun in Halbtagsblöcke zusammengefasst sind. Für die Einhaltung summiert man für alle Trainings die Zeiteinheiten des jeweiligen Unterrichts u durch die Binärvariable $x_{k,t,z,u}$ und setzt die Summe gleich dem Parameter d_u:

$$\sum_{t \in T, z \in Z} x_{k,t,z,u} = d_u, \quad \forall k \in K, u \in U.$$

Zusätzlich muss beachtet werden, dass einige Unterrichte eine Alternative haben. Es darf nur eine der beiden Möglichkeiten geplant werden. Zur Hilfe definiert man eine Menge A_u, die den originalen Unterricht u enthält und den Alternativunterricht a_u:

$$A_u = \{u, a_u\}.$$

Die obere Bedingung der Unterrichtsbelegung wird nun folgendermaßen umformuliert:

$$\sum_{u \in A_u} \sum_{t \in T, z \in Z} x_{k,t,z,u} = d_u, \quad \forall k \in K, u \in U.$$

Des Weiteren gibt es die Einschränkung, dass der Unterricht $u = \{1, 2, 3\}$ nicht am ersten Tag des Planungshorizonts unterrichtet werden darf. Es wird definiert, dass $x_{k,t,z,u}$ gleich 0 ist für den Unterricht $u = \{1, 2, 3\}$ am ersten Tag des Zeitraums für die ersten 13 Trainings und $x_{k,t,z,u}$ gleich 0 ist für den vierten Tag der anderen 13 Trainings.

$$\begin{aligned} x_{k,1,z,u} &= 0, \quad \forall k \in K_1, z \in Z, u \in \{1, 2, 3\}, \\ x_{k,4,z,u} &= 0, \quad \forall k \in K_2, z \in Z, u \in \{1, 2, 3\}. \end{aligned}$$

Eine weitere harte Nebenbedingung ist, dass die Unterrichte der Menge NF nicht am Freitag stattfinden dürfen. Das heißt, dass die Binärvariable $x_{k,t,z,u}$ gleich 0 ist

für alle Unterrichte der Menge NF:

$$x_{k,t,v,u} = 0, \quad \forall k \in K, t \in Fr, u \in NF.$$

Unter 4.3.1 wird außerdem vorausgesetzt, dass $u = 42$ nicht an einem Mittwoch geplant werden darf. Die Variable $x_{k,t,z,42}$ ist für alle Mittwoche gleich 0:

$$x_{k,t,z,42} = 0, \quad \forall k \in K, t \in Mi, z \in Z.$$

Zusätzlich gibt es Einschränkungen bezüglich der Alternativunterrichte. Es darf entweder nur der originale Unterricht oder die Alternative geplant werden. Hierfür bestimmt man eine neue Binärvariable $v_{k,u}$, die angibt, ob im Training k der Unterricht u stattfindet:

$$v_{k,u} = \begin{cases} 1 & \text{Training } k \text{ hat den Unterricht } u \\ & \forall\ k \in K, u \in U, \\ 0 & \text{sonst.} \end{cases}$$

Die Einschränkung muss mit Hilfe von drei Nebenbedingungen formuliert werden.
Der Unterricht $u = 21$ hat eine Alternative $u = 22$. Wenn $u = 21$ geplant wird, darf $u = 22$ nicht im Stundenplan vorkommen. Durch die neu eingeführte Variable $v_{k,u}$ lässt sich Folgendes formulieren.

$$v_{k,21} + v_{k,22} = 1, \quad \forall k \in K$$

Daraus resultiert, dass eine der beiden Binärvariablen auf 1 gesetzt wird. Die Variablen $v_{k,u}$ hängen von $x_{k,t,z,u}$ ab.

$$x_{k,t,z,21} \leq v_{k,21}, \quad \forall k \in K, t \in T, z \in Z,$$
$$x_{k,t,z,22} \leq v_{k,22}, \quad \forall k \in K, t \in T, z \in Z.$$

Das heißt, dass wenn der Unterricht $u = 21$ geplant wird, ist die Variable $x_{k,t,z,21} = 1$ und somit ist auch die Variable $v_{k,21} = 1$. Damit muss gelten $v_{k,22} = 0$ und somit

auch $x_{k,t,z,22} = 0$. Im Allgemeinen formuliert man:

$$\sum_{u \in A_u} v_{k,u} = 1, \quad \forall k \in K, u \in U,$$
$$x_{k,t,z,u} \leq v_{k,u}, \quad \forall k \in K, t \in T, z \in Z, u \in U.$$

In den Bedingungen der Unterrichte ist außerdem vorgeschrieben, in welcher Reihenfolge die Module geplant werden müssen. Für die Umsetzung der Bedingung definiert man eine Funktion, die den Positionen im Stundenplan einen Wert in aufsteigender Form zuweist.

Slotindex

$$s(t,z) = 2 \cdot (t-1) + \mathbb{1}_{\{z=n\}}.$$

Mit Hilfe der Funktion kann nun modelliert werden, dass ein Unterricht $\tilde{u}$ erst nach einem anderen Unterricht u stattfinden darf. Die Variable $x_{k,\tilde{t},\tilde{z},\tilde{u}}$ muss für alle Zeiteinheiten vor dem Unterricht $u$gleich 0 sein:

$$x_{k,\tilde{t},\tilde{z},\tilde{u}} \leq 1 - x_{k,t,z,u}, \quad \forall (\tilde{t},\tilde{z}) \in \{T \times Z : s(\tilde{t},\tilde{z}) \leq s(t,z)\}$$
$$\forall k \in K, t \in T, z \in Z, u \in \{1,2,3\}, \tilde{u} \in \{4\}.$$

Die Mengen von u und $\tilde{u}$ können beliebig verändert werden. Die Bedingung kann für alle Unterrichte definiert werden, die eine Einschränkung bezüglich ihrer Reihenfolge haben.
Eine weitere Einschränkung bei der Reihenfolge ist, dass gewisse Unterrichte spätestens nach dem ersten Unterrichtstag eines anderen Unterrichts stattfinden müssen. Die Bedingung kann wie die vorherige formuliert werden mit dem Unterschied, dass nun zwei Unterrichtseinheiten des Unterrichts, der vor dem anderen liegen muss, mehr verplant werden können:

$$\sum_{\substack{\tilde{t} \in T, \tilde{z} \in Z: \\ s(\tilde{t},\tilde{z}) \leq s(t,z)}} x_{k,\tilde{t},\tilde{z},\tilde{u}} \leq 2 + (1 - x_{k,t,z,u}), \quad \forall k \in K, t \in T, z \in Z, u \in \{1,2,3\}, \tilde{u} \in \{4\}.$$

Auch hier sind die Mengen von u und $\tilde{u}$ nur exemplarisch angegeben.
Weiterhin gibt es Unterrichte, die erst stattfinden dürfen, nachdem ein Tag von

einem anderen Unterricht gelehrt wurde. Das heißt, dass die Summe über die Unterrichte $\tilde{u}$ mindestens 1 sein muss für alle Zeiteinheiten vor dem Unterricht u:

$$\sum_{\substack{\tilde{t}\in T,\tilde{z}\in Z:\\ s(\tilde{t},\tilde{z})\leq s(t,z)}} x_{k,\tilde{t},\tilde{z},\tilde{u}} \geq x_{k,t,z,u}, \quad \forall k \in K, t \in T, z \in Z, u \in \{1,2,3\}, \tilde{u} \in \{4\}.$$

Zusätzlich müssen einige Unterrichtskurse innerhalb einer fest vorgegebenen Zeit unterrichtet werden. Für die Formulierung der Nebenbedingung wird eine neue Binärvariable $z_{k,t}$ deklariert:

$$z_{k,t} = \begin{cases} 1 & \text{Training } k \text{ und Tag } t \\ & \forall\ k \in K, t \in T, \\ 0 & \text{sonst.} \end{cases}$$

Man summiert nun die Binärvariable $x_{k,t,z,u}$ über einen bestimmten Zeithorizont (m Tage) $t \leq \tilde{t} \leq t + m$ und definiert, dass die Summe mindestens der Anzahl der zu planenden Unterrichtseinheiten des jeweiligen Unterrichts entspricht. Da die Bedingung nur einmal für ein Training gilt, multipliziert man d_u mit der neu definierten Binärvariable $z_{k,t}$, die für alle Tage des Planungszeitraums einmal 1 ist:

$$\sum_{\substack{k\in K,\tilde{t}\in T,z\in Z:\\ t\leq\tilde{t}\leq t+m}} x_{k,\tilde{t},z,u} \geq d_u z_{k,t}, \quad \forall k \in K, t \in T, u \in \{1,2,3\},$$

$$\text{wobei gilt } \sum_{t\in T} z_{k,t} = 1, \quad \forall k \in K.$$

Des Weiteren gibt es eine zeitliche Einschränkung, die besagt, wie viele Trainings einen Unterricht zeitgleich besuchen können. Dafür wurde bereits der Parameter l_u eingeführt. Es wird die Variable $x_{k,t,z,u}$ über alle Trainings k summiert und kleiner gleich der oberen Schranke l_u gesetzt:

$$\sum_{k\in K} x_{k,t,z,u} \leq l_u, \quad \forall t \in T, z \in Z, u \in U.$$

Zusätzlich gibt es eine Einschränkung bestimmter Unterrichtstypen ($u \in R$) auf Grund des Ressourcenbedarfs. Der Parameter b_u ordnet dem Unterricht die benötigte Anzahl an Ressourcen zu. Sei n die Anzahl der zur Verfügung stehenden Ressourcen. Die Summe der Binärvariable $x_{k,t,z,u}$ über alle Unterrichte u, die den

Ressourcentyp benötigen ($u \in R_n$), multipliziert mit der benötigten Anzahl muss für alle Zeiteinheiten kleiner gleich der maximalen Anzahl n sein:

$$\sum_{k \in K, u \in R_n} x_{k,t,z,u} \cdot b_u \leq n, \quad \forall t \in T, z \in Z.$$

Außerdem wird in Kapitel 4.3.1 vorgeschrieben, dass zwischen Unterrichtseinheiten eine Lernpause von einem Tag eingehalten werden muss. Das heißt, dass der Folgetag nicht mit dem Unterricht $\tilde{u}$ belegt werden darf. Man modelliert, dass die Summe der $x_{k,t,z,u}$ und $x_{k,t+1,z,\tilde{u}}$ maximal 1 ist:

$$x_{k,t,z,u} + x_{k,t+1,z,\tilde{u}} \leq 1, \quad \forall k \in K, t \in T, z \in Z, u, \tilde{u} \in U.$$

Eine weitere Nebenbedingung, die für alle Unterrichte gilt, ist, dass an jedem Tag nur ein Typ von Unterricht stattfinden darf. Ähnlich wie in der vorherigen Bedingung definiert man, dass die Summe der $x_{k,t,z,u}$ an einem Tag für verschiedene Unterrichte maximal 1 ist:

$$\begin{aligned} x_{k,t,v,u} + x_{k,t,n,\tilde{u}} &\leq 1, \quad \forall k \in K, t \in T, u, \tilde{u} \in U, \\ x_{k,t,n,u} + x_{k,t,v,\tilde{u}} &\leq 1, \quad \forall k \in K, t \in T, u, \tilde{u} \in U. \end{aligned}$$

5.2.2 Weiche Nebenbedingungen

Weiche Nebenbedingungen sind Einschränkungen des Stundenplans, die nicht zwingend berücksichtigt werden müssen. Um dennoch zu garantieren, dass möglichst viele weiche Bedingungen eingehalten werden, definiert man Variablen y, die das Verletzen weicher Nebenbedingungen bestrafen. Die y-Variablen werden in der Zielfunktion addiert. Durch das Minimieren der Zielfunktion werden somit so viele weiche Nebenbedingungen wie möglich berücksichtigt.
Zu den weichen Nebenbedingungen zählt, dass einige Unterrichte am Freitag stattfinden sollen. Man definiert, dass die Summe der Binärvariable $x_{k,t,z,u}$, der Unterrichte, die am Freitag geplant werden sollen, und einer y-Variable über alle Freitage gleich der vorgeschriebenen Anzahl der Unterrichtsblöcke d_u ist. Wenn der Unterricht u nicht am Freitag stattfinden kann, erhöht sich die y-Variable und

somit der Zielfunktionswert:

$$\sum_{u \in F} x_{k,t,v,u} + y_1 = \sum_{u \in F} d_u, \quad \forall k \in K, t \in Fr.$$

Als eine weitere weiche Nebenbedingung wird formuliert, dass verschiedene Unterrichte nicht an einem Tag stattfinden sollen. Die gleiche Bedingung wurde als eine harte Bedingung aufgeführt und wird hier als eine Option angegeben. Als weiche Nebenbedingung modelliert man, dass $x_{k,t,v,u}$ abzüglich $x_{k,t,n,u}$ kleiner als eine Binävariable $y_{2_{k,t}}$ ist:

$$y_{2_{k,t}} = \begin{cases} 1 & \text{Training } k \text{ und Tag } t \\ & \forall\ k \in K, t \in T, \\ 0 & \text{sonst.} \end{cases}$$

Falls am Vormittag und am Nachmittag nicht der gleiche Unterricht geplant ist, folgt $y_{2_{k,t}} = 1$:

$$x_{k,t,v,u} - x_{k,t,n,u} \leq y_{2_{k,t}} \quad \forall k \in K, t \in T, u \in U \backslash HT.$$

5.3 Zielfunktion

Wie bei der Definition der weichen Nebenbedingungen schon erwähnt, werden in der Zielfunktion die Variablen summiert, die die Nichteinhaltung der weichen Nebenbedingungen bestrafen. Das Ziel ist es, einen möglichst optimalen Stundenplan zu erstellen, das durch die Minimierung der Summe, der y-Variablen umgesetzt wird. Das heißt, dass so viele weiche Nebenbedingungen wie möglich berücksichtigt werden. Für die Zielfunktion ergibt sich folgende Gestalt:

$$\min \quad y_1 + \sum_{k \in K, t \in T} y_{2_{k,t}}.$$

6 Rechenergebnisse

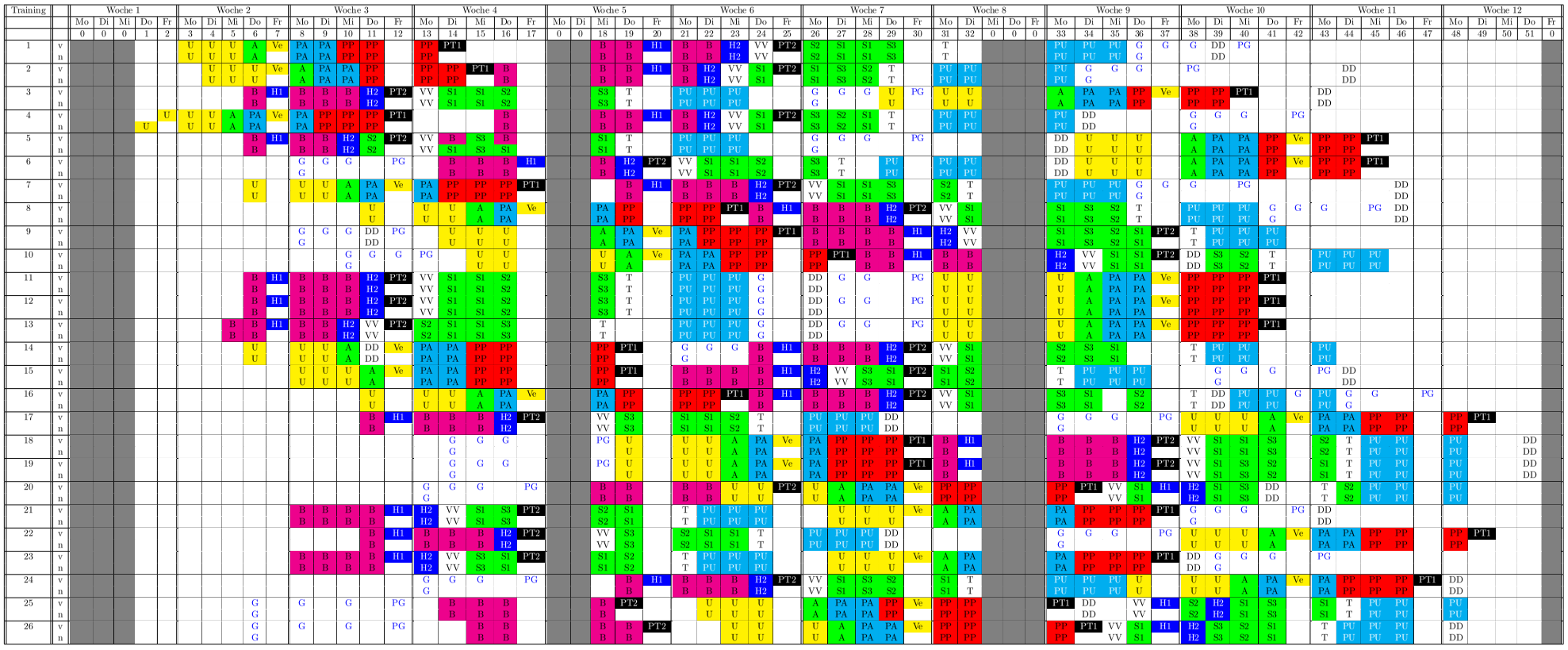
Die Lösungsfindung geschieht mit Hilfe eines Verfahrens (Kapitel 5), das unter Verwendung der bereitgestellten Daten (Kapitel 4) das mathematische Modell (Kapitel 3) löst. Im Anschluss muss die erhaltene Lösung analysiert und als akzeptabel und brauchbar bewertet werden [5].

Ziel dieser Arbeit war es, einen zulässigen Stundenplan für eine Unteroffiziersschule der Bundeswehr mit Hilfe der ganzzahligen Programmierung zu erarbeiten. In dem Kapitel 2.1.2 wurde die Veröffentlichung von MirHassani vorgestellt in der er erwähnt, dass die ganzzahlige Optimierung eine zuverlässige Lösungsmethode des Timetabling-Problems sei, jedoch der jeweilige Rechenaufwand stets in Relation zu dem Problem zu setzen ist. Die Laufzeiten der Algorithmen können bei einer großen Anzahl von Variablen unwirtschaftlich steigen [13]. Auch in der hier modellierten Rechenvorschrift ist das der Fall. Der Algorithmus, der einen Rechenaufwand von exponentieller Größenordnung besitzt, findet in keiner angemessenen Zeit eine Lösung des beschriebenen Stundenplanproblems, das 91.340 Variablen und 3.816.809 Nebenbedingungen umfasst.
Alle Berechnungen, die im Folgenden vorgestellt werden, wurden auf einem Rechner mit dem Betriebssystem Linux, 256 GB RAM und Intel Xeon E5-2667 Multicore-Prozessoren ($16 \cdot 8$ Prozessorkerne) mit 3,3 GHz auf einem Prozessorkern ausgeführt.

6.1 Auswertung des Stundenplans der Bundeswehr

Die Abbildung 6.1 zeigt den bisherigen Plan der Unteroffiziersschule.

Abbildung 6.1: Aktueller Stundenplan der Unteroffiziersschule der Bundeswehr



Mit Hilfe des implementierten Programms wurde getestet, inwieweit der vorliegende Plan den unter 4.3.1 formulierten Bedingungen entspricht. Hierfür werden die einzelnen Trainings unabhängig voneinander betrachtet.

Training 1 und 6

- Es wird die Voraussetzung missachtet, dass der Unterricht 33 spätestens nach dem ersten Unterrichtstag von 12-32 stattfinden muss.
- Die Bedingung, dass die Unterrichte 12-37 an maximal 16 zusammenhängenden Tagen unterrichtet werden müssen, wird ebenfalls nicht eingehalten.

Training 2

- Die Bedingung, dass die Unterrichte 12-37 an maximal 16 zusammenhängenden Tagen unterrichtet werden müssen, wird nicht eingehalten.
- Des Weiteren wird nicht berücksichtigt, dass der Unterricht 05 vor dem Unterricht 04 stattfinden muss.

Training 3, 7, 8, 16, 17, 22 und 24

- Die Bedingung, dass die Unterrichte 12-37 an maximal 16 zusammenhängenden Tagen unterrichtet werden müssen, wird nicht eingehalten.
- Außerdem werden die Unterrichte 33 und 34 nicht an 4 zusammenhängenden Tagen geplant.

Training 4

- Die Bedingung, dass die Unterrichte 12-37 an maximal 16 zusammenhängenden Tagen unterrichtet werden müssen, wird nicht eingehalten.
- Es wird die Voraussetzung missachtet, dass der Unterricht 33 spätestens nach dem ersten Unterrichtstag von 12-32 stattfinden muss.

- Des Weiteren werden die Unterrichte 12-15 (16-19) nicht an 5 zusammenhängenden Tagen geplant.
- Außerdem wurde missachtet, dass am ersten Tag des Planungshorizonts der Unterricht 01-03 nicht durchgeführt werden darf.

Training 5

- Der Plan für das Training 5 verletzt die Voraussetzung, dass die Unterrichte 12-37 an maximal 16 zusammenhängenden Tagen unterrichtet werden müssen.
- Des Weiteren wird einer der Unterrichte 23-24 (26-28) und 12-15 (16-19) nicht an einem Tag geplant. Der Zielfunktionswert ist somit 2.
- Außerdem wird nicht beachtet, dass die Unterrichte 12-15 (16-19) an maximal 4 zusammenhängenden Tagen geplant werden müssen.
- Die Bedingung, dass die Unterrichte 23-25 und 29 (26-28 und 30) an maximal 6 zusammenhängenden Tagen stattfinden müssen, wird ebenfalls nicht berücksichtigt.
- Weiterhin wird die Einschränkung, dass der Unterricht 12-15 (16-19) vor dem Unterricht 20 liegen muss, nicht eingehalten.
- Eine weitere Vorgabe, die nicht berücksichtigt wird, ist dass der Unterricht 21 (22) vor den Unterrichten 23-25 und 29 (26-28 und 30) geplant werden muss.

Training 9 und 15

- Es wird die Voraussetzung missachtet, dass der Unterricht 33 spätestens nach dem ersten Unterrichtstag von 12-32 stattfinden muss.
- Zusätzlich wird nicht beachtet, dass der Unterricht 20 spätestens nach einem der Unterrichte von 23-25 und 29 (26-28 und 30) geplant werden muss.

Training 10 und 11

- Die Bedingung, dass die Unterrichte 12-37 an maximal 16 zusammenhängenden Tagen unterrichtet werden müssen, wird nicht eingehalten.
- Es wird die Voraussetzung missachtet, dass der Unterricht 33 spätestens nach dem ersten Unterrichtstag von 12-32 stattfinden muss.
- Zusätzlich wird nicht beachtet, dass der Unterricht 20 spätestens nach einem der Unterrichte von 23-25 und 29 (26-28 und 30) geplant werden muss.

Training 12

- Die Bedingung, dass die Unterrichte 12-37 an maximal 16 zusammenhängenden Tagen unterrichtet werden müssen, wird nicht eingehalten.
- Außerdem werden die Unterrichte 33 und 34 nicht an 4 zusammenhängenden Tagen geplant.
- Des Weiteren finden die Unterrichte 38-40 nicht an 4 zusammenhängenden Tagen statt.

Training 13

- Die Bedingung, dass die Unterrichte 12-37 an maximal 16 zusammenhängenden Tagen unterrichtet werden müssen, wird nicht eingehalten.
- Es wird die Voraussetzung missachtet, dass der Unterricht 33 spätestens nach dem ersten Unterrichtstag von 12-32 stattfinden muss.
- Des Weiteren finden die Unterrichte 38-40 nicht an 4 zusammenhängenden Tagen statt.

Training 14

- Der Unterricht 41 wird nicht im Stundenplan aufgeführt. Alle Einschränkungen, die mit diesem Unterricht verbunden sind, können nicht geprüft werden.

- Die Bedingung, dass die Unterrichte 12-37 an maximal 16 zusammenhängenden Tagen unterrichtet werden müssen, wird nicht eingehalten.
- Des Weiteren wird nicht berücksichtigt, dass die Unterrichte 05-10 an nicht mehr als 7 aufeinanderfolgenden ganzen Unterrichtstagen (Montag - Donnerstag) geplant werden müssen.
- Außerdem werden die Unterrichte 35-37 nicht an maximal 4 zusammenhängenden Tagen gehalten.

Training 18

- Die Bedingung, dass die Unterrichte 12-37 an maximal 16 zusammenhängenden Tagen unterrichtet werden müssen, wird nicht eingehalten.
- Der Unterricht 33 soll an einem Freitag stattfinden. Diese Bedingung wird nicht berücksichtigt, weshalb sich der Zielfunktionswert auf 1 erhöht.

Training 20

- Die Bedingung, dass die Unterrichte 12-37 an maximal 16 zusammenhängenden Tagen unterrichtet werden müssen, wird nicht eingehalten. Hier werden die Unterrichte nicht an zusammenhängenden Tagen geplant, sondern in zwei Gruppen geteilt, die unabhängig voneinander berücksichtigt werden.
- Des Weiteren werden die Unterrichte 23-25 und 29 (26-28 und 30) nicht an maximal 6 zusammenhängenden Unterrichtstagen geplant.
- Es wird die Voraussetzung missachtet, dass der Unterricht 33 spätestens nach dem ersten Unterrichtstag von 12-32 stattfinden muss.

Training 21

- Die Bedingung, dass die Unterrichte 12-37 an maximal 16 zusammenhängenden Tagen unterrichtet werden müssen, wird nicht eingehalten.

- Es wird die Voraussetzung missachtet, dass der Unterricht 33 spätestens nach dem ersten Unterrichtstag von 12-32 stattfinden muss.
- Zusätzlich wird nicht beachtet, dass der Unterricht 20 spätestens nach einem der Unterrichte von 23-25 und 29 (26-28 und 30) geplant werden muss.
- Des Weiteren wird nicht berücksichtigt, dass der Unterricht 04 frühestens nach dem Unterricht 05 durchgeführt werden darf.

Training 25

- Die Bedingung, dass die Unterrichte 12-37 an maximal 16 zusammenhängenden Tagen unterrichtet werden müssen, wird nicht eingehalten. Hier werden die Unterrichte nicht an zusammenhängenden Tagen geplant, sondern in zwei Gruppen aufgeteilt, die am Anfang und am Ende des Planungshorizonts berücksichtigt werden.
- Des Weiteren finden die Unterrichte 38-40 nicht an 4 zusammenhängenden Tagen statt.
- Es wird die Voraussetzung missachtet, dass der Unterricht 33 spätestens nach dem ersten Unterrichtstag von 12-32 gehalten werden muss.
- Der Unterricht 20 soll an einem Freitag geplant werden. Diese Bedingung wird nicht berücksichtigt, weshalb sich der Zielfunktionswert auf 1 erhöht.

Training 26

- Die Bedingung, dass die Unterrichte 12-37 an maximal 16 zusammenhängenden Tagen unterrichtet werden müssen, wird nicht eingehalten. Hier werden die Unterrichte nicht an zusammenhängenden Tagen geplant, sondern in zwei Gruppen aufgeteilt, die am Anfang und am Ende des Planungshorizonts berücksichtigt werden.
- Des Weiteren finden die Unterrichte 38-40 nicht an 4 zusammenhängenden Tagen statt.
- Es wird die Voraussetzung missachtet, dass der Unterricht 33 spätestens nach dem ersten Unterrichtstag von 12-32 gehalten werden muss.

In den vorangehenden Paragraphen wird gezeigt, dass kein Training korrekt geplant wurde. Es fällt auf, dass fast alle Trainings die Nebenbedingungen, dass die Unterrichte 12-37 an maximal 16 zusammenhängenden Tagen unterrichtet werden müssen, verletzen. Des Weiteren werden die Einschränkungen rund um den Unterricht 33 in vielen Trainingsplänen nicht berücksichtigt.
Zusammenfassend kann gesagt werden, dass der jetzige Stundenplan der Unteroffiziersschule der Bundeswehr nicht alle Nebenbedingungen, die unter 4.3.1 formuliert wurden, erfüllt. Die Einschränkungen sollten dahingehend überarbeitet werden, dass harte Bedingungen gegebenenfalls in weiche Nebenbedingungen umformuliert werden.

6.2 Stundenplan für ein Training

Die Erstellung des Stundenplans für ein Training dauert 3.832,99 Sekunden. Es ergeben sich 3.209 Variablen und 146.662 Nebenbedingungen. Hierbei ist festzustellen, dass der Plan keine optimale Lösung besitzt. Der kleinste Zielfunktionswert ist 3. Das bedeutet, dass weiche Nebenbedingungen verletzt werden. Der Stundenplan ist in Abbildung 6.2 dargestellt.

Training		Woche 1					Woche 2					Woche 3					Woche 4				
		Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr
		0	0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
1	v				DD				U	U					U		A	PA	PA	PP	Ve
	n				DD				U	U					U		A	PA	PA	PP	

Training		Woche 5					Woche 6					Woche 7					Woche 8				
		Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr
		0	0	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	0	0	0
1	v			PP	PP	PT1	T	H2	VV	B	T	B	B	B	S2	PT2	S1	S1			
	n			PP	PP		H1	H2	VV	B		B	B	B	S2		S1	S1			

Training		Woche 9					Woche 10					Woche 11					Woche 12				
		Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr
		33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	0
1	v	S3	PU	PU	PU										G		G	G		PG	
	n	S3	PU	PU	PU												G				

Abbildung 6.2: Stundenplan für ein Training ohne Anpassung der Nebenbedingungen.

Die nicht berücksichtigten Bedingungen sind zum einen, dass gleiche Unterrichte nicht an einem Tag stattfinden, zum anderen, dass der Unterricht, der am Freitag geplant werden sollte, an einem anderen Tag stattfindet.

Die Analyse des jetzigen Plans der Bundeswehr, die im Kapitel 6.1 aufgezeigt ist, ergab, dass eine Nebenbedingung bei fast allen Trainings verletzt wird. Passt man die Nebenbedingung, dass die Unterrichte 12-37 an maximal 16 zusammenhängenden Tagen unterrichtet werden müssen an, so dass die Unterrichte 12-37 an 18 zusammenhängenden Tagen geplant werden, ergibt sich ein Zielfunktionswert von 0 für ein Training. Das heißt, dass es nun möglich ist, einen optimalen Stundenplan für ein Training in 362,23 Sekunden zu ermitteln. Einen möglichen optimalen Stundenplan für ein Training zeigt Abbildung 6.3.

Abbildung 6.3: Stundenplan für ein Training mit Anpassung der Nebenbedingung, dass die Unterrichte 12-37 an maximal 18 zusammenhängenden Tagen unterrichtet werden müssen.

Training		Woche 1					Woche 2					Woche 3					Woche 4					Woche 5					Woche 6					Woche 7					Woche 8					Woche 9					Woche 10					Woche 11					Woche 12				
		Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr
		0	0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	0	0	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	0	0	0	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	0
1	v				DD		U	U	U	A	Ve	PA	PA	PP	PP		PP	G	G	G				PG	T	H1	H2	B	B	B		B	VV		S3	PT2	S1	S1				S2	PU	PU	PU																	
	n				DD		U	U	U	A		PA	PA	PP	PP		PP	PT1		G					T		H2	B	B	B		B	VV		S3		S1	S1				S2	PU	PU	PU																	

Eine weitere Nebenbedingung, die bei sehr vielen Trainings verletzt wird, ist die, dass der Unterricht 33 spätestens nach dem ersten Unterrichtstag von 12-32 stattfinden muss. Vernachlässigt man diese Einschränkung anstelle der Anpassung, dass die Unterrichte 12-37 an maximal 16 zusammenhängenden Tagen geplant werden müssen, kann ein optimaler Stundenplan für ein Training in 503,00 Sekunden erstellt werden. Die Abbildung 6.4 stellt eine Variante des Stundenplans vor.

Abbildung 6.4: Stundenplan für ein Training unter Vernachlässigung der Nebenbedingung, dass der Unterricht 33 spätestens nach dem ersten Unterrichtstag von 12-32 stattfinden muss.

Training		Woche 1					Woche 2					Woche 3					Woche 4					Woche 5					Woche 6					Woche 7					Woche 8					Woche 9					Woche 10					Woche 11					Woche 12				
		Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr
		0	0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	0	0	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	0	0	0	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	0
1	v						U	U	U	A	Ve	PA	PA	PP	PP		PP			DD							B	B	B	B	PT2	VV	S3	S1	S1	H1	H2	S2				T	PU	PU	PU					G	G													
	n						U	U	U	A		PA	PA	PP	PP		PP	PT1		DD							B	B	B	B		VV	S3	S1	S1		H2	S2				T	PU	PU	PU					G	G			PG								

Vernachlässigt man die Einschränkung für den Unterricht 33 und passt zusätzlich die genannte Nebenbedingung auf 18 zusammenhängende Tage an, ist ein optimaler Stundenplan in 114,40 Sekunden erstellt. Die folgende Abbildung 6.5 zeigt eine Möglichkeit des Plans.

Abbildung 6.5: Stundenplan für ein Training unter Vernachlässigung der Nebenbedingung, dass der Unterricht 33 spätestens nach dem ersten Unterrichtstag von 12-32 stattfinden muss und die Unterrichte 12-37 an maximal 18 zusammenhängenden Tagen unterrichtet werden müssen.

Training		Woche 1					Woche 2					Woche 3					Woche 4					Woche 5					Woche 6					Woche 7					Woche 8					Woche 9					Woche 10					Woche 11					Woche 12				
		Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr
		0	0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	0	0	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	0	0	0	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	0
1	v						U	U	U	A	Ve	PA	PA	PP	PP	G	DD	G	U	PG					B		B	B	B	B	H1	T	H2	VV	S1	PT2	S3	S1				S2	PU	PU	PU					G	G	G	A	PA	PA		PP	PP	PP		
	n						U	U	U	A		PA	PA	PP	G		DD	G	U	DD					B		B	B	B	B		T	H2	VV	S1		S3	S1				S2	PU	PU	PU					G		G	A	PA	PA		PP	PP	PP	PT1	

Der MIP-Solver CPLEX findet berits in 1.255,00 Sekunden 50 optimale Stundenpläne für ein Training mit der Anpassung, dass die Unterrichte 12-37 an maximal 18 zusammenhängenden Tagen unterrichtet werden müssen. Durch die Ausgaben von Stundenplanvarianten für ein Training kann dem Planer aufgezeigt werden, welche Möglichkeiten er hat, die Unterrichtseinheiten anzuordnen. Das erleichtert die zukünftige Planung insofern, dass im Anschluss lediglich trainingsübergreifende Bedingungen berücksichtigt werden müssen.

6.3 Stundenplan für mehrere Trainings

Mit dem erstellten Algorithmus ist es möglich, einen zulässigen Stundenplan (siehe Abbildung 6.6 mit 12.830 Variablen und 586.237 Nebenbedingungen für 4 Trainings in 9.477,50 Sekunden zu erstellen. Hierbei wurde die Nebenbedingung, dass die Unterrichte 12-37 an maximal 16 zusammenhängenden Tagen unterrichtet werden müssen, angepasst, sodass die Unterrichte 12-37 an 23 zusammenhängenden Tagen geplant werden. Außerdem wurde vernachlässigt, dass der Unterricht 33 spätestens nach dem ersten Unterrichtstag von 12-32 stattfinden muss. Auch die trainingsübergreifenden Einschränkungen wurden nicht berücksichtigt.

Abbildung 6.6: Stundenplan für 4 Trainings mit angepasster Nebenbedingung, dass die Unterrichte 12-37 an 23 zusammenhängenden Tagen geplant werden müssen, unter Vernachlässigung der Einschränkung, dass der Unterricht 33 spätestens nach dem ersten Unterrichtstag von 12-32 stattfinden muss und ohne Berücksichtigung trainingsübergreifender Bedingungen.

Training		Woche 1					Woche 2					Woche 3					Woche 4					Woche 5					Woche 6					Woche 7					Woche 8					Woche 9					Woche 10					Woche 11					Woche 12				
		Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr	Mo	Di	Mi	Do	Fr
		0	0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	0	0	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	0	0	0	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	0
1	v						U	U		U			A	Ve	PA		PA	PP	PP	PP	PT1			VV	B		B	B	B	PT2	H1	H2	T		S1		S3	S2				S1	PU	PU	PU					G	G	G			PG			DD				
	n						U	U		U			A		PA		PA	PP	PP	PP				VV	B		B	B	B			H2	T		S1		S3	S2				S1	PU	PU	PU							G						DD				
2	v				B		B	B	B			H2	VV	PT2	T		S3	S1	S2	S1				PU	PU		PU							U			U	DD					U		G		G			A	Ve	PA	PA	PP	PP		PP				
	n				B		B	B	B	H1		H2	VV		T		S3	S1	S2	S1				PU	PU		PU							U			U	DD					U	G			G		PG	A		PA	PA	PP	PP		PP	PT1			
3	v						U			DD		U			U					A	Ve			PA	PA		PP	PP	PP							VV	T					H1	G	H2	PG	B	B	B	B		S3		S2	S1		S1	PU	PU	PU	
	n						U			DD		U			U					A				PA	PA		PP	PP	PP	PT1						VV	T				G	G	G	H2		B	B	B	B		S3	PT2	S2	S1		S1	PU	PU	PU	
4	v							U	U	U		DD	B	B	B		B	T		VV				PT2	S1	H1	H2	S3	S1	S2		PU		PU	PU									G	G						A	PA		PA	Ve	PP	PP	PP	PT1	
	n							U	U	U		DD	B	B	B		B	T		VV					S1		H2	S3	S1	S2		PU		PU	PU								G	G			PG					A	PA		PA		PP	PP	PP		

Bereits für diesen Plan ist die Rechenlaufzeit sehr hoch. Aus dem Grund wurde auf das Erstellen eines Stundenplans für mehr als 4 Trainings unter Berücksichtigung aller Nebenbedingungen verzichtet.

7 Fazit

Das Ziel, einen zulässigen Stundenplan für die Unteroffiziersschule der Bundeswehr mittels eines Algorithmus der ganzzahligen Programmierung zu erstellen, wurde nicht erreicht.

Die vorgestellte Rechenvorschrift scheitert an der Laufzeit des MIP-Solvers. Wie bereits aufgearbeitet, ist es möglich, sich Lösungen für ein Training in angemessener Zeit ausgeben zu lassen. Ein daraus resultierender neuer Lösungsansatz wäre, ein zweites Verfahren aufzustellen, das die möglichen Pläne für ein Training in einem Stundenplan für 26 Trainings zusammensetzt. Das heißt, dass man das Problem in zwei Teilprobleme splittet und separat löst.

In dem Paper „University course timetabling with Moses: System demonstration“ wurde der Algorithmus *Moses* vorgestellt, der an der TU Berlin zur Stundenplanerstellung genutzt wird. Die Voraussetzungen wurden ausschließlich als weiche Nebenbedingungen implementiert. Das bietet den Vorteil, dass der Planer im Anschluss sieht, an welcher Bedingung der Algorithmus scheitert. Eine zweite Herangehensweise wäre also, die Rechenvorschrift so anzupassen, dass alle Einschränkungen als weiche Nebenbedingungen formuliert werden. Durch die resultierenden Ergebnisse aus dem Verfahren können anschließend die Vorgaben der Bundeswehr dementsprechend angepasst werden.

Das Kapitel 6.1 zeigt, dass es der Bundeswehr bis jetzt nicht möglich war, einen Stundenplan, der alle Einschränkungen des Lehrgangs 0001 berücksichtigt, aufzustellen. Man kann also schlussfolgern, dass es bisher keinen Plan gibt, der alle Bedingungen erfüllt.
Eine Kombination aus der Gliederung des Problems in Teilprobleme und das Überarbeiten der Bedingungen sowie die Möglichkeit des Umformulierens der Einschränkung in weiche Nebenbedingungen ermöglicht einen optimistischen Blick

in die Zukunft der Lösung des Problems mit Hilfe der ganzzahligen Programmierung.
Alles in allem bietet das vorgestellte Verfahren der ganzzahligen Programmierung zum Erstellen eines zulässigen Stundenplans für die Unteroffiziersschule der Bundeswehr zwar keine Lösung des Problems, aber eine vielversprechende Grundlage, auf die in unterschiedlicher Weise aufgebaut werden kann.

A Implementierung

In diesem Kapitel wird der Aufbau des Programms für die Erstellung eines möglichen Stundenplans beschrieben. Zu beachten ist, dass das Programm nur den Lehrgang 0001 der Militärschule plant. Alle anderen Lehrgänge werden nicht berücksichtigt. Im Folgenden wird die Implementierung der Bedingungen, die im Kapitel 4.3.1 aufgearbeiteten wurden, vorgestellt. Das Modell wurde in der Programmiersprache zimpl verfasst und anschließend mit dem MIP-Solver IBM ILOG CPLEX gelöst.

A.1 Mengen und Funktionen

Am Anfang des Programms werden im Bereich #SETS Mengen und Funktionen definiert.

```
#SETS

#Tage des PH
set T := { 1 to 51 };

#Unterteilung der Tage in Vormittag und Nachmittag
set Z := { "v", "n" };

#Zuteilung der Tage zu den Wochentagen
set Mo := { read "Montag.dat" as "<1n>" comment "#" };
set Di := { read "Dienstag.dat" as "<1n>" comment "#" };
set Mi := { read "Mittwoch.dat" as "<1n>" comment "#" };
set Do := { read "Donnerstag.dat" as "<1n>" comment "#" };
set Fr := { read "Freitag.dat" as "<1n>" comment "#" };

#Anzahl der Trainings (Klassen)
```

```
17 set K := { 1 to 26 };
18
19 #Menge der möglichen zu belegenden Zeiteinheiten
20 set SLOTS := { <k,t,z> in K*T*Z};
21
22 #zu planende Unterrichte
23 set U := { read "Kurse.dat" as "<1s>" comment "#" };
24
25 #Unterrichte, die nicht am Freitag stattfinden dürfen
26 set NichtFreitag := { "A", "PA", "PP", "DD" };
27
28 #Unterrichte, die am Freitag stattfinden sollen
29 set Freitag := { "PT2", "H1" };
30
31 #Unterrichte, die nur eine Unterrichtseinheit haben
32 set HT := { "Ve", "PT1", "PT2", "H1", "PG", "G2", "G3" };
33
34 set G := { "G1", "G2", "G3" };
35
36 #Unterrichte, die den ResTyp x-mal bereitstellen
37 set ResTyp_11:= { read "ResTyp_11.dat" as "<1s>" comment "#" };
38 set ResTyp_4 := { read "ResTyp_4.dat" as "<1s>" comment "#" };
39 set ResTyp_2 := { read "ResTyp_2.dat" as "<1s>" comment "#" };
40
41 #optional, um Zuteilung der Unterrichte zu testen
42 set Loesung :=  { read "out_neu.dat" as "<1n, 2n, 3s, 4s>"
     comment "#" };
43
44 #Funktion ordnet den Tagen und Zeiten aufsteigend einen Wert zu
45 defnumb slotindex(t,z) := 10 + 2* (t-1) + if z == "v" then 0
     else 1 end;
```

A.2 Parameter

Nachdem nun Mengen und Funktionen definiert sind, werden benötigte Parameter festgelegt. Dies passiert unter #PARAMETER.

```
47 #PARAMETER
48
```

```
49 #Anzahl der zu planenden Zeiteinheiten pro Unterrichtsblock
50 param d[U] := read "Kurse.dat" as "<1s> 2n" comment "#";
51
52 #gibt Alternative zurück, wenn es eine gibt
53 param alt[U] := read "Kurse.dat" as "<1s> 3s" comment "#";
54
55 #gibt die Anzahl des benötigten ResTyp für den jeweiligen
     Unterricht an
56 param ben[U] := read "ResTyp.dat" as "<1s> 2n" comment "#";
57
58 #parameterabhängige Mengen
59 #in der Menge ist der Unterricht X und XA enthalten bzw. nur X
     und X
60 set AltU[<u> in U] := { u, alt[u] };
```

A.3 Variablen

Neben den Parametern mit fest zu geordneten Werten werden nun Variablen eingeführt. Diese werden unter anderem für die Zielfunktion benötigt. Ihre Definition erfolgt unter #VARIABLEN.

```
65 #VARIABLEN
66
67 #Zuordnung Training, Tag, Zeit, Unterricht
68 var x[SLOTS*U] binary;
69
70 #Strafvariable für Unterricht an verschiedenen Tagen
71 var y1[K*T] binary;
72
73 #Strafvariable für Unterricht, der am Freitag stattfinden soll
74 var y2 real;
75
76 #Hilfsvariablen für Unterrichte mit Alternative
77 var v[K*U] binary;
78 var vh1[K] binary;
79 var vh2[K] binary;
80 var vh3[K] binary;
81 var vh4[K] binary;
82 var vh5[K] binary;
```

```
var vh6[K] binary;

#Hilfsvariablen für Unterrichte an zusammenhängenden Tagen
var z1[<k,t> in K*T with t <= card(T)-4] binary;
var z9[<k,t> in K*T with t <= card(T)-7] binary;
var z3[<k,t> in K*T with t <= card(T)-14] binary;
var z4[<k,t> in K*T with t <= card(T)-3] binary;
var z5[<k,t> in K*T with t <= card(T)-2] binary;
var z6[<k,t> in K*T with t <= card(T)-2] binary;
var z7[<k,t> in K*T with t <= card(T)-3] binary;
var z8[<k,t> in K*T with t <= card(T)-4] binary;

#Hilfsvariablen für die Bedingung, dass der ResTyp für PU an
  jedem Ort zur Verfügung steht
var w1[K*T*Z] binary;
var w2[K*T*Z] binary;
var w3[K*T*Z] binary;
```

A.4 Zielfunktion

Um ein Problem zu optimieren, wird eine Funktion benötigt, die das Problem beschreibt und anschließend durch Minimierung oder Maximierung zur optimalen Lösung dessen führt. Auch in der Stundenplanoptimierung definiert man eine Zielfunktion. Dies geschieht unter #OBJECTIVE. Hier werden die y-Variablen addiert, da diese die Aufgabe haben, das Nichteinhalten von Nebenbedingungen zu bestrafen. Ziel ist es, den kleinstmöglichen Zielfunktionswert zu erhalten, das heißt, dass über die Summe aller y minimiert wird.

```
#OBJECTIVE

#Summe der Strafvariablen wird minimiert
minimize soll: sum <k,t> in K*T do y1[k,t] + y2;
```

A.5 Nebenbedingungen

Schlussendlich müssen die Nebenbedingungen implementiert werden. Dies geschieht im Bereich #CONSTRAINTS mit Hilfe der bereits definierten Mengen, Funktionen, Parameter und Variablen. Hier können sowohl weiche als auch harte Nebenbedingungen formuliert werden. Die harten Nebenbedingungen schränken den Lösungsraum ein, die weichen Nebenbedingung erhöhen bei Nichteinhaltung den Zielfunktionswert.

```
105 #CONSTRAINTS
106
107 #alle Variablen der Lösungsmenge zum Testen des Plans werden
      auf 1 gesetzt
108 subto Loesungsmenge:
109       forall <k,t,z,u> in Loesung do x[k,t,z,u] == 1;
110
111 #die erste Hälfte der Kurse beginnt am 1. Tag des PH und endet
      am 48. Tag
112 subto erster_PH:
113       forall <k> in K with 2*k <= card(K) do
114       sum <t,z,u> in T*Z*U with 48<=t do x[k,t,z,u] == 0;
115
116 #die zweite Hälfte der Kurse beginnt am 3. Tag des PH und endet
       am 51. Tag
117 subto zweiter_PH:
118       forall <k> in K with card(K) < 2*k do
119       sum <t,z,u> in T*Z*U with t <= 3 do x[k,t,z,u] == 0;
120
121 #jede Zeiteinheit darf maximal von einem Unterricht belegt
      werden
122 subto max_Unterricht_pro_SLOT:
123       forall <k,t,z> in SLOTS do
124       sum <u> in U do x[k,t,z,u] <= 1;
125
126 #Unterrichte dürfen nicht am Freitagnachmittag liegen
127 subto Unterricht_nicht_am_Frn:
128       forall <k> in K do
129       sum <t,"n",u> in Fr*Z*U do x[k,t,"n",u] == 0;
130
131 #am 1. Tag des PH darf kein Unterricht U stattfinden
```

```
132 subto erster_Tag_kein_U_PH1:
133       forall <k,z> in K*Z do x[k,1,z,"U"] == 0;
134
135 subto erster_Tag_kein_U_PH2:
136       forall <k,z> in K*Z with card(K) < 2*k do
137       x[k,4,z,"U"] == 0;
138
139 #Unterricht muss in der vorgeschriebenen Anzahl verplant werden
140 subto Unterrichtsstunden_verplanen_mit_Alternative:
141       forall <k,u> in K*U do
142       sum <ualt> in AltU[u] do
143       sum <t,z> in T*Z do x[k,t,z,ualt] == d[u];
144
145 #Unterrichte, die nicht am Freitag stattfinden dürfen
146 subto Unterricht_nicht_am_Fr:
147       forall <k,t,u> in K*Fr*NichtFreitag do
148       x[k,t,"v",u] == 0;
149
150 #DD darf nicht am Mittwoch geplant werden
151 subto DD_nicht_am_Mi:
152       forall <k,t,z> in K*Mi*Z do x[k,t,z,"DD"] == 0;
153
154 #Unterricht, der am Freitag stattfinden sollte
155 subto Unterricht_sollte_am_Fr:
156       sum <k,t,"v",u> in K*Fr*Z*Freitag do
157       x[k,t,"v",u] + y2 == d[u];
158
159 #Unterrichte eines Typs, die am gleichen Tag unterrichtet
    werden sollen
160 subto Strafe_fuer_Unterricht_an_versch_Tagen_1:
161       forall <k,t,u> in K*T*(U-HT) do
162       (x[k,t,"v",u] - x[k,t,"n",u]) <= y1[k,t];
163
164 subto Strafe_fuer_Unterricht_an_versch_Tagen_2:
165       forall <k,t,u> in K*T*(U-HT) do
166       (-x[k,t,"v",u] + x[k,t,"n",u]) <= y1[k,t];
167
168 #Alternative Unterrichte -> es darf entweder nur der Unterricht
     X oder nur der Unterricht XA verplant werden
169 subto vB_oder_vBA:
170       forall <k> in K do (v[k,"B"] + v[k,"BA"]) == 1;
171 subto vB_Bed:
```

```
172       forall <k,t,z> in SLOTS do x[k,t,z,"B"] <= v[k,"B"];
173 subto vBA_Bed:
174       forall <k,t,z> in SLOTS do x[k,t,z,"BA"] <= v[k,"BA"];
175
176 subto vVV_oder_vVVA:
177       forall <k> in K do (v[k,"VV"] + v[k,"VVA"]) ==1;
178 subto vVV_Bed:
179       forall <k,t,z> in SLOTS do x[k,t,z,"VV"] <= v[k,"VV"];
180 subto vVVA_Bed:
181       forall <k,t,z> in SLOTS do x[k,t,z,"VVA"] <= v[k,"VVA"];
182
183 subto vS1_oder_vSA1:
184       forall <k> in K do (v[k,"S1"] + v[k,"SA1"]) == 1;
185 subto vS1_Bed:
186       forall <k,t,z> in SLOTS do x[k,t,z,"S1"] <= v[k,"S1"];
187 subto vSA1_Bed:
188       forall <k,t,z> in SLOTS do x[k,t,z,"SA1"] <= v[k,"SA1"];
189
190 subto vS2_oder_vSA2:
191       forall <k> in K do (v[k,"S2"] + v[k,"SA2"]) == 1;
192 subto vS2_Bed:
193       forall <k,t,z> in SLOTS do x[k,t,z,"S2"] <= v[k,"S2"];
194 subto vSA2_Bed:
195       forall <k,t,z> in SLOTS do x[k,t,z,"SA2"] <= v[k,"SA2"];
196
197 subto vS3_oder_vSA3:
198       forall <k> in K do (v[k,"S3"] + v[k,"SA3"]) == 1;
199 subto vS3_Bed:
200       forall <k,t,z> in SLOTS do x[k,t,z,"S3"] <= v[k,"S3"];
201 subto vSA3_Bed:
202       forall <k,t,z> in SLOTS do x[k,t,z,"SA3"] <= v[k,"SA3"];
203
204 subto vT_oder_vTA:
205       forall <k> in K do (v[k,"T"] + v[k,"TA"]) == 1;
206 subto vT_Bed:
207       forall <k,t,z> in SLOTS do x[k,t,z,"T"] <= v[k,"T"];
208 subto vTA_Bed:
209       forall <k,t,z> in SLOTS do x[k,t,z,"TA"] <= v[k,"TA"];
210
211 subto S1_und_S2_oder_SA1_und_SA2:
212       forall <k> in K do (vh1[k] + vh2[k]) == 2;
213 subto S1_oder_SA2:
```

```
214       forall <k> in K do (v[k,"S1"] + v[k,"SA2"]) <= vh1[k];
215 subto SA1_oder_S2:
216       forall <k> in K do (v[k,"SA1"] + v[k,"S2"]) <= vh2[k];
217
218 subto S1_und_S3_oder_SA1_und_SA3:
219       forall <k> in K do (vh3[k] + vh4[k]) == 2;
220 subto S1_oder_SA3:
221       forall <k> in K do (v[k,"S1"] + v[k,"SA3"]) <= vh3[k];
222 subto SA1_oder_S3:
223       forall <k> in K do (v[k,"SA1"] + v[k,"S3"]) <= vh4[k];
224
225 subto S3_und_S2_oder_SA3_und_SA2:
226       forall <k> in K do (vh5[k] + vh6[k]) == 2;
227 subto S3_oder_SA2:
228       forall <k> in K do (v[k,"S3"] + v[k,"SA2"]) <= vh5[k];
229 subto SA3_oder_S2:
230       forall <k> in K do (v[k,"SA3"] + v[k,"S2"]) <= vh6[k];
231
232 #Unterrichte vor anderen -> Unterricht X muss vor Unterricht Y
      geplant werden
233 subto U_vor_Ve:
234       forall <k,t,z> in SLOTS do
235       forall <t2,z2> in T*Z
236       with slotindex(t2,z2) <= slotindex(t,z) do
237       x[k,t2,z2,"Ve"] <= 1 - x[k,t,z,"U"];
238
239 subto U_vor_A:
240       forall <k,t,z> in SLOTS do
241       forall <t2,z2> in T*Z
242       with slotindex(t2,z2) <= slotindex(t,z) do
243       x[k,t2,z2,"A"] <= 1 - x[k,t,z,"U"];
244
245 subto U_vor_PA:
246       forall <k,t,z> in SLOTS do
247       forall <t2,z2> in T*Z
248       with slotindex(t2,z2) <= slotindex(t,z) do
249       x[k,t2,z2,"PA"] <= 1 - x[k,t,z,"U"];
250
251 subto U_vor_PP:
252       forall <k,t,z> in SLOTS do
253       forall <t2,z2> in T*Z
254       with slotindex(t2,z2) <= slotindex(t,z) do
```

```
255       x[k,t2,z2,"PP"] <= 1 - x[k,t,z,"U"];
256
257 subto U_vor_PT1:
258       forall <k,t,z> in SLOTS do
259       forall <t2,z2> in T*Z
260       with slotindex(t2,z2) <= slotindex(t,z) do
261       x[k,t2,z2,"PT1"] <= 1 - x[k,t,z,"U"];
262
263 subto A_vor_Ve:
264       forall <k,t,z> in SLOTS do
265       forall <t2,z2> in T*Z
266       with slotindex(t2,z2) <= slotindex(t,z) do
267       x[k,t2,z2,"Ve"] <= 1 - x[k,t,z,"A"];
268
269 subto A_vor_PA:
270       forall <k,t,z> in SLOTS do
271       forall <t2,z2> in T*Z
272       with slotindex(t2,z2) <= slotindex(t,z) do
273       x[k,t2,z2,"PA"] <= 1 - x[k,t,z,"A"];
274
275 subto A_vor_PT1:
276       forall <k,t,z> in SLOTS do
277       forall <t2,z2> in T*Z
278       with slotindex(t2,z2) <= slotindex(t,z) do
279       x[k,t2,z2,"PT1"] <= 1 - x[k,t,z,"A"];
280
281 subto Ve_vor_PT1:
282       forall <k,t,z> in SLOTS do
283       forall <t2,z2> in T*Z
284       with slotindex(t2,z2) <= slotindex(t,z) do
285       x[k,t2,z2,"PT1"] <= 1 - x[k,t,z,"Ve"];
286
287 subto PA_vor_PP:
288       forall <k,t,z> in SLOTS do
289       forall <t2,z2> in T*Z
290       with slotindex(t2,z2) <= slotindex(t,z) do
291       x[k,t2,z2,"PP"] <= 1 - x[k,t,z,"PA"];
292
293 subto PA_vor_PT1:
294       forall <k,t,z> in SLOTS do
295       forall <t2,z2> in T*Z
296       with slotindex(t2,z2) <= slotindex(t,z) do
```

```
297       x[k,t2,z2,"PT1"] <= 1 - x[k,t,z,"PA"];
298
299 subto PP_vor_PT1:
300       forall <k,t,z> in SLOTS do
301       forall <t2,z2> in T*Z
302       with slotindex(t2,z2) <= slotindex(t,z) do
303       x[k,t2,z2,"PT1"] <= 1 - x[k,t,z,"PP"];
304
305 subto B_vor_PT2:
306       forall <k,t,z> in SLOTS do
307       forall <t2,z2> in T*Z
308       with slotindex(t2,z2) <= slotindex(t,z) do
309       x[k,t2,z2,"PT2"] <= 1 - (x[k,t,z,"B"] + x[k,t,z,"BA"]);
310
311 subto VV_und_VVA_vor_S1_und_SA1:
312       forall <k,t,z> in SLOTS do
313       forall <t2,z2> in T*Z
314       with slotindex(t2,z2) <= slotindex(t,z) do
315       (x[k,t2,z2,"S1"] + x[k,t2,z2,"SA1"])
316       <= 1 - (x[k,t,z,"VV"] + x[k,t,z,"VVA"]);
317
318 subto VV_und_VVA_vor_S2_und_SA2:
319       forall <k,t,z> in SLOTS do
320       forall <t2,z2> in T*Z
321       with slotindex(t2,z2) <= slotindex(t,z) do
322       (x[k,t2,z2,"S2"] + x[k,t2,z2,"SA2"])
323       <= 1 - (x[k,t,z,"VV"] + x[k,t,z,"VVA"]);
324
325 subto VV_und_VVA_vor_S3_und_SA3:
326       forall <k,t,z> in SLOTS do
327       forall <t2,z2> in T*Z
328       with slotindex(t2,z2) <= slotindex(t,z) do
329       (x[k,t2,z2,"S3"] + x[k,t2,z2,"SA3"])
330       <= 1 - (x[k,t,z,"VV"] + x[k,t,z,"VVA"]);
331
332 subto B_und_BA_vor_PU:
333       forall <k,t,z> in SLOTS do
334       forall <t2,z2> in T*Z
335       with slotindex(t2,z2) <= slotindex(t,z) do
336       x[k,t2,z2,"PU"] <= 1 - (x[k,t,z,"B"] + x[k,t,z,"BA"]);
337
338 subto PT2_vor_PU:
```

```
339       forall <k,t,z> in SLOTS do
340       forall <t2,z2> in T*Z
341       with slotindex(t2,z2) <= slotindex(t,z) do
342       x[k,t2,z2,"PU"] <= 1 - x[k,t,z,"PT2"];
343
344 subto VV_und_VVA_vor_PU:
345       forall <k,t,z> in SLOTS do
346       forall <t2,z2> in T*Z
347        with slotindex(t2,z2) <= slotindex(t,z) do
348       x[k,t2,z2,"PU"] <= 1 - (x[k,t,z,"VV"] + x[k,t,z,"VVA"]);
349
350 subto S1_und_SA1_vor_PU:
351       forall <k,t,z> in SLOTS do
352       forall <t2,z2> in T*Z
353       with slotindex(t2,z2) <= slotindex(t,z) do
354       x[k,t2,z2,"PU"] <= 1 - (x[k,t,z,"S1"] + x[k,t,z,"SA1"]);
355
356 subto S2_und_SA2_vor_PU:
357       forall <k,t,z> in SLOTS do
358       forall <t2,z2> in T*Z
359       with slotindex(t2,z2) <= slotindex(t,z) do
360       x[k,t2,z2,"PU"] <= 1 - (x[k,t,z,"S2"] + x[k,t,z,"SA2"]);
361
362 subto S3_und_SA3_vor_PU:
363       forall <k,t,z> in SLOTS do
364       forall <t2,z2> in T*Z
365       with slotindex(t2,z2) <= slotindex(t,z) do
366       x[k,t2,z2,"PU"] <= 1 - (x[k,t,z,"S3"] + x[k,t,z,"SA3"]);
367
368 subto T_und_TA_vor_PU:
369       forall <k,t,z> in SLOTS do
370       forall <t2,z2> in T*Z
371       with slotindex(t2,z2) <= slotindex(t,z) do
372       x[k,t2,z2,"PU"] <= 1 - (x[k,t,z,"T"] + x[k,t,z,"TA"]);
373
374 subto H1_vor_H2:
375       forall <k,t,z> in SLOTS do
376       forall <t2,z2> in T*Z
377       with slotindex(t2,z2) <= slotindex(t,z) do
378       x[k,t2,z2,"H2"] <= 1 - x[k,t,z,"H1"];
379
380 subto G1_vor_PG:
```

```
381       forall <k,t,z> in SLOTS do
382       forall <t2,z2> in T*Z
383       with slotindex(t2,z2) <= slotindex(t,z) do
384       x[k,t2,z2,"PG"] <= 1 - x[k,t,z,"G1"];
385
386 subto G2_vor_PG:
387       forall <k,t,z> in SLOTS do
388       forall <t2,z2> in T*Z
389       with slotindex(t2,z2) <= slotindex(t,z) do
390       x[k,t2,z2,"PG"] <= 1 - x[k,t,z,"G2"];
391
392 subto G3_vor_PG:
393       forall <k,t,z> in SLOTS do
394       forall <t2,z2> in T*Z
395       with slotindex(t2,z2) <= slotindex(t,z) do
396       (x[k,t2,z2,"PG"]) <= 1 - x[k,t,z,"G3"];
397
398 #Unterricht X muss spätestens nach dem ersten Unterrichtstag
    von Y liegen
399 subto PP_1x_vor_Ve:
400       forall <k,t,z> in SLOTS do
401       sum <t2,z2> in T*Z
402       with slotindex(t2,z2) <= slotindex(t,z) do
403       x[k,t2,z2,"PP"] <= 2 + (1 - x[k,t,z,"Ve"])*(d["PP"]);
404
405 subto S1_S2_S3_1x_vor_PT2:
406       forall <k,t,z> in SLOTS do
407       sum <t2,z2> in T*Z
408       with slotindex(t2,z2) <= slotindex(t,z) do
409       (x[k,t2,z2,"S1"] + x[k,t2,z2,"S2"] + x[k,t2,z2,"S3"]
410       + x[k,t2,z2,"SA1"] + x[k,t2,z2,"SA2"] + x[k,t2,z2,"SA3"])
411       <= 2 + (1 - x[k,t,z,"PT2"])
412       *(d["S1"] + d["S2"] + d["S3"]);
413
414 subto B_PT2_VV_S1_S2_S3_V_1x_vor_H1:
415       forall <k,t,z> in SLOTS do
416       sum <t2,z2> in T*Z
417       with slotindex(t2,z2) <= slotindex(t,z) do
418       (x[k,t2,z2,"B"] + x[k,t2,z2,"BA"] + x[k,t2,z2,"PT2"]
419       + x[k,t2,z2,"VV"] + x[k,t2,z2,"VVA"]
420       + x[k,t2,z2,"S1"] + x[k,t2,z2,"SA1"]
421       + x[k,t2,z2,"S2"] + x[k,t2,z2,"SA2"]
```

```
422       + x[k,t2,z2,"S3"] + x[k,t2,z2,"SA3"]
423       + x[k,t2,z2,"T"] + x[k,t2,z2,"TA"])
424       <= 2 + (1 - x[k,t,z,"H1"])*(d["B"] + d["PT2"]
425       + d["VV"] + d["S1"] + d["S2"] + d["S3"] + d["T"]);
426
427 #einer der Unterrichte muss vor H1 geplant werden
428 subto H1_1x_nach_B_PT2_VV_S1_S2_S3_V:
429       forall <k,t,z> in SLOTS do
430       x[k,t,z,"H1"] <= sum <t2,z2> in T*Z
431       with slotindex(t2,z2) <= slotindex(t,z) do
432       (x[k,t2,z2,"B"] + x[k,t2,z2,"BA"] + x[k,t2,z2,"PT2"]
433       + x[k,t2,z2,"VV"] + x[k,t2,z2,"VVA"]
434       + x[k,t2,z2,"S1"] + x[k,t2,z2,"SA1"]
435       + x[k,t2,z2,"S2"] + x[k,t2,z2,"SA2"]
436       + x[k,t2,z2,"S3"] + x[k,t2,z2,"SA3"]
437       + x[k,t2,z2,"T"] + x[k,t2,z2,"TA"]);
438
439 #H2 muss spätestens nach dem 9. Unterrichtstag der Kurse
    stattfinden
440 subto B_PT2_VV_S1_S2_S3_V_9x_vor_H2:
441       forall <k,t,z> in SLOTS do
442       sum <t2,z2> in T*Z
443       with slotindex(t2,z2) <= slotindex(t,z) do
444       (x[k,t2,z2,"B"] + x[k,t2,z2,"BA"] + x[k,t2,z2,"PT2"]
445       + x[k,t2,z2,"VV"] + x[k,t2,z2,"VVA"]
446       + x[k,t2,z2,"S1"] + x[k,t2,z2,"SA1"]
447       + x[k,t2,z2,"S2"] + x[k,t2,z2,"SA2"]
448       + x[k,t2,z2,"S3"] + x[k,t2,z2,"SA3"]
449       + x[k,t2,z2,"T"] + x[k,t2,z2,"TA"])
450       <= 18 + (1 - x[k,t,z,"H2"])*(d["B"] + d["PT2"]
451       + d["VV"] + d["S1"] + d["S2"] + d["S3"] + d["T"]);
452
453 #zusammenhängende Unterrichtstage -> die Unterrichte müssen an
    x zusammenhängenden Tagen unterrichtet werden
454 subto S_zusammenhaengend:
455       forall <k,t> in K*T with t <= card(T)-4 do
456       (d["S1"] + d["S2"] + d["S3"])*z1[k,t]
457       <= sum <k,t2,z> in SLOTS with t <= t2 and t2 <= t+5 do
458       (x[k,t2,z,"S1"] + x[k,t2,z,"S2"] + x[k,t2,z,"S3"]
459       + x[k,t2,z,"SA1"] + x[k,t2,z,"SA2"] + x[k,t2,z,"SA3"]);
460 subto NB1:
461       forall <k> in K do sum <t> in T with t <= card(T)-4 do
```

```
462       z1[k,t] == 1;
463
464 subto A_PA_PP_PT1:
465       forall <k,t> in K*T with t <= card(T)-7 do
466       (d["A"]+d["PA"]+d["PP"]+d["PT1"])*z9[k,t]
467       <= sum <k,t2,z2> in SLOTS with t<=t2 and t2<=t+7 do
468       (x[k,t2,z2,"A"] + x[k,t2,z2,"PA"]
469       + x[k,t2,z2,"PP"] + x[k,t2,z2,"PT1"]);
470 subto NB2:
471       forall <k> in K do sum <t> in T with t <= card(T)-7 do
472       z9[k,t] == 1;
473
474 subto K:
475       forall <k,t> in K*T with t <= card(T)-14 do
476       (d["B"] + d["PT2"] + d["VV"] + d["S1"] + d["S2"]
477       + d["S3"] + d["T"] + d["H1"] + d["H2"] + d["PU"])
478       *z3[k,t]
479       <= sum <k,t2,z2> in SLOTS with t<=t2 and t2<=t+15 do
480       (x[k,t2,z2,"B"] + x[k,t2,z2,"BA"]
481       + x[k,t2,z2,"VV"] + x[k,t2,z2,"VVA"]
482       + x[k,t2,z2,"S1"] + x[k,t2,z2,"SA1"]
483       + x[k,t2,z2,"S2"] + x[k,t2,z2,"SA2"]
484       + x[k,t2,z2,"S3"] + x[k,t2,z2,"SA3"]
485       + x[k,t2,z2,"T"] + x[k,t2,z2,"TA"])
486       + x[k,t2,z2,"H1"] + x[k,t2,z2,"H2"]
487       + x[k,t2,z2,"PT2"] + x[k,t2,z2,"PU"]);
488 subto NB3:
489       forall <k> in K do sum <t> in T with t <= card(T)-14 do
490       z3[k,t] == 1;
491
492 subto B:
493       forall <k,t> in K*T with t <= card(T)-3 do
494       d["B"]*z4[k,t] <= sum <k,t2,z2> in SLOTS
495       with t<=t2 and t2<=t+4 do
496       (x[k,t2,z2,"B"] + x[k,t2,z2,"BA"]);
497 subto NB4:
498       forall <k> in K do sum <t> in T with t <= card(T)-3 do
499       z4[k,t] == 1;
500
501 subto PU:
502       forall <k,t> in K*T with t <= card(T)-2 do
503       d["PU"]*z5[k,t]
```

```
504       <= sum <k,t2,z2> in SLOTS with t<=t2 and t2<=t+3 do
505       x[k,t2,z2,"PU"];
506 subto NB5:
507       forall <k> in K do sum <t> in T with t <= card(T)-2 do
508       z5[k,t] == 1;
509
510 subto H:
511       forall <k,t> in K*T with t <= card(T)-2 do
512       (d["H1"] + d["H2"])*z6[k,t]
513       <= sum <k,t2,z2> in SLOTS with t<=t2 and t2<=t+3 do
514       (x[k,t2,z2,"H1"] + x[k,t2,z2,"H2"]);
515 subto NB6:
516       forall <k> in K do sum <t> in T with t <= card(T)-2 do
517       z6[k,t] == 1;
518
519 subto G:
520       forall <k,t> in K*T with t <= card(T)-3 do
521       (d["G1"] + d["G2"] + d["G3"])*z7[k,t]
522       <= sum <k,t2,z2> in SLOTS with t<=t2 and t2<=t+3 do
523       (x[k,t2,z2,"G1"] + x[k,t2,z2,"G2"] + x[k,t2,z2,"G3"]);
524 subto NB7:
525       forall <k> in K do sum <t> in T with t <= card(T)-3 do
526       z7[k,t] == 1;
527
528 subto G_PG:
529       forall <k,t> in K*T with t <= card(T)-4 do
530       (d["G1"] + d["G2"] + d["G3"] + d["PG"])*z8[k,t]
531       <= sum <k,t2,z> in SLOTS with t<=t2 and t2<=t+5 do
532       (x[k,t2,z,"G1"] + x[k,t2,z,"G2"]
533       + x[k,t2,z,"G3"] + x[k,t2,z,"PG"]);
534 subto NB8:
535       forall <k> in K do sum <t> in T with t <= card(T)-4 do
536       z8[k,t] == 1;
537
538 #wenn letztes G am Freitag, dann PG am Montag, sonst ein Tag
    Pause
539 subto Lernpause_NB:
540       forall <k,z,u> in K*Z*G do
541       sum <t> in Fr with t<=47 and 7<=t do
542       (x[k,t-3,z,u] + x[k,t-2,z,u] + x[k,t-1,z,u]
543       + x[k,t,z,u] + x[k,t+1,z,"PG"]) <= 4;
544
```

```
545 subto Lernpause_zw_G1_und_PG_nFr:
546       forall <k,t,z> in K*(T-Fr)*Z with t<=50 do
547       (x[k,t,z,"G1"] + x[k,t+1,z,"PG"]) <= 1;
548
549 subto Lernpause_zw_G2_und_PG_nFr:
550       forall <k,t> in K*(T-Fr) with t<=50 do
551       sum <z> in Z do (x[k,t,z,"G2"] + x[k,t+1,z,"PG"]) <= 1;
552
553 subto Lernpause_zw_G3_und_PG_nFr:
554       forall <k,t> in K*(T-Fr) with t<=50 do
555       sum <z> in Z do (x[k,t,z,"G3"] + x[k,t+1,z,"PG"]) <= 1;
556
557 #jeder Unterrichtstyp von G muss an einem anderen Tag
      stattfinden
558 subto G2_G3_nicht_an_einem_Tag_1:
559       forall <k,t> in K*T do
560       (x[k,t,"v","G2"] + x[k,t,"n","G3"]) <= 1;
561 subto G2_G3_nicht_an_einem_Tag_2:
562       forall <k,t> in K*T do
563       (x[k,t,"n","G2"] + x[k,t,"v","G3"]) <= 1;
564
565 subto G2_PG_nicht_an_einem_Tag_1:
566       forall <k,t> in K*T do
567       (x[k,t,"v","G2"] + x[k,t,"n","PG"]) <= 1;
568 subto G2_PG_nicht_an_einem_Tag_2:
569       forall <k,t> in K*T do
570       (x[k,t,"n","G2"] + x[k,t,"v","PG"]) <= 1;
571
572 subto G3_PG_nicht_an_einem_Tag_1:
573       forall <k,t> in K*T do
574       (x[k,t,"v","G3"] + x[k,t,"n","PG"]) <= 1;
575 subto G3_PG_nicht_an_einem_Tag_2:
576       forall <k,t> in K*T do
577       (x[k,t,"n","G3"] + x[k,t,"v","PG"]) <= 1;
578
579 #zeitgleich dürfen maximal x Unterrichte von X in allen
      Trainings liegen
580 subto max_3_U:
581       forall <t,z> in T*Z do
582       sum <k> in K do (x[k,t,z,"U"] + x[k,t,z,"Ve"]) <= 3;
583 subto max_2_A:
584       forall <t,z> in T*Z do
```

```
585       sum <k> in K do x[k,t,z,"A"] <= 2;
586 subto max_3_PP:
587       forall <t,z> in T*Z do
588       sum <k> in K do x[k,t,z,"PP"] <= 3;
589 subto max_5_A_PA_PP:
590       forall <t,z> in T*Z do
591       sum <k> in K do (x[k,t,z,"A"] + x[k,t,z,"PA"]
592       + x[k,t,z,"PP"]) <= 5;
593 subto max_3_G_PG:
594       forall <t,z> in T*Z do
595       sum <k> in K do
596       (x[k,t,z,"G1"] + x[k,t,z,"G2"] + x[k,t,z,"G3"] + x[k,t,z,
            "PG"]) <= 3;
597 subto max_2_DD:
598       forall <t,z> in T*Z do
599       sum <k> in K do x[k,t,z,"DD"] <= 2;
600
601 #der ResTyp steht an den Unterrichtsorten maximal x mal bereit
602 subto ResTyp_11:
603       forall <t,z> in T*Z do
604       sum <k,u> in K*ResTyp_11 do
605       (x[k,t,z,u]*ben[u] + w1[k,t,z]*x[k,t,z,"PU"]) <= 11;
606 subto ResTyp_4:
607       forall <t,z> in T*Z do
608       sum <k,u> in K*ResTyp_4 do
609       (x[k,t,z,u]*ben[u] + w2[k,t,z]*x[k,t,z,"PU"]) <= 4;
610 subto ResTyp_2:
611       forall <t,z> in T*Z do
612       sum <k,u> in K*ResTyp_2 do
613       (x[k,t,z,u]*ben[u] + w3[k,t,z]*x[k,t,z,"PU"]) <= 2;
614
615 #der ResTyp für PU steht an jedem Unterrichtsort bereit, darf
    aber nur an einem verbraucht werden
616 subto PU_ResTyp:
617       forall <k,t,z> in SLOTS do
618       (w1[k,t,z] + w2[k,t,z] + w3[k,t,z]) == 1;
```

A.6 Eingelesene Dateien

Montag.dat

```
#Montag
3
8
13
21
26
31
33
38
43
48
```

Dienstag.dat

```
#Dienstag
4
9
14
22
27
32
34
39
44
49
```

Mittwoch.dat

```
#Mittwoch
5
10
15
18
23
28
35
40
45
```

```
11 50
```

Donnerstag.dat

```
1  #Donnerstag
2  1
3  6
4  11
5  16
6  19
7  24
8  29
9  36
10 41
11 46
12 51
```

Freitag.dat

```
1  #Freitag
2  2
3  7
4  12
5  17
6  20
7  25
8  30
9  37
10 42
11 47
```

Kurse.dat

```
1  #Kurse          Dauer           Alternative
2  U               6               U
3  Ve              1               Ve
4  A               2               A
5  PA              4               PA
6  PP              6               PP
7  PT1             1               PT1
8  B               8               BA
9  BA              8               B
10 PT2             1               PT2
```

```
11 VV              2              VVA
12 VVA             2              VV
13 S1              4              SA1
14 S2              2              SA2
15 S3              2              SA3
16 SA1             4              S1
17 SA2             2              S2
18 SA3             2              S3
19 T               2              TA
20 TA              2              T
21 H1              1              H1
22 H2              2              H2
23 PU              6              PU
24 G1              2              G1
25 G2              1              G2
26 G3              1              G3
27 PG              1              PG
28 DD              2              DD
```

ResTyp.dat

```
1  #Unterricht, der          Anzahl der benötigten
2  #ResTyp benötigt          ResTypen
3  B
                               0
4  BA
                               0
5  VV
                               1
6  VVA
                               1
7  T
                               1
8  TA
                               1
9  PU
                               1
10 S1
                               2
11 S2
                               2
12 S3
                               2
```

```
13 SA1
                                    2
14 SA2
                                    2
15 SA3
                                    2
```

ResTyp_11.dat

```
1 #ResTyp_11
2 B
3 VV
4 T
5 S1
6 S2
7 S3
```

ResTyp_4.dat

```
1 #ResTyp_4
2 BA
3 VVA
4 SA1
5 SA3
6 TA
```

ResTyp_2.dat

```
1 #ResTyp_2
2 SA2
```

Literatur

[1] E. Burke, K. Jackson, J. H. Kingston und R. Weare. „Automated university timetabling: The state of the art“. In: *The computer journal* 40.9 (1997), S. 565–571 (siehe S. 10–13).

[2] M. W. Carter, G. Laporte und J. W. Chinneck. „A general examination scheduling system“. In: *Interfaces* 24.3 (1994), S. 109–120 (siehe S. 11).

[3] G. B. Dantzig. „Linear programming“. In: *Operations research* 50.1 (2002), S. 42–47 (siehe S. 9, 10).

[4] S. Daskalaki, T. Birbas und E. Housos. „An integer programming formulation for a case study in university timetabling“. In: *European Journal of Operational Research* 153.1 (2004), S. 117–135 (siehe S. 14, 16, 17, 19).

[5] W. Domschke, A. Drexl, R. Klein und A. Scholl. *Einführung in Operations Research*. Springer Berlin Heidelberg, 2015 (siehe S. 21–23, 26, 27, 29, 45, 59).

[6] T. Ellinger, G. Beuermann und R. Leisten. *Operations Research: Eine Einführung*. Springer Berlin Heidelberg, 2013 (siehe S. 21, 22, 25–27).

[7] A. Jaeger und G. Dantzig. *Lineare Programmierung und Erweiterungen*. Springer Berlin Heidelberg, 2013 (siehe S. 9, 10).

[8] S. N. Jat und S. Yang. „A memetic algorithm for the university course timetabling problem“. In: *Tools with Artificial Intelligence, 2008. ICTAI'08. 20th IEEE International Conference on*. Bd. 1. 2008, S. 427–433 (siehe S. 14).

[9] P. Kall. *Mathematische Methoden des Operations Research: Eine Einführung*. Vieweg+Teubner Verlag, 2013 (siehe S. 23).

[10] U. Kathöfer und U. Müller-Funk. *BWL-Crash-Kurs Operations Research*. UTB GmbH, 2008 (siehe S. 21–24, 28, 29).

[11] A. Koop und H. Moock. *Lineare Optimierung - eine anwendungsorientierte Einführung in Operations Research*. Spektrum Akademischer Verlag, 2007 (siehe S. 24).

[12] G. Lach, M. Lach, J. Schick und E. Zorn. „University course timetabling with Moses: System demonstration“. In: *Proceedings of the 11th International Confenference on Practice and Theory of Automated Timetabling (PATAT-2016)* (2016), S. 507–510 (siehe S. 19, 77).

[13] S. MirHassani. „A computational approach to enhancing course timetabling with integer programming“. In: *Applied Mathematics and Computation* 175.1 (2006), S. 814–822 (siehe S. 15, 16, 59).

[14] K. Neumann. *Operations Research Verfahren*. Hanser Verlag, 1975 (siehe S. 23–25, 27, 28).

[15] S. Nickel, O. Stein und K. Waldmann. *Operations Research*. Springer Berlin Heidelberg, 2014 (siehe S. 23–25, 27, 29).

[16] N. Pillay. „An overview of school timetabling research“. In: *Proceedings of the 8th International Conference on the Practice and Theory of Automated Timetabling (PATAT 2010)*. 2010, S. 321–335 (siehe S. 13).

[17] A. Schaerf. „A survey of automated timetabling“. In: *Artificial intelligence review* 13.2 (1999), S. 87–127 (siehe S. 30).

[18] K. Schimmelpfeng und S. Helber. „Application of a real-world university-course timetabling model solved by integer programming“. In: *Or Spectrum* 29.4 (2007), S. 783–803 (siehe S. 17, 18).

[19] H. Taha und J. Schmidt. *Integer Programming: Theory, Applications, and Computations*. Elsevier Science, 2014 (siehe S. 24).

[20] C. Valouxis und E. Housos. „Constraint programming approach for school timetabling“. In: *Computers & Operations Research* 30.10 (2003), S. 1555–1572 (siehe S. 14, 15).

[21] D. J. Welsh und M. B. Powell. „An upper bound for the chromatic number of a graph and its application to timetabling problems“. In: *The Computer Journal* 10.1 (1967), S. 85–86 (siehe S. 14).

Abbildungsverzeichnis

Tabellenverzeichnis

Selbstständigkeitserklärung

Der Verfasser erklärt, dass er die vorliegende Arbeit selbständig, ohne fremde Hilfe und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt hat. Die aus fremden Quellen (einschließlich elektronischer Quellen) direkt oder indirekt übernommenen Gedanken sind ausnahmslos als solche kenntlich gemacht. Wörtlich und inhaltlich verwendete Quellen wurden entsprechend den anerkannten Regeln wissenschaftlichen Arbeitens zitiert. Die Arbeit ist in gleicher oder vergleichbarer Form, auch nicht auszugsweise im Rahmen einer anderen Prüfung bei einer anderen Hochschule vorgelegt worden.

Sie wurde bisher auch nicht veröffentlicht.

Ich erkläre mich damit einverstanden, dass die Arbeit mit Hilfe eines Plagiatserkennungsdienstes auf enthaltene Plagiate überprüft wird.

Tabea Werger

München, 26. November 2018

IMPRESSUM

Brandenburgische Technische Universität Cottbus-Senftenberg
Fakultät 1 | MINT - Mathematik, Informatik, Physik, Elektro- und Informationstechnik
Institut für Mathematik
Platz der Deutschen Einheit 1
D-03046 Cottbus

Professur für Ingenieurmathematik und Numerik der Optimierung
Professor Dr. rer. nat. Armin Fügenschuh

E fuegenschuh@b-tu.de
T +49 (0)355 69 3127
F +49 (0)355 69 2307

Cottbus Mathematical Preprints (COMP), ISSN (Print) 2627-4019
Cottbus Mathematical Preprints (COMP), ISSN (Online) 2627-6100

www.b-tu.de/cottbus-mathematical-preprints
cottbus-mathematical-preprints@b-tu.de
doi.org/10.26127/btuopen-4803