Ein neuer Algorithmus zur Lösung des Raumplanungsproblems an Universitäten

Gerald Lach, Erhard Zorn
lach@math.tu-berlin.de, erhard@math.tu-berlin.de

Abstract: Die zeitweise Bereitstellung von Ressourcen, die nicht verbraucht werden und nur in beschränkter Menge zur Verfügung stehen, führt zu einem Verteilungsproblem, das allgemein als Timetabling Problem bekannt ist. Die Aufgabe besteht darin, eine Verteilung der Ressourcen zu finden, die alle Wünsche "möglichst gut" befriedigt. Die Güte einer Lösung wird etwa an der Zufriedenheit der Nutzer mit der gefundenen Lösung oder an den (realen) Kosten für die Lösung gemessen. Dabei kann z.B. die Zufriedenheit der Nutzer in abstrakten "Kosten" gemessen werden. Es handelt sich also um ein Optimierungsproblem, das mit Methoden der diskreten Optimierung behandelt werden muss, da die auftretenden Größen nur diskrete Werte annehmen können. Wir stellen einen Algorithmus zur Lösung des Raumplanungsproblems vor, mit dem Lösungen für Instanzen gefunden werden können, die an großen Universitäten wie der TU Berlin (ca. 30.000 Studierende) auftreten. Die Größe der Instanzen, die mit unserem Algorithmus gelöst werden können, liegt um einen Faktor 10 über der Größenordnung, die mit bisher bekannten Algorithmen gelöst werden können, wobei die benötigte Rechenzeit mit unter 2 Stunden auf einem handelsüblichen aktuellen PC vollkommen ausreichend ist.

1 Einleitung

Die zeitweise Bereitstellung von Ressourcen, die nicht verbraucht werden und nur in beschränkter Menge zur Verfügung stehen, führt zu einem Verteilungsproblem, das allgemein als Timetabling Problem bekannt ist. Die Aufgabe besteht darin, eine Verteilung der Ressourcen zu finden, die alle Wünsche "möglichst gut" befriedigt. Die Güte einer Lösung wird etwa an der Zufriedenheit der Nutzer mit der gefundenen Lösung oder an den (realen) Kosten für die Lösung gemessen. Dabei kann z.B. die Zufriedenheit der Nutzer in abstrakten "Kosten" gemessen werden. Es handelt sich also um ein Optimierungsproblem, das mit Methoden der diskreten Optimierung behandelt werden muss, da die auftretenden Größen nur diskrete Werte annehmen können.

Abhängig von den konkreten Nebenbedingungen entstehen verschiedene Varianten des Timetabling Problems. Aus mathematischer Sicht handelt es sich um ein "schweres" Problem, da der Rechenaufwand nicht nach einem einfachen Gesetz (d.h. polynomial) mit der Größe des Problems (Anzahl der Kurse) wächst. Probleme dieser Art heißen NP-schwer (für Details von NP-schweren Problemen siehe (GJ78), zur Komplexität des Stundenplanproblems siehe (uJHK95)).

Im universitären Bereich treten - abhängig von den Nebenbedingungen - folgende wichtige Varianten des Timetabling Problems auf:

- **Curriculum based course timetabling (Raumplanungsproblem):** Die in einem Semester anzubietenden Veranstaltungen müssen so auf die zur Verfügung stehenden Räume verteilt werden, dass Studierende alle Veranstaltungen besuchen können, die sie laut ihrem Studienplan besuchen müssen. Die Termine jedes Dozenten müssen ebenfalls überschneidungsfrei stattfinden. Als weitere Bedingung muss die Raumgröße für die zu erwartende Teilnehmerzahl ausreichen; in Abhängigkeit von der Veranstaltung können weitere Anforderungen an die Raumausstattung gestellt werden. Es handelt sich hierbei um das generelle Problem zur Erstellung eines Vorlesungsplans an einer Universität.

- **Post enrollment based course timetabling (Tutorieneinteilung):** Zu einzelnen Veranstaltungen, z.B. große Vorlesungen mit mehreren Hundert Studierenden, finden kleine Übungsgruppen (Tutorien) statt, auf die alle Teilnehmer der Veranstaltung überschneidungsfrei verteilt werden sollen. Dieses Problem ist dem 1. Problem nachgeordnet, weil sich die Studierenden zunächst zur Teilnahme

an einer Veranstaltung entscheiden und anschließend auf die kleinen Übungsgruppen verteilt werden. Die Tutorien zu einer Veranstaltung können nicht zu einem Termin stattfinden, sondern müssen im Allgemeinen über die gesamte Woche verteilt werden, da weder kleine Räume noch Personal zur Verfügung stehen, um alle Tutorien einer Veranstaltung gleichzeitig stattfinden zu lassen. Daher kann diese Aufgabe nicht zusammen mit dem 1. Problem gelöst werden.

- **Examination timetabling (Klausurterminplanung):** Bei diesem Problem müssen Termine für Klausuren bestimmt werden, sodass Studierende an allen Klausuren teilnehmen können, die sie laut ihrem Studienplan absolvieren müssen. Als weitere Nebenbedingung soll zwischen verschiedenen Klausuren, die von den gleichen Studierenden geschrieben werden, ein ausreichender Abstand liegen. Im Gegensatz zum 1. Problem werden hierbei im Allgemeinen keine Anforderungen an die Raumausstattung zu berücksichtigen sein. Dafür sind einer Klausur im Allgemeinen mehrere Räume zuzuordnen, die die erforderlichen Sitzplätze bieten.

Alle aufgeführten Probleme sind für das Raummanagement einer Universität, für die Studierbarkeit von Studiengängen und für die Zufriedenheit der Studierenden und Lehrenden sehr wichtig. Sie sind mit der Optimierung der Raumnutzung und der Personalressourcen (etwa durch gute Auslastung von Tutorien) und damit mit der Optimierung "realer" Kosten verbunden. Eine gute Auslastung der zur Verfügung stehenden Räume ist für die Planung neuer Räume bzw. für die Anmietung von Räumen wichtig.

Die Einführung der Bachelor- und Masterstudiengänge an deutschen Universitäten führt zu einem verstärkten Bewusstsein für diese Probleme: Wegen der Vielzahl der neuen Veranstaltungen können nicht - wie oftmals bisher - über lange Zeit angepasste und bewährte Stundenpläne verwendet werden. Zusätzlich ist der Unterricht in einigen Bereichen verschulter als bisher; die Studierenden sind durch Anwesenheitspflicht und Abhängigkeiten der Module mehr auf eine Studierbarkeit der Studienpläne angewiesen. Bei der Akkreditierung neuer Studiengänge wird auch die Studierbarkeit bewertet.

Seit 2003 setzen wir für die Tutorieneinteilung an der TU Berlin einen Algorithmus ein, der in unserer Arbeitsgruppe entwickelt wurde (vgl. (JLPZ07), (JLP$^+$06)).

Zum Klausurenproblem wurde im Rahmen einer Diplomarbeit an der TU Berlin eine Lösung entwickelt, deren Implementierung z.Z. umgesetzt wird, um das Problem für realer Daten zu lösen (vgl. (Lac08))).

In diesem Artikel betrachten wir das erste Problem, das Raumplanungsproblem. Der dargestellte neue Algorithmus wurde im Rahmen seiner Diplomarbeit von einem der Autoren (G.L.) entwickelt. Zu den mathematischen Details vergleich (LL07).

Der Artikel ist folgendermaßen aufgebaut: In Abschnitt 2 wird das Problem detailliert beschrieben, und wir geben einen Überblick über die bisher bekannten Lösungen. In Abschnitt 3 stellen wir unseren Lösungsansatz vor. An Hand eines einfachen Beispiels beschreiben wir die Modellierung des Problems, woraus ersichtlich wird, mit welchen Methoden es lösbar ist. In Abschnitt 4 stellen wir dar, mit welchem Rechenaufwand wir Probleme von der Größenordnung, wie sie an Universitäten wie der TU Berlin auftreten, lösen können, und vergleichen unsere Ergebnisse mit anderen bekannten Lösungen.

2 Das Problem und aktuelle Lösungsmethoden

2.1 Einführung in das Problem

Selbst für das *curriculum based course timetabling* Problem sind mehrere Versionen bekannt. Die genaue Formulierung des Problems hängt oft von universitätsspezifischen Anforderungen an das allgemeine Stundenplanproblem ab. Die Nebenbedingungen können aber typischerweise in zwei Gruppen unterteilt werden, *weiche* und *harte Nebenbedingungen*. Harte Nebenbedingungen können als Bedingungen betrachtet werden, deren Beachtung einen realisierbaren Stundenplan garantiert. Weiche Nebenbedingungen sind Bedingungen, deren Verletzung erlaubt, aber bestraft wird. Diese Nebenbedingungen werden also so interpretiert, dass ihre Erfüllung zwar wünschenswert ist, sie jedoch nicht zwingend für das universitäre Stundenplanproblem beachtet werden müssen. Unser Ansatz konzentriert sich daher auf *harte Nebenbedingungen* und vernachlässigt einige *weiche Nebenbedingungen*. Mit dieser Idee sind wir in der Lage,

große Instanzen (größer als alle Instanzen, die in der Literatur zu finden sind) dieses Problemtyps zu lösen.

Im Rahmen der PATAT08, einer Konferenz, deren Schwerpunkt ausschließlich auf verschiedenen Stundenplanproblemen liegt, wurde 2006 eine genaue Formulierung des *curriculum based course timetabling* Problems definiert, vgl. (ITC), (DGMS07). Nach dieser Charakterisierung gibt es vier verschiedene Typen harter Nebenbedingungen:

- Alle Vorlesungen einer Veranstaltung müssen festgelegt werden.
- Ein Dozent kann nur eine Vorlesung zu gleichen Zeit halten.
- Die verschiedenen Vorlesungen eines Curriculums dürften nicht gleichzeitig stattfinden.
- Ein Raum kann nicht zweimal zur gleichen Zeit belegt werden.

Und es gibt vier Arten weicher Nebenbedingungen:

- Für alle Kursen müssen die Räume mit der kurseigenen Ausstattung ausgestattet sein.
- Die Vorlesungen der Veranstaltungen müssen auf eine gegebene Anzahl an Tagen aufgeteilt werden.
- Die Vorlesungen eines Curriculums müssen zu nacheinander liegenden Zeitintervallen stattfinden.
- Die Vorlesungen eines Veranstaltung sollten immer im gleichen Raum stattfinden.

Wir wenden uns nun einer leicht unterschiedlichen Interpretation des Problems zu. Unsere Interpretation ist mehr an die Bedürfnisse einer deutschen Universität angepasst. Alle harten Nebenebedingungen der PATAT08 Formulierung werden als harte Bedingungen betrachtet. Zusätzlich behandeln wir die erste weiche Nebenbedingung als hart. Alle anderen weichen Nebenbedingungen werden außer Acht gelassen. Damit haben alle Kursanbieter die Möglichkeit, jede Kurszeit and jeden möglichen Raum mit einer Priorität zu versehen. Die Zeit-Priortät spiegelt den Wunsch des Anbieters wider, eine Vorlesung möglichst zu einer bestimmten Zeit durchzuführen, und die Raum-Priorität den Wunsch, die Vorlesung in einem bestimmten Raum stattfinden zu lassen.
Als weiteren grundsätzlichen Unterschied zwischen dem PATAT08 Modell und unserem Modell nehmen wir nicht an, dass sich die (für alle Kurse) möglichen Zeiten nicht überschneiden. Wir lassen also Zeiträume zu, die zu sich überschneidenden Veranstaltungszeiten gehören. Diese Annahme erschwert eine Lösung, ist jedoch unbedingt notwendig, wenn das Modell auf deutsche Universitäten angewandt werden soll, da dies insbesondere an Universitäten mit weit auseinander liegenden Campi (z.B. HU Berlin, Universität Stuttagart) unbedingt notwendig ist.

2.2 Andere Lösungsmethoden

In der Literatur sind verschiedene Lösungsmethoden verschiedener spezieller Problemtypen bekannt, die meisten sind heuristischer Natur, vgl. (BJKW97), (Lew07), (MO07). Weiter gibt es Methoden, die auf der ganzzahligen Programmierung basieren (DB05), (QS05), (SH07), (RFL06). Von beiden Lösungemethoden sind bereits einige im praktischen Einsatz und werden eingesetzt, um für kleine Universitäten oder Teile von Universitäten die Stundenplanung durchzuführen (BHW06). Jedoch ist es mit keiner Methode möglich, das Problem für reale Daten einer großen Universität zu lösen. Unsere Methode wurde erfolgreich für einige gebräuchliche kleine Datensätze von PATAT08 und für simulierte Daten der Größe der TU Berlin getestet. Reale Daten der TU Berlin konnten bisher nicht getestet werden, da die Daten noch nicht in geeigneter Form zur Verfügung gestellt werden konnten.

3 Unsere Lösungsmethode

Das Ziel unserer Lösungsmethode besteht darin, einen Stundenplan zu generieren, bei dem allen Kursen konfliktfrei Termine zugewiesen werden, also alle harten Nebenbedingungen erfüllt sind, und bei dem zusätzlich die weichen Nebenbedingungen so gut wie möglich befriedigt werden. Für reale Daten kann man im Normalfall nicht gewährleisten, dass eine Lösung existiert, die keine harte Nebenbedingungen verletzt. Es gibt fast immer Kurse, die in einem unlösbaren Konflikt zueinander stehen. Unter Berücksichtigung dieser Umstände, mussten wir das Ziel unserer Lösungsmethode neu definieren. Das Hauptziel muss sein, so viele Kurstermine wie möglich konfliktfrei zuzuweisen. Für diese Anzahl an konfliktfrei zugewiesenen Kursterminen, versuchen wir eine Lösung zu finden, die die Wünsche der Kursanbieter bestmöglich befriedigt. In den nun folgenden Abschnitten werden wir eine detaillierte Beschreibung, der Ein- und Ausgabeinformationen angeben, die von unserer Lösungsmethode benötigt bzw. erstellt werden.

3.1 Eingabedaten

Die Eingabedaten bestehen aus drei wesentlichen Objekten: *die Räume, die Kurse und die Curricula*. Typischerweise können die Räume durch eine beliebig große Anzahl an Attributen beschrieben werden. Dieses könnten z.B sein: Kapazität, Lage, vorhandener Beamer, Tafel. Die Attribute unterscheiden wir in zwei Typen, exklusive und inklusive. Exklusive Attribute können von einem Kurs nicht gleichzeitig in Anspruch genommen werden (z.B. verschiedene Raumkapazitäten). Wenn ein Attribut nicht exklusiv ist, so ist es inklusiv (z.B. Tafel und Beamer). Diese Unterscheidung spielt eine fundamentale Rolle in der Lösungsmethode und muss stets korrekt erfasst worden sein. Aufbauend auf dieser Beschreibung der Räume, haben die Kurse die Möglichkeit, die Attribute auszuwählen, die für ihre Durchführung unerlässlich sind. Anschließend kommen alle Räume als potentieller Veranstaltungsraum dieses Kurses in Frage, die über alle angeforderten Attribute verfügen. Für eine ausführlichere Beschreibung dieser Problemstellung siehe (LL07).

Desweiteren beinhalten die Daten der Kurse alle grundlegenden Informationen, wie z.B. den Dozenten, die Anzahl der Kurstermine, die möglichen Kurstermine, die benötigten Raumattribute, etc. Zudem besteht die Möglichkeit, dass die Kursanbieter die möglichen Kurstermine mit einer Priorität belegen. Eine Lösung des Problems versucht nun, die Prioritäten so gut wie möglich zu berücksichtigen. In der Definition der Curricula ist eindeutig festgelegt, in welchem Semester welche Studierenden welchen Kurs belegen müssen. Aufbauend auf diesen Informationen, muss die Lösungsmethode einen konfliktfreien Stundenplan konstruieren.

3.2 Ausgabedaten

Die Ausgabe der Lösungsmethode kann in zwei verschiedene Teile aufgeteilt werden. Der eine Teil besteht aus den konfliktfrei zugewiesenen Veranstaltungsterminen der Kurse zusammen mit den Räumen, wo diese stattfinden. Der andere Teil stellt die nicht zugewiesene Termine der Kurse zusammen mit einer Analyse dar. Die Analyse gibt die Gründe an, warum eine komplette Zuweisung aller Kurstermine nicht möglich war.

3.3 Idee der Lösungsmethode

Wenn wir die verschiedenen harten Nebenbedingungen genauer betrachten, stellen wir fest, dass man diese ebenfalls wieder in zwei Arten unterteilen kann, *die Zeitkonflikt- und Raumkonflikt-Bedingungen*. Die *Raumkonflikt-Bedingungen* können beschrieben werden als die Bedingungen, die berücksichtigen, dass die Ressource „Raum “beschränkt ist, z.B. fordert man, dass zu keinem Zeitpunkt ein Raum doppelt belegt werden darf. Alle anderen harten Nebenbedingungen sind die *Zeitkonflikt-Bedingungen*.

Diese beachten, dass die Ressource „Zeit “limitiert ist, z.B. kann ein Dozent nur ein Kurs zur selben Zeit halten.
Die Idee hinter unser Lösungsmethode ist, den Lösungsprozess in zwei von einander unabhängige Teile zu trennen. In einem ersten Schritt weisst man jedem Kurs seine Veranstaltungstermine zu. In dem folgenden zweiten Schritt, wird jedem Verantstaltungstermin ein Raum zugeordnet, in dem dieser stattfindet. Wir lösen also zuerst die *Zeitkonflikt-* und dann die *Raumkonflikt-Bedingungen* auf.
Allerdings muss man beachten, dass diese Dekomposition des Problems in zwei unabhängige Teile nicht einfach zu bewerkstelligen ist. Es könnte sein, dass Kurse in einer Raumkonflikt-Beziehung stehen, ohne in einer Zeitkonflikt-Beziehung zu sein. Somit muss man einen Weg finden, diese zu berücksichtigen, ohne sie explizit zu betrachten. Diese Erwägung ist nicht neu, siehe (QS05), (SH07), aber in allen uns bekannten Verfahren wurde dabei das Problem stets so vereinfacht, dass es nicht auf Probleme anwendbar ist, die in der Realität auftreten.

3.4 Realisierung

Die Umsetzung der Lösungsidee wird mit Hilfe von Techniken aus der ganzzahligen Programmierung und der kombinatorischen Optimierung verwirklicht. Ein ganzzahliges Programm (IP) ist eine mathematische Modellierungssprache. Ein IP besteht aus: *den Variablen, den Bedingungen und der Zielfunktion*. Die Bedingungen sind lineare (Un)Gleichungen und die Zielfunktion eine lineare Funktion, die jeweils auf den Variablen definiert sind. Ein Variablen-Belegung nennt man zulässig oder eine zulässige Lösung, wenn alle Bedingungen erfüllt sind. Das Ziel eines IP ist es, eine zulässige Lösung zu finden, bei dem die Zielfunktion das Minimum für alle zulässigen Lösungen annimmt. Desweiteren fordern wir für eine zulässige Lösung, dass alle Variablen ganzzahlige Werte angenommen haben. Es ist bekannt, dass im allgemeinen das Finden einer optimalen Lösung eines IP ein NP-schweres Problem darstellt. Somit kann man nicht erwarten, schnell eine Lösung zu finden (in diesem Zusammenhang heißt schnell in einem Jahr oder sogar später). Allerdings zeigen die Erfahrungen der letzten Jahre, dass viele Probleme auf diese Art effizient modelliert und gelöst werden können. Beide Schritte der Lösungsmethode wurden mittels ganzzahliger Programmie-

$$
\begin{aligned}
&\min -x_1 - x_2 - x_3 \\
&s.t. \\
&\qquad \begin{aligned} x_1 + x_2 &\leq 1 \\ x_1 + x_3 &\leq 1 \\ x_2 + x_3 &\leq 1 \\ x_i &\in \{0,1\} \end{aligned}
\end{aligned}
$$

Abbildung 1: Ein Beispiel-IP

$x_1 = 1$	$x_2 = 0.5$	$x_3 = 0.5$	unzulässig
$x_1 = 0.5$	$x_2 = 0.5$	$x_3 = 0.5$	unzulässig
$x_1 = 0$	$x_2 = 0$	$x_3 = 0$	zulässig, suboptimal
$x_1 = 1$	$x_2 = 0$	$x_3 = 0$	zulässig, optimal

Abbildung 2: Und einige mögliche Variablen Belegungen

rung umgesetzt, wobei die Implementierung des zweiten Schrittes einfach umzusetzen ist (wenn man sich mit kombinatorischen Algorithmen auskennt), ist der erste Schritt nicht trivial zu realisieren. Dafür benötigt man tiefgehende Resultate aus der Kombinatorik und der Polyedertheorie. Die zentrale Herausforderung besteht dabei darin, die Raum-Konflikt-Beziehungen auszudrücken, ohne die Räume selbst in Betracht zu ziehen. Dafür konstruiert man sich zu jedem Zeitpunkt p einen bipartiten Graphen $G = (\mathcal{C} \cup \mathcal{R}, E)$.
Die Knotenmenge $\mathcal{C}$ repräsentiert dabei die Menge der Kurse, die zu dem Zeitpunkt p stattfinden könnten und die Knotenmenge $\mathcal{R}$ repräsentiert die Menge der Räume, die zu dem Zeitpunkt p verfügbar sind. Zwischen Knoten $c \in \mathcal{C}$ und $r \in \mathcal{R}$ existiert eine Kante, wenn der Kurs c in dem Raum r stattfinden kann. Mit diesem Graph können wir nun Bedingungen, finden, mit denen wir eine raumkonfliktefreie Zuweisung der Veranstaltungstermine im zweiten Schritt garantieren können, ohne dabei die Räume selber zu betrachten. Um dies umsetzen zu können, mussten wir ein bekanntes Resultat aus der Kombinatorik, *das marriage theorem*, stärker formulieren.

Die Zeitkonflikt-Beziehungen werden ebenfalls mit Hilfe eines Graphen modelliert. Zu jedem Zeitpunkt p konstruieren wir einen Zeitkonflikt-Graphen $G_{conf} = (\mathcal{C}, E)$. Jeder Knoten $x_{c,p} \in \mathcal{C}$ repräsentiert dabei einen Kurs c, der zum Zeitpunkt p stattfinden könnte. Zwischen zwei Knoten $x_{c_1,p}, x_{c_2,p} \in \mathcal{C}$ existiert eine Kante, wenn beide Kurse nicht parallel stattfinden können. Die Veranstalungstermine einer zulässigen Lösung werden also von einer Teilmenge $\mathcal{F}$ der Knoten des Zeitkonflikt-Graphen dargestellt. Für je zwei beliebige $x_{c_i,ts_j}, x_{c_k,ts_l} \in \mathcal{F}$ gilt, dass die Kante $x_{c_i,ts_j} x_{c_k,ts_l}$ keine Element aus $E(G_{conf})$ ist. Somit ist $\mathcal{F}$ ein *stable set* in G_{conf}.

3.5 Beispiel

Um die Lösungsmethode besser zu verstehen, geben wir ein einfaches Beispiel an. Offensichtlich ist dieses Beispiel sehr viel kleiner als jede Universität. Aber die zu realen Universitäten gehörigen IPs könnte man niemals übersichtlich darstellen. In der Instanz aus Tabelle 1 nehmen wir an, dass jeder Kurs zu je-

(a) Raum Beschreibung

	Attribut 1	Attribut 2
Raum R_1	1	1
Raum R_2	1	0

(b) Kurs Beschreibung

	# Termine	Dozent	Attribut 1	Attribut 2
C_1	1	1	1	1
C_2	1	2	1	0
C_3	2	3	1	1
C_4	1	2	1	0

(c) Curricula Beschreibung

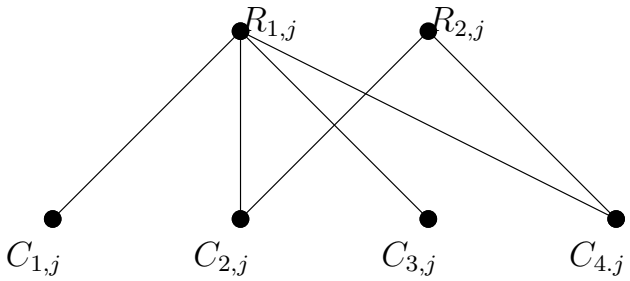
	Curricula 1	Curricula 2
C_1	1	0
C_2	1	0
C_3	0	1
C_4	0	1

(d) Mögliche Kurstermine

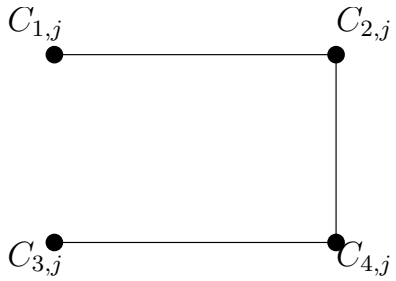
	Termin 1	Termin 2	Termin 3	Termin 4
C_1	1	1	1	1
C_2	1	1	1	1
C_3	1	1	1	1
C_4	1	1	1	1

Tabelle 1: Beschreibung der Testproblem-Instanz

dem Zeitpunkt stattfinden kann. Die dazugehörigen Raumkonflikt- und Zeitkonflikt-Graphen werden in Abbildung 3.5 dargestellt. Da wir angenommen haben, dass jeder Kurs immer stattfinden kann, sehen die Raumkonflikt- und Zeitkonflikt-Graphen zu jedem Zeitpunkt gleich aus. Nun sind wir in der Lage, die



(a) Raumkonflikt-Graph zum Zeitpunkt j



(b) Zeitkonflikt-Graph zum Zeitpunkt j

IP-Formulierung des ersten Schrittes anzugeben.

$$\min \sum_{x_{c_i,p_j}} c_{i,j} \cdot x_{c_i,p_j} \tag{1}$$

$$s.t. \tag{2}$$

$$x_{c_1,p_j} + x_{c_3,p_j} \leq 1 \qquad \forall j \in \{1,2,3,4\} \tag{3}$$

$$\sum_{i=1}^{4} x_{c_i,p_j} \leq 2 \qquad \forall j \in \{1,2,3,4\} \tag{4}$$

$$x_{c_1,p_j} + x_{c_2,p_j} \leq 1 \qquad \forall j \in \{1,2,3,4\} \tag{5}$$

$$x_{c_2,p_j} + x_{c_4,p_j} \leq 1 \qquad \forall j \in \{1,2,3,4\} \tag{6}$$

$$x_{c_3,p_j} + x_{c_4,p_j} \leq 1 \qquad \forall j \in \{1,2,3,4\} \tag{7}$$

$$\sum_{j=1}^{4} x_{c_i,p_j} = 2 \qquad \forall i \in \{4\} \tag{8}$$

$$\sum_{j=1}^{4} x_{c_i,p_j} = 1 \qquad \forall i \in \{1,2,3\} \tag{9}$$

$$x_{c_i,p_j} \in \{0,1\} \tag{10}$$

Die Summe 1 ist die Zielfunktion des IP's und deren Koeffizienten stellen die Terminwünsche der Kursanbieter da. Die Ungleichungen 3 und 4 sind die Hall-Bedingungen und die Ungleichungen 5-7 sind die Bedingungen, die die Zeitkonflikt-Beziehungen betrachten. Die Gleichungen 8-9 stellen sicher, dass jedem Kurs eine korrekte Anzahl an Veranstaltungsterminen zugewiesen wird.
Offensichtlich existiert eine zulässige Lösung des IP 1-10 (dem IP des ersten Schrittes). Solch eine Lösung kann einfach zu einer Lösung des gesamten Problems erweitert werden. In Tabelle 2 sind zwei mögliche zulässige Lösungen abgebildet.

	Termin Nr.	Zeitpunkt	Raum
C_1	1	1	1
C_2	1	2	2
C_3	1	3	1
C_4	1	1	2
C_4	2	2	1

	Termin Nr.	Zeitpunkt	Raum
C_1	1	2	1
C_2	1	1	1
C_3	1	3	1
C_4	1	2	2
C_4	2	3	2

Tabelle 2: Zwei zulässige Lösungen des Beispiels

4 Optimierungs-Ergebnisse

Alle Berechnungen wurden auf einem 3.2 GHz Pentium 4 Linux PC mit 1GB Arbeitsspeicher durchgeführt. Die IP's wurden mit CPLEX 10.1, einem kommerzielles Software-Tool, gelöst. Wir stellen die Laufzeiten der beiden Schritte dar. Die Laufzeit des ersten Schrittes besteht nicht nur aus dem Lösungsprozess des IP's, sondern auch aus einigen Berechnungen, die vorab erfolgen müssen. Für eine genauere Beschreibung siehe (Lac07) oder (LL07).

Für diejenigen, die nicht auf dem Gebiet des Timetabling beheimatet sind, merken wir an, dass es mit den heutigen Methoden häufig nicht möglich ist einen zulässigen Stundenplan in einer angemessenen Laufzeit zu finden. Die meisten Algorithmen erstellen Stundenpläne, die zu kleinen Probleme gehören, sehr schnell. Aber je größer das Problem wird, desto stärker wächst die Laufzeit des Algorithmus an. Mit den bisher bekannten Methoden ist es nicht möglich, einen Stundenplan für mehr als 300 Kurse zu erstellen.

4.1 Der internationale PATAT08 Timetabling Wettbewerb

Im Rahmen der PATAT08 Konferenz findet ein internationaler Timetabling Wettbewerb statt. Dafür wurden sieben verschiedene Probleminstanzen veröffentlich. In Tabelle 3 sind die Statistiken unserer Lösungsmethode dargestellt. Dabei müssen wir allerdings bemerken, dass wir die Laufzeit darstellen, die wir benötigten, um nur alle harten Nebenbedingungen aufzulösen.

Name	Kurse	Kurs-Termine	Räume	Konflikte	Teil 1	Teil 2
comp01	30	160	6	0	0.05 sec.	0.02 sec.
comp02	82	283	16	0	0.19 sec.	0.02 sec.
comp03	72	251	16	0	0.17 sec.	0.02 sec.
comp04	79	286	18	0	0.24 sec.	0.03 sec.
comp05	54	152	9	0	0.56 sec.	0.02 sec.
comp06	108	361	18	0	0.43 sec.	0.04 sec.
comp07	131	434	20	0	0.57 sec.	0.04 sec.

Tabelle 3: Statistiken und Resultate der PATAT08 Instanzen

4.2 Statistiken und Resultate für simulierte Daten

Name	Kurse	Kurs-Termine	Räume	Konflikte	Teil 1	Teil 2
klein	180	420	35	0	54 sec.	3 sec.
mittel	950	2100	165	0	359 sec.	6 sec.
groß	2100	4640	345	0	6341 sec.	5 sec.

Tabelle 4: Statistiken und Resultate für simulierte Instanzen

Wie man leicht sehen kann, sind die PATAT08 Instanzen keine Herausforderung für unser Lösungsmodell unserer Interpretation des curriculum based timetabling Problems. Um einen besseren Eindruck von der Leistungsfähigkeit unserer Methode zu erlangen, haben wir ein Simulationsprogramm entwickelt, mit dem große und realitätsnahe Probleminstanzen erzeugt werden können. In Tabelle 4 zeigen wir die Statistiken der drei simulierten Probleminstanzen. Dabei spiegelt die großse Instanz eine Universität der Größe der TU Berlin wider. Somit haben wir gezeigt, dass ein zulässiger Stundenplan für eine Universität der Größe der TU Berlin in unter zwei Stunden erstellt werden kann. Zusätzlich werden sogar die Wünsche der Kursanbieter sehr gut berücksichtigt.

	klein	mittel	groß
Zeit-Priorität	1.2	1.2	1.5
Raum-Priorität	1.2	1.2	1.2

Tabelle 5: Durchschnitts-Prioritäten der Instanzen

5 Zusammenfassung

In diesem Artikel haben wir einen Algorithmus zur Lösung des curriculum based course timetabling Problems vorgestellt, mit dem Lösungen für Instanzen gefunden werden können, die an großen Universitäten

wie der TU Berlin (ca. 30.000 Studierende) auftreten.

Die Größe der Instanzen, die mit unserem Algorithmus gelöst werden können, liegt um einen Faktor 10 über der Größenordnung, die mit bisher bekannten Algorithmen gelöst werden kann, wobei die benötigte Rechenzeit mit unter 2 Stunden auf einem handelsüblichen aktuellen PC vollkommen ausreichend ist.

Unser Algorithmus konnte bisher nur auf simulierten Testinstanzen ausprobiert werden, da die Daten an der TU Berlin nicht in geeigneter Form vorliegen. Eine Umsetzung an der TU Berlin ist geplant, und es liegt bereits die Anfrage einer anderen deutschen Universität zum Einsatz vor.

Literatur

[BHW06] J.J.J. van den Broek, C.A.J. Hurkens und G. Woeginger. Timetabling problems at the TU Eindhoven. In Burke und Rudová (BR07), Seiten 141–156.

[BJKW97] E. Burke, K. Jackson, J.H. Kingston und R. Weare. Automated University Timetabling: The State of the Art. *Comput. J.*, 40(9):565–571, 1997.

[BR07] E.K. Burke und H. Rudová, Hrsg. *PATAT 2006: Proceedings of the 6th International Conference on the Practice and Theory of Automated Timetabling*, Lect. Notes Comp. Science, Berlin, 2007. Springer. To appear; available online at `http://patat06.muni.cz/proceedings.html`.

[DB05] S. Daskalaki und T. Birbas. Efficient solutions for a university timetabling problem through integer programming. *European J. Oper. Res.*, 127(1):106–120, January 2005.

[DGMS07] L. Di Gaspero, B. McCollum und A. Schaerf. The Second International Timetabling Competition (ITC-2007): Curriculum-based Course Timetabling (Track 3). 2007.

[GJ78] M.R. Garey und D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, Berlin, 1978.

[ITC] ITC2007: 2nd International Timetabling Competition. `http://www.cs.qub.ac.uk/itc2007/`.

[JLP+06] S. Jeschke, R. Luce, O. Pfeiffer, E. Zorn, S. Grottke, G. Lach und J. Sablatnig. MosesKonto: Optimiertes Verteilungsverfahren für Tutorien und Studierendenverwaltung an der TU Berlin. *Proceedings of the DeLFI 2006*, Seiten 385–386, 2006.

[JLPZ07] S. Jeschke, R. Luce, O. Pfeiffer und E. Zorn. An optimized algorithm for distributing large numbers of students to small exercise groups. *Proceedings of the ICALT 2007*, Seiten 232–234, 2007.

[Lac07] G. Lach. Modelle und Algorithmen zur Optimierung der Raumvergabe der TU Berlin. Diplomarbeit, Technische Universität Berlin, Institut für Mathematik, 2007. In German.

[Lac08] M. Lach. Ein Verfahren zur Optimierung der Klausurterminplanung an der TU-Berlin. Diplomarbeit, Technische Universität Berlin, Institut für Mathematik, 2008. In German, to appear.

[Lew07] R. Lewis. A survey of metaheuristic-based techniques for University Timetabling problems. *OR Spectrum*, 2007. In press.

[LL07] G. Lach und M. Luebbecke. Optimal University Course Timetables and the Partial Transversal Polytope. 2007. submitted to IPCO 2008: Integer Programming and Combinatorial Optimization in Bertinoro, Italy.

[MO07] C. Meyers und J.B. Orlin. Very Large-Scale Neighborhood Search Techniques in Timetabling Problems. In Burke und Rudová (BR07), Seiten 36–52.

[QS05] A. Qualizza und P. Serafini. A Column Generation Scheme for Faculty Timetabling. In E.K. Burke und M.A. Trick, Hrsg., *PATAT 2004: Proceedings of the 5th International Conference on the Practice and Theory of Automated Timetabling*, Jgg. 3616 of *Lect. Notes Comp. Science*, Seiten 161–173, Berlin, 2005. Springer.

[RFL06] G. Ribeiro Filho und L.A.N. Lorena. An Integer Programming Model for the School Timetabling Problem. In *XIII CLAIO: Congreso Latino-Iberoamericano de Investigación Operativa*, 2006.

[SH07] K. Schimmelpfeng und S. Helber. Application of a real-world university-course timetabling model solved by integer programming. *OR Spectrum*, 29:783–803, 2007.

[uJHK95] T. B. Cooper und J. H. Kingston. The Complexity of Timetable Construction Problems. *Proceedings of the First International Conference on the Practice and Theory of Automated Timetabling (ICPTAT '95)*, 1995.