$$\min\{f(x)\,|\,x\in S\}$$

LEHRBUCH

Josef Kallrath

Gemischt-ganzzahlige Optimierung: Modellierung in der Praxis

Mit Fallstudien aus Chemie, Energiewirtschaft, Papierindustrie, Metallgewerbe, Produktion und Logistik

2. Auflage

Springer Spektrum

Gemischt-ganzzahlige Optimierung: Modellierung in der Praxis

Josef Kallrath

Gemischt-ganzzahlige Optimierung: Modellierung in der Praxis

Mit Fallstudien aus Chemie, Energiewirtschaft, Papierindustrie, Metallgewerbe, Produktion und Logistik

2., überarbeitete und erweiterte Auflage

Springer Spektrum

Josef Kallrath
Weisenheim am Berg, Deutschland
josef.kallrath@t-online.de

ISBN 978-3-658-00689-1 ISBN 978-3-658-00690-7 (eBook)
DOI 10.1007/978-3-658-00690-7

Springer Spektrum

Lektorat: Ulrike Schmickler-Hirzebruch | Barbara Gerlach

Gedruckt auf säurefreiem und chlorfrei gebleichtem Papier.

Springer Spektrum ist eine Marke von Springer DE. Springer DE ist Teil der Fachverlagsgruppe Springer Science+Business Media
www.springer-spektrum.de

Vorwort zur 2. Auflage

Bei dem vorhandenen Interesse an meinem Werk fällt es leicht und bereitet Freude, genau 10 Jahre nach Fertigstellung der ersten Auflage, die erste Auflage um einige Inhalte zu ergänzen, die nach einer Dekade an Bedeutung gewonnen oder zu einer Perspektivenänderung geführt haben. Hierzu zählen *Polylithische Modellierungs- und Lösungsansätze* mit Anwendungen aus der Papierindustrie und die neu aufgenommenen Verschnittprobleme aus der Metallindustrie, bei denen der Einsatz globaler Optimierungstechniken sinnvoll ist.

Überhaupt wird dem Gebiet der *Globalen Optimierung* mehr Bedeutung zu gemessen, da sich dieses Gebiet erst nach dem Erscheinen der ersten Auflage im Jahre 2002 richtig zu entwickeln begonnen hat, weil etwa zu dieser Zeit kommerzielle Software mit deterministischen globalen Optimierungstechniken verfügbar wurde. Zwar gehört es nicht direkt in die Klasse gemischt-ganzzahliger Optimierungsprobleme, aber die hier verwendeten Lösungstechniken haben enge Verwandtschaft mit den Verfahren in der gemischt-ganzzahligen Optimierung. Eine weitere Kopplung von Globaler Optimierung und gemischt-ganzzahliger Optimierung zeigt sich bei der Berechnung optimaler Breakpointsysteme im Zusammenhang mit der Modellierung nichtlinearer Terme mit Hilfe von SOS-2 Variablen. Im Zusammenhang mit gemischt-ganzzahlig linearen Problemen wurde im Kapitel *Polylithische Modellierung* auch das Thema *Lagrange-Relaxierung* zur Verbesserung unterer Schranken aufgenommen.

Im Kapitel *Optimierung in der Praxis* wurde ein Abschnitt zu algebraischen Modellierungssprachen aufgenommen sowie in einem weiteren Abschnitt thematisiert, dass Optimalität möglicherweise nicht genug zur Lösung praktischer Probleme ist und auf die Bedeutung von Nachhaltigkeit und deren Notwendigkeit nach einem möglichen Optimierungsprojekt hingewiesen.

Fehler der ersten Auflage wurden korrigiert; den Studierenden und Lesern, die mich darauf hingewiesen haben, sei dafür gedankt. Wie auch schon in der ersten Auflage, wird in diesem Werk eine streng formale Vorgehensweise vermieden; es erschien wichtiger, die zur Lösung praktischer Fragestellung erforderlichen Aspekte, wozu auch Abbildungstreue, geistige Strenge und Genauigkeit sowie Sicherheit in den möglicherweise abzuleitenden Schlüssen zählen, in den Vordergrund zu stellen. Möge auf diesem Hintergrund und dieser Zielsetzung auch die vorliegende, erweiterte 2. Auflage immer wieder mit Freude und Gewinn gelesen werden und zu eigenen Modellierungsleistungen motivieren.

Danksagung

Es ist eine Freude, wieder einigen Freunden und Kollegen zu danken, die mich während vieler Jahre in meinem Arbeits- oder privaten Umfeld begleitet haben und in verschiedener Weise direkt oder indirekt zum Gelingen dieses Werkes beigetragen haben. Dies gilt für all diejenigen, welche schon in der Danksagung der ersten Auflage genannt wurden. In der zwischen 2002 und 2012 vergangenen Zeit sind jedoch viele neue Kollaborationspartner, zu denen ein enges und vertrautes Verhältnis erwuchs oder aus denen Freunde wurden, hinzugekommen. Hier zählen insbesondere Dr. Franz Nelißen und Dr. Michael

Bussieck (GAMS GmbH, Köln), Prof. Dr. Christoudoulos A. Floudas (Princeton University, Princeton, NJ, USA), Prof. Dr. Panos M. Pardalos (Center of Applied Optimization, University of Gainesville, Gainesvielle, FL) und Prof. Dr. Steffen Rebennack (Colorado School of Mines, Golden, CO). Die Kontakte, die aus meiner Leitung der GOR Arbeitsgruppe Praxis der mathematischen Optimierung" erwuchsen, trugen indirekt auch zu dieser erweiterten zweiten Auflage bei.

Ein besonderes Wort des Dankes sei an all die Projektpartner vieler Jahre gerichtet: Viele von ihnen habe ich wegen ihres Enthusiasmus und ihrer tiefen Kenntnisse in den Problemfeldern, die sie mit Hilfe gründlicher mathematischer Modellierung, die häufig auf gemischt-ganzzahliger Optimierung führte, verbessert sehen wollten, in guter Erinnerung. Die Interaktion und Kommunikation mit ihnen ist eine wichtige und unersetzliche Quelle für das Gelingen dieses Buches. Aus der Vielzahl der Projektpartner, die über mehrere Jahre hinweg indirekt zu diesem Buch beigetragen haben, seien genannt: Dr. Wolfram Schmidt (Ludwigshafen, jetzt Kirchheim an der Weinstraße) und sein Team mit Dr. Markus Klumpe und Bernd Heisel-Hoffmann (Ludwigshafen), Norbert Vormbrock (Ludwigshafen) mit einer gemeinsamen Bonner Vergangenheit und Dr. Gerd Fischer (BASF SE, Ludwigshafen). Leider ist im industriellen Umfeld mehr noch als vor 10 Jahren zu bemerken, dass der Zeitgeist – besser Unzeitgeist dieser Zeit – mit seiner eher eindimensionalen, monetär kurzfristigen Messskala die Chancen, auf derartige Menschen, mit dem Bedürfnis, ein Problem so tief und so gründlich wie möglich zu verstehen und zu lösen, zu treffen, immer geringer werden, und so man denn auf solche trifft, diese Menschen leider auch immer weniger Zeit haben, ein Problem in Ruhe und Gründlichkeit anzugehen. Es bleibt zu befürchten, dass die Menge der nicht optimal gelösten Probleme derartig erdrückend wird, dass dies zu großen Wettbewerbsnachteilen und kultureller Schwächung führen wird. Vielleicht findet dadurch der Unzeitgeist sein wünschenswertes und unrühmliches Ende und – so die Hoffnung – wird einmünden in eine Denkweise, die auf Gründlichkeit und Nachhaltigkeit setzt und nicht mit 80%-Lösungen zufrieden ist. Gute, nachhaltige Lösungen benötigen Verständnis, Liebe zum Detail, den Willen, ein Problem aus eigenem Antrieb heraus grundlegend lösen zu wollen und Zeit – den oben genannten Projektpartnern wünsche ich, dass ihnen solche Zeit verbleibt und sie beispielsweise auch dieses Werk mit Muße genießen können.

Für gründliche Prüfung und Korrekturlesen des Manuskriptes sei Michael Bussieck (GAMS GmbH, Köln), Prof. Dr. Steffen Rebennack und Timo Lohmann (Colorado School of Mines, Golden, CO), Dr. Alexander Badinski, Dr. Gerd Fischer, Markus Horenburger und Steffen Klosterhalfen (BASF SE, Ludwigshafen), Hermann Gold (Infineon Technologies AG, Regensburg) und Prof. Dr. Siegfried Jetzke (Ostfalia Hochschule für angewandte Wissenschaften, Salzgitter) herzlichst gedankt; viele konstruktive Kommentare habe ich gern aufgenommen und haben das Buch verbessert. Die Qualität der Abbildungen – mehrere davon in Neuanfertigung – wurde dank der Hilfe von Dr. Julia Kallrath (Weisenheim am Berg) erheblich verbessert. Schließlich sei Ulrike Schmickler-Hirzebruch gedankt, die auf Seiten des Springer Spektrum Verlages die Aufgabe des Lektorats inne hatte und über viele Jahre in Kontakt blieb und somit auch letztlich an der Entstehung dieser zweiten Auflage ein Stück Mitverantwortung trägt.

Weisenheim am Berg im August 2012 Josef Kallrath

Vorwort

Dieses Buch beschreibt und lehrt ausgehend von praktischen Fragestellungen, wie in der Industrie – vornehmlich der Prozessindustrie, aber auch anderen Gewerben – gemischt-ganzzahlige Optimierung eingesetzt wird, wie Probleme modelliert und erfolgreich gelöst werden. Es verbindet Modellbildung und algorithmische Aspekte aus den Bereichen kontinuierlicher und gemischt-ganzzahliger, linearer und nichtlinearer und globaler Optimierung. Betrachtet wird auch der Einfluss dieser Methodik in der heutigen Industriegesellschaft insbesondere auf dem Hintergrund von Supply Chain Management.

Es wendet sich an Studierende der Mathematik, Physik oder Wirtschaftswissenschaften mit Interesse an Modellbildung, Optimierung und der Lösung realer Probleme, sowie an Praktiker und Berater. Es tut dies ausgehend von praktischen Problemen in methodisch wissenschaftlicher Weise, veranschaulicht den Modellbildungs- und Lösungsprozess mit Hilfe vieler Fallstudien, und zeigt, welche Techniken in der Praxis verwendet werden und welche Schwierigkeiten dort auftreten. Es stellt in diesem Sinne ein Bindeglied dar zwischen den eher algorithmisch-theoretisch ausgerichteten Fachbüchern mathematischer Optimierung und den methodisch weniger in die Tiefe gehenden Werken, die sich auf die Beschreibung der Anwendungen konzentrieren. Das Buch ist für Leser gedacht, die sich dafür interessieren, *wie* Mathematik angewendet wird und bietet vielleicht für Studierende in den ersten Studienjahren die Motivation, im weiteren Verlauf des Studiums die mathematische Optimierung als Schwerpunkt in Betracht zu ziehen.

Da in diesem Buch Modelle, Modellbildung und Anwendungen der Modelle sowie vertiefende Fallstudien im Vordergrund stehen, wird im Eingangskapitel mit einem Abschnitt „*Zur Bedeutung von Modellen und Modellierung*" begonnen. Der Erfolg von Modellen beruht auf einer engen Verknüpfung von Modellformulierung und Algorithmen. Um verständlich zu machen, warum einige Modellformulierungen gutartiges, andere ein klägliches Laufzeitverhalten haben, werden in angemessenem Umfang Algorithmen zur gemischt-ganzzahligen, linearen und nichtlinearen sowie globalen Optimierung methodisch behandelt. Lösungstechniken für die dabei auftretenden kontinuierlichen linearen und nichtlinearen Optimierungsprobleme werden ebenfalls detailliert beleuchtet.

Mit Hilfe der mathematischen Optimierung lassen sich erhebliche Kosten einsparen oder Gewinnsteigerungen erzielen; im Buch werden Beispiele genannt, in denen durch bessere Nutzung verfügbarer Ressourcen finanzielle Verbesserungen in Höhe einiger Millionen Dollar gewonnen wurden. Die mathematische Optimierung bietet Lösungsansätze, die es erlauben, Probleme zu lösen, die wegen ihrer Dimension oder Struktur mit anderen Methoden nicht gelöst werden können. Bedeutsam ist hierbei auch, dass uns die Ergebnisse mathematischer Optimierung in unserem täglichen Leben überall umgeben: Die Befriedigung des privaten Wasserbedarfs in einer Großstadt wie London, energieminimale Fahrweisen von U-Bahnen, Hub-Lokalisations-Probleme im Flugverkehr, Frequenzzuweisungen in Mobilfunknetzwerken, Minimalflächen beim Autoscheinwerferdesign, die Beschriftung von Landkarten, die automatische Erzeugung von Flussdiagrammen, Fahrdienstplanung im Kranken- und Behindertentransport oder die Steuerung der Energieproduktion unter Berücksichtigung von Spitzenzeiten - bei all diesen Beispielen wäre ein Aufkleber *mathematical optimization inside* angebracht; Zeitschriften wie *Journal of Operational Research*, *OR Spectrum* oder *European Journal of Operational Research* bieten eine Fülle weiterer Anwendungsbeispiele.

Das Buch greift zum Teil zurück auf Vorlesungen, die an der Universität Heidelberg und University of Florida (Gainesville) gehalten wurden, sowie auf Konferenzbeiträge oder Projektberichte kommerzieller Industrieprojekte und fügt diese zu einem zusammenhängenden instruktiven Lehrbuch zusammen. Dabei wurde bewusst der Schwerpunkt auf Modellierung und didaktische Gesichtspunkte gelegt, mit dem Ziel, dem interessierten Neuling in den Fächern und Disziplinen Angewandte Mathematik, Ingenieurswissenschaften, Operations Research und Wirtschaftswissenschaften einen guten Einstieg in die gemischt-ganzzahlige Optimierung zu ermöglichen, aber auch dem erfahrenden Praktiker einen Überblick über Techniken und hilfreiche Hinweise zu geben, die entscheidend dafür sind, komplexe Probleme zu lösen oder weiterführende Ideen zu entwickeln.

Aufbau des Buches

Das Buch beginnt mit einem kurzen Überblick zur Geschichte der Optimierung, speziell der gemischt-ganzzahligen Optimierung. Da bei dieser die Modellierung eine besondere Rolle spielt, wird ihr in Kapitel 1 ein besonderer Abschnitt *Zur Bedeutung von Modellen* gewidmet. Schließlich werden das grundsätzliche Vorgehen bei Optimierungsproblemen und die grundlegenden Elemente eines Optimierungsmodells einführend diskutiert: die Variablen, die Nebenbedingungen und die Zielfunktion. Abschließend wird eine formale Definition gemischt-ganzzahliger Optimierungsprobleme gegeben.

In Kapitel 2 werden einige einführende Beispiele zur Modellbildung linearer gemischtganzzahliger und nichtlinearer kontinuierlicher Optimierung gegeben; sie sollen ein erstes Verständnis im Umgang mit der Struktur und Bestandteilen von Optimierungsmodellen vermitteln. Dabei wird auf die schon in Kapitel 1 gelegten allgemeinen Grundlagen zur Modellbildung Bezug genommen. Bei den noch recht übersichtlichen Problemen ist es noch möglich, die Probleme exakt in einem mathematischen Modell abzubilden.

Aspekte der Optimierung in der Praxis, die insbesondere in der Strukturierungs- oder Frühphase eines Projektes Beachtung finden sollten, werden in Kapitel 3 diskutiert. Hier wird auch ausführlich auf Optimierung im Supply Chain Management eingegangen.

Kapitel 4 gibt einen einführenden Überblick über den mathematischen und algorithmischen Hintergrund der linearen Programmierung (LP), der gemischt-ganzzahligen linearen Programmierung (MILP), der kontinuierlichen nichtlinearen Programmierung (NLP), der gemischt-ganzzahligen nichtlinearen Programmierung (MINLP) und schließlich der Globalen Optimierung (GO). Eine strengere und ausführlichere mathematische Behandlung einiger Aspekte, die dem kombinatorischen Umfeld der Algorithmen zuzuordnen sind, ist dem Anhang A vorbehalten. Hier und auch in den späteren Kapiteln wird die lineare vor der nichtlinearen Optimierung bzw. Modellierung behandelt. Zwar ist die nichtlineare, insbesondere die nichtlineare unbeschränkte, Optimierung in einem gewissen Sinne intuitiv und konzeptionell einfacher als die lineare Optimierung, weil sich die Grundgedanken z. B. anhand eindimensionaler Beispiele erläutern lassen, und zudem die lineare Optimierung elegant als Spezialfall der nichtlinearen Optimierung abgeleitet werden kann, aber schließlich wurde der linearen Optimierung doch der Vorrang gegeben. Entscheidend für die Wahl dieser Reihenfolge war das Argument, dass die Aufstellung linearer Modelle in der Regel doch einfacher ist als die von nichtlinearen Modellen und die mathematischen Anforderungen an ein technisches Verständnis der Algorithmen geringer sind. Da es andernfalls den Rahmen dieses Buches sprengen würde, wird über Algorithmen zur Lösung nichtlinearer kontinuierlicher Probleme nur ein knapper Überblick gegeben, jedoch bei der Modellierung die Bedeutung der Wahl der Startwerte und Konvergenzeigenschaften der Algorithmen hervorgehoben.

Ein zentraler Bestandteil des Buches ist Kapitel 5 mit der Vorstellung verschiedener spezieller Konstrukte der Modellierung, die in praktischen Problemen auftreten. Dazu zählen logische Relationen, die sich mit Hilfe von Binärvariablen formulieren lassen, aber auch bestimmte nichtlineare Strukturen, die in gemischt-ganzzahlige Formulierungen transformiert werden können. Es werden Methoden und Formulierungen behandelt, mit denen sich die Rechenzeit ganzzahliger Probleme erheblich reduzieren lässt: gute Modellbildung, spezielle Verzweigungstechniken und eine Kontrolle der Branch&Bound-Strategien. Während bei LP-Problemen sich gute Modellbildung zwar auch positiv auf die Rechenzeit auswirkt, ist sie bei den meisten gemischt-ganzzahligen Problemen für die effiziente Berechnung der Lösung geradezu notwendig. Für eine gute Modellbildung ist eine gründliche Kenntnis der Lösungsverfahren, aber vor allem auch Erfahrung sehr wesentlich. Die Wahl der richtigen Variablen und die geeignete Formulierung ihrer Beziehungen zueinander, nicht die Anzahl der Variablen und Nebenbedingungen, ist ausschlaggebend. Das Kriterium, das in der gemischt-ganzzahligen linearen Optimierung eine gute von einer schlechten Formulierung trennt, ist der „Abstand" zwischen der konvexen Hülle und dem zulässigen Bereich der Relaxierung des linearen Problems hinsichtlich der Ganzzahligkeitsbedingungen und damit die erforderliche Rechenzeit zur Bestimmung der Lösung. Neben einigen Aspekten effizienter Modellbildung, die vom Modellentwickler beeinflusst werden kann, werden in diesem Kapitel auch automatische Reformulierungsverfahren [283] für Optimierungsprobleme mit binären Variablen und *Preprocessing*-Techniken behandelt, die zu besseren Lösungsverhalten führen.

Kapitel 6 enthält einige Fallstudien zur Linearen Programmierung, die aus realen Problemen hervorgegangen sind und erfolgreich gelöst werden konnten. Behandelt werden die Optimierung der Produktion eines chemischen Reaktors, ein augenscheinlich nichtlineares Mischungsproblem, das sich unter Ausnutzung bestimmter Monotonieeigenschaften in ein lineares Problem transformieren lässt, und ein Verschnittproblem aus der Papierindustrie. Sämtliche Probleme besitzen Erweiterungen im Kontext ganzzahliger Optimierung. Während der Modellierungsphase treten immer wieder Fragen nach der Struktur der Zielfunktion auf; häufig sollen mehrere Zielkriterien simultan berücksichtigt werden. Deshalb werden in diesem Kapitel, wenngleich sie nicht in direktem Zusammenhang mit den Fallstudien stehen, schließlich *Goal-Programmierung*, eine spezielle Technik zur Lösung multikriterieller Optimierungsprobleme, und einige Grenzen der linearen Programmierung diskutiert.

Kapitel 7 behandelt Fallstudien zunehmender Komplexität zur gemischt-ganzzahligen linearen Optimierung mit den Themenfeldern: Standort- und Personaleinsatzplanung, Vertragsallokation, Metallverschnitt, Projektplanung und schließlich ein Routenplanung mit Abholung und Zeitfenstern. In den Kapiteln 9 und 10 werden Praxisprobleme diskutiert, die auf NLP- bzw. MINLP-Probleme führen. Schließlich enthält Kapitel 11 einige Beispiele zur Globalen Optimierung.

Das Glossar sowie der Sachindex am Ende des Buches – ergänzend sind hier auch die meisten der üblichen englischen Begriffe aus dem Umfeld der mathematischen Optimierung enthalten – mögen schließlich eine weitere Hilfe sein für Leser, die punktuell Interesse an einzelnen Themen oder Abschnitten im Buch haben.

Das Buch ist im wesentlichen so gegliedert, dass die Kapitel aufeinander aufbauen, Ausnahmen sind die Kapitel 6-10, die unabhängig voneinander sind und damit auch in beliebiger Reihenfolge gelesen werden können. Die Kapitel des Buches können natürlich in der angeordneten Reihenfolge gelesen werden, aber je nach Interesse mag ein Leser davon absehen wollen. Da sich das Buch an verschiedene Leserkreise richtet, sind nachfolgend

einige mögliche Pfade vorgeschlagen, die ein Leser je nach Vorkenntnis, Interesse und Ziel [Orientierung und Praxis (1); Praxis und mathematische Grundlagen (2), Unterstützung bei der eigenen Arbeit in der Praxis oder z. B. beim Anfertigen einer Diplom-, Master- oder Doktorarbeit (3)] beim Lesen dieses Buches einschlagen kann:

Fachrichtung	Hintergrund	Ziel	Pfad
Mathematiker Physiker	Grundstudium	(1)	1,2,3,(4),6-11,5,12
		(2)	1,2,3,(4),5,6-11,12
		(3)	(1,2,3),(4),5,6-11,(12)
Mathematiker Physiker	Hauptstudium	(1)	1,2,3,(4),5,6-11,12
		(2)	1,2,3,(4),5,6-11,4,5,12
		(3)	(1,2,3),(4),5,6-11,(12)
Wirtschafts-studiengänge	gute Kenntnisse in Mathematik	(1)	1,2,3,(4),5,6-11,12
		(2)	1,2,3,(4),5,6-11,4,5,12
		(3)	(1,2,3),(4),5,6-11,(12)
Wirtschafts-studiengänge	wenig Kenntnisse in Mathematik	(1)	1,2,3,(4),6-11,12,5
		(2)	1,2,3,(4),5,6-11,12
		(3)	(1,2,3),(4),5,6-11,(12)
Praktiker	gute Kenntnisse in Mathematik	(1)	1,2,3,(4),5,6-11,12
		(2)	(1,2),6-11,(3),(4),5,12,A
		(3)	(1,2,3),(4,A),5,6-11,(12)
Praktiker	wenig Kenntnisse in Mathematik	(1)	1,2,3,(4),6-11,12,5
		(2)	1,2,3,(4),6-11,5,12
		(3)	(1,2,3),(4),5,6-11,(12)

Zudem sind die meisten Abschnitte so geschrieben, dass sie trotz zahlreicher Querverweise für sich stehend gelesen werden können. Damit ist es möglich, auch punktuell mit diesem Buch zu arbeiten. Schließlich sei bemerkt, dass das Buch in den meisten Fällen elementar und leicht verständlich in eine Thematik einführt, möglicherweise dann aber ein Tempo einschlägt, das nicht allen Lesern adäquat erscheinen mag. Diese Vorgehensweise wurde gewählt, um das Buch so weit wie möglicht thematisch in sich abgeschlossen zu machen, ein möglichst breites Spektrum von Lesern anzusprechen und die Wahrscheinlichkeit zu erhöhen, dass jeder Leser für sich etwas interessantes findet und das Buch immer mal wieder mit Freude zur Lektüre in die Hand nimmt.

Danksagung

Es ist eine Freude, einigen Freunden und Kollegen zu danken, die mich während vieler Jahre in meinem Arbeits- oder privaten Umfeld begleitet haben und in verschiedener Weise direkt oder indirekt zum Gelingen dieses Werkes beigetragen haben. Hier möchte ich an erster Stelle Frau Dr. Anna Schreieck (Ludwigshafen) danken, die durch ihre besondere Fähigkeit, mit nicht einfachen Randbedingungen und Situationen umzugehen, den

Freiraum für viel, in der Industrie eher ungewöhnliches Tun ermöglicht hat. Wenngleich in den letzten Jahren eher aus der Ferne wirkend, so waren dennoch für das Entstehen dieses Buches die beiden folgend genannten Menschen sehr wichtig: mein langjähriger Studienfreund PD Dr. Horst Fichtner (Bochum) mit seiner wohlwollenden, aber gnadenlos kritischen Art sowie David Desantis (Neshanic Station, New Jersey, USA), der mit seinen nie ruhenden *general intellectual capabilities*, als Projektpartner später als Freund auf seine spezielle humorvolle und insistierende Weise für die Entwicklung vieler Ideen in diesem Buch verantwortlich war.

Dr. Thomas I. Maindl und Michael Schnorbach möchte ich für ihre Beiträge zum Thema „Supply Chain Management" und ihre Unterstützung in Abschnitt 3.5.1 danken. Seit vielen Jahren hinweg sind Bob Daniel und Marilyn Dalton (Blisworth, England) inspirative Gesprächspartner und mir wohl gesonnene Menschen, nicht nur im Umfeld gemischt-ganzzahliger Optimierung. Ihnen sei hierfür an dieser Stelle gedankt.

Ein besonderes Wort des Dankes sei an all die Projektpartner vieler Jahre gerichtet: viele von ihnen habe ich wegen ihres Enthusiasmus und ihrer tiefen Kenntnisse in den Problemfeldern, die sie mit Hilfe gründlicher mathematischer Modellierung, die häufig auf gemischt-ganzzahliger Optimierung führte, verbessert sehen wollten, in guter Erinnerung. Die Interaktion und Kommunikation mit ihnen ist eine wichtige und unersetzliche Quelle für das Gelingen dieses Buches. Aus der Vielzahl der Projektpartner, die über mehrere Jahre hinweg indirekt zu diesem Buch beigetragen haben, seien genannt: Dr. Peter Bassler (Ludwigshafen), David Desantis, Dr. Klaus Plitzko (Münster), Dr. Gerhard Schnabel und Dr. Tim Jungkamp (Ludwigshafen). Leider ist im industriellen Umfeld zu bemerken, dass die Chancen, auf derartige Menschen, mit dem Bedürfnis, ein Problem so tief und so gründlich wie möglich zu verstehen und zu lösen, zu treffen immer geringer wird, und so man denn auf solche trifft, diese immer weniger Zeit haben, ein Problem in dieser bewährten Methode anzugehen.

Zum Gelingen eines solchen Werkes bedarf es der gründlichen Prüfung und Korrektur des Manuskriptes. Hierfür sei PD Dr. Horst Fichtner (Bochum), Hans-Georg Freiermuth (New York), Dr. Susanne Heipcke (Marseille), Dr. Cornelia Liesenfeld (Augsburg), Isabelle Luidolt und Eltern (Mannheim) und Dr. Anna Schreieck (Ludwigshafen) herzlichst gedankt. Weiterhin sind in diesem Buch konstruktive Kommentare von Dr. Axel Hecker (Heidelberg), Dr. Gerhard Krennrich (Ludwigshafen), Dr. Marco Lübbecke (Braunschweig), Dr. Johannes P. Schlöder und Martin Spoden (Heidelberg), Christian Timpe (Ludwigshafen), Dr. Norbert Trautmann (Karlsruhe), Prof. Dr. Uwe Zimmermann (Braunschweig) und insbesondere von Prof. Dr. Arnold Neumaier (Wien) und Prof. Dr. Alexander A. Kolokolov (Omsk, Russland) eingeflossen. Schließlich sei Ulrike Schmickler-Hirzebruch gedankt, die auf Seiten des Vieweg-Verlags die Aufgabe des Lektorats inne hatte und durch zahlreiche Gespräche die Entstehung dieses Werkes unterstützte.

Ludwigshafen im August 2002 Josef Kallrath

Für die,

die den Sinn hinter den Worten verstehen ...

die das wahrhaft Schöne,
die Schauspiele des Himmels
mit Sonne, Mond, Planeten und Kometen
mit tiefer Freude erfüllt ...

die nicht aufhören, dem Zeitgeist zu widerstehen ...

Inhaltsverzeichnis

Abbildungsverzeichnis

Verzeichnis von Modellbeispielen

Die im Buch verwendeten Beispiele von Modellen werden hier alphabetisch und mit Seitenzahl zusammengestellt, damit sie einfacher im Buch gefunden werden können.

1 Einführung: Modelle, Modellbildung und Optimierung

Dieses Kapitel beginnt mit einem kurzen Überblick zur Geschichte der Optimierung, speziell der gemischt-ganzzahligen Optimierung, wobei vorgreifend schon einige Begriffe verwendet, die erst im späteren Verlauf des Buches erläutert werden. Da bei der gemischt-ganzzahligen Optimierung die Modellierung eine besondere Rolle spielt, wird ihr in diesem Kapitel ein eigener Abschnitt unter dem Titel „*Zur Bedeutung von Modellen*" gewidmet. Schließlich werden das grundsätzliche Vorgehen bei Optimierungsproblemen und die wesentlichen Elemente eines Optimierungsmodells, das sind die Variablen, die Nebenbedingungen[1] und die Zielfunktion, einführend diskutiert.

1.1 Was ist gemischt-ganzzahlige Optimierung?

Der Begriff der *Optimierung*, im mathematischen Sinne verwendet, bedeutet die Bestimmung des Maximums oder Minimums einer Funktion f, die auf einem (beschränkten) Bereich S oder Zustandsraum definiert ist. Die klassische Optimierungstheorie (Differentialrechnung, Variationsrechnung, Optimale Steuerung) behandelt die Fälle, in denen S zusammenhängend und die zu optimierende Funktion f, im Umfeld der Optimierung auch *Zielfunktion* genannt, zumindest stetig ist. In praktischen Problemen kann die mathematische Optimierung Entscheidungsprozesse unterstützen. Die anstehenden Entscheidungen sind mit den Freiheitsgraden des Problems verknüpft; in einem Optimierungsmodell werden sie durch die Variablen repräsentiert, denen durch den Optimierungsalgorithmus schließlich Werte derartig zugeordnet werden, so dass die Zielfunktion, die die Konsequenzen der Entscheidungen quantifiziert, einen optimalen Wert annimmt.

Die ganzzahlige und gemischt-ganzzahlige, manchmal auch synonym kombinatorische oder *diskrete Optimierung*[2] genannt, vor 1980 noch ein Randgebiet der Wissenschaft

[1] Der im deutschen Sprachgebrauch häufig verwendete Begriff *Nebenbedingungen* ist leider nicht so griffig und auch nicht so allgemein akzeptiert, wie der in der angelsächsischen Literatur verwendete Begriff *Constraints*. Synonym werden wir daher in diesem Buch auch die Begriffe Zulässigkeitsbedingungen, Beschränkungen, Restriktionen oder den englischen Ausdruck *Constraints* selbst verwenden. Auf diese Problematik der Terminologie wird nochmal ausführlicher in Abschnitt 1.6 eingegangen.

[2] Unter *kombinatorische* oder *diskreter Optimierung* könnte man zum einen die Disziplin verstehen, die sich mit Optimierungsproblemen beschäftigt, die diskrete Variablen (siehe Seite 11) enthalten. Zum anderen kann man sie, wie eher im Informatikumfeld üblich, als ein Teilgebiet der mathematischen Optimierung ansehen, deren Anliegen es ist, diskrete, endliche, kombinatorische Optimierungsprobleme effizient zu lösen. Dabei werden Methoden und Resultate

J. Kallrath, *Gemischt-ganzzahlige Optimierung: Modellierung in der Praxis*,
DOI 10.1007/978-3-658-00690-7_1, © Springer Fachmedien Wiesbaden 2013

und Teilgebiet der mathematischen Optimierung, spielt eine zunehmend bedeutsamere Rolle und umfasst Optimierungsprobleme, in denen alle oder zumindest einige Freiheitsgrade ganzzahlig bzw. diskret sind. Die Ganzzahligkeitsbedingungen rühren bei praktischen Fragestellungen z. B. daher, dass Null-Eins-Entscheidungen – etwa die, ob ein Arbeitsschritt von einem bestimmten Mitarbeiter zu einem Zeitpunkt bearbeitet wird oder nicht – zu treffen sind oder Größen wie Stufenzahl einer Kolonne oder Containergröße nur ganzzahlige oder nur bestimmte, diskrete Werte annehmen können. Einige Spezialfälle der ganzzahligen Optimierung, die Gemischt-Ganzzahlige Lineare Optimierung [engl.: *mixed integer linear programming; MILP*] und die Gemischt-Ganzzahlige Nichtlineare Optimierung [engl.: *mixed integer nonlinear programming; MINLP*] haben zwischen 1990 und 2012 erhebliche Verbreitung gefunden [121]. Sie werden zunehmend in den Gebieten Logistik, Transport[3], Produktionsplanung, Finanzen, Kommunikation[4] und Design[5] eingesetzt, zum Teil auch deshalb, weil sich mit ihrer Hilfe und insbesondere durch die Verwendung von Binärvariablen[6] Probleme der *kombinatorischen Optimierung*[7] wie das Rundreiseproblem [engl.: *traveling salesman problem*], Maschinenbelegungsprobleme, Reihenfolgeprobleme [engl.: *sequencing problems*], Transport- und Zuordnungsprobleme [engl.: *transport and assigment problems*] formulieren lassen.

Die Disziplin der mathematischen Optimierung profitiert erheblich von dem bemerkenswerten Fortschritt in der Entwicklung neuer Algorithmen und der beeindruckenden Steigerung der Rechnerleistungen. Mit Hilfe moderner Verfahren wie Innere-Punkte-Methoden [engl.: *interior point methods, IPM*) oder Sequentielle Quadratischer Programmierung [engl.: *sequential quadratic programming*] können große nichtlineare kontinuierliche Optimierungsprobleme gelöst werden; verbesserte Simplexverfahren lösen inzwischen z. B. in der zivilen Luftfahrt LP-Probleme mit mehr als einer Million Variablen und Hunderttausenden von Beschränkungen.

In der gemischt-ganzzahlige Optimierung [engl.: *mixed integer programming; MIP*] spielen Modellbildung (Kallrath & Wilson 1997,[167]; Williams 1973, [291]; Ashford & Daniel 1992,[21]) und Erfahrung eine bedeutsame Rolle. Die Wahl der richtigen Variablen und die geeignete Formulierung ihrer Beziehungen zueinander, nicht die Anzahl der Variablen und Nebenbedingungen, ist ausschlaggebend. Mittlerweile können infolge leistungsfähigerer Hardware, effizienter Simplexverfahren, neuer Ergebnisse in der Polyedertheorie und verbesserter Preprocessing-Techniken, Branch&Bound- und Branch&Cut-Verfahren sehr große MILP-Probleme in der Produktionsplanung mit zehntausenden von

aus verschiedenen Gebieten der Mathematik und Informatik wie z. B. Kombinatorik, Graphentheorie, Stochastik und Komplexitätstheorie verwendet; gemischt-ganzzahlig nichtlineare Probleme oder Fragen der globalen Optimalität werden hier jedoch kaum betrachtet.

[3] Beispiel: Lokalisation von Umschlagpunkten im Flugverkehr [205].

[4] Beispiel: Frequenzzuweisung in Mobilfunknetzwerken [79].

[5] Beispiel: Orthogonale Platzierungsprobleme [177], wie sie z. B. bei der Erstellung von Landkarten (Zuordnung der Städtenamen zu den Städten).

[6] Dies sind Variablen, die nur die Werte 0 und 1 annehmen können.

[7] Wenn nicht synonym zur ganzzahligen Optimierung gebraucht, versteht man unter *kombinatorischen Optimierungsproblemen* jene Probleme, bei denen erheblich mehr ganzzahlige Variablen als kontinuierliche Variablen vorhanden sind, die ganzzahligen Variablen nur wenige Werte annehmen können, die Struktur der Nebenbedingungen begrifflich einfach, aber mathematisch schwierig beschreibbar ist und sich das Problem eher graphentheoretisch formulieren lässt [121]. Eines der wichtigsten Probleme in der kombinatorischen Optimierung ist das Rundreiseproblem bzw. Problem des Handlungsreisenden [engl.: *traveling salesman problem*].

Binärvariablen, Millionen von kontinuierlichen Variablen und hunderttausenden von Nebenbedingungen in weniger als 15 Minuten auf PCs oder Workstations gelöst werden.

1.2 Zur Geschichte der gemischt-ganzzahligen Optimierung

Als früheste Vorläufer der gemischt-ganzzahligen Optimierung kann in einem gewissen Sinne die Beschäftigung mit diophantischen Gleichungen angesehen werden – und diese haben ihre Wurzeln in der Antike. Es ist bemerkenswert, dass Gomory, einer der Pioniere der ganzzahligen Optimierung, in seinen ersten Arbeiten versuchte, Methoden zur Lösung diophantischer Gleichungen auf Ungleichungssysteme zu übertragen, wie er in Gomory (2002,[113]) berichtet. Diese Vorgehensweise ist durchaus verständlich, wenngleich aus heutiger Sicht bekannt ist, dass dies nur bedingt nutzbringend ist.

Aus heutiger Sicht setzt die gemischt-ganzzahlige Optimierung zunächst aber voraus, dass man in der Lage ist, beschränkte Optimierungsprobleme zu lösen. Bereits im frühen 19. Jahrhundert spielten im Rahmen der klassischen Mechanik beschränkte Optimierungsprobleme eine Rolle und wurden mit Hilfe von Lagrange-Multiplikatoren gelöst. Aus dem Jahre 1823 liegt eine Arbeit von Fourier vor, die im Zusammenhang mit der linearen Programmierung steht. Als Geburtsjahr der linearen Optimierung moderner Prägung gilt jedoch meist das Jahr 1947, in dem Dantzig das Simplexverfahren entwickelte und auf ein Luftwaffen-Planungsmodell anwendete, wenngleich es auch schon vorher vereinzelt wissenschaftliche Arbeiten zu diesem Thema gab; siehe hierzu Seite 72.

Etwa zu dieser Zeit, während des zweiten Weltkrieges und in der direkten Nachkriegszeit, entwickelte sich der Begriff *Operations Research*. Die zu lösenden Optimierungsprobleme standen in engem Zusammenhang militärisch durchzuführender Operationen. Meist arbeiteten interdisziplinär zusammengesetzte Teams mit mathematisch-naturwissenschaftlichem Hintergrund gemeinsam an Problemstellungen; häufig fanden sich Physiker in diesen Gruppen. In den 50er Jahren wurden diese Techniken mehr und mehr auf zivile Probleme angewendet und eine Vielzahl von Algorithmen entwickelt. Schließlich begann in den 60er Jahren ein Prozess, für diese Verfahren wohldefinierte mathematische Grundlagen zu entwickeln. Es entstand der Begriff *Mathematischen Programmierung*[8] als Verallgemeinerung von Linearer Programmierung und ihrer Erweiterung auf nichtlineare, beschränkte Optimierungsprobleme, der heute passender durch den Begriff *mathematische Optimierung* abgelöst wird; die lineare Programmierung ist ein darin enthaltener Spezialfall. Die Optimierung wiederum kann als Teilgebiet einer seit den 80er Jahren praxis-orientierten Disziplin der Mathematik, dem *wissenschaftlichen Rechnen* [engl.: *Scientific Computing*] aufgefasst werden.

Das methodische Vorgehen der Disziplinen wissenschaftliches Rechnen oder Operations Research im engeren Sinne zur Lösung praktischer Probleme hohen Komplexitätsgrades besteht darin, ein gegebenes praktisches Problem [engl.: *real-world problem*] auf ein *mathematisches Modell* abzubilden, d. h. in der Sprache der Mathematik mit Hilfe von Variablen und Relationen zu formulieren, Techniken (*Algorithmen*) zur Lösung des mathematischen Modells bzw. Problems zu entwickeln, diese Algorithmen auf einer Maschine (*Computer*) zu implementieren und schließlich die Ergebnisse zu interpretieren.

[8] Zu den historischen Ursprüngen des Bezeichnungen *Lineare Programmierung*, *Mathematische Programmierung* und anderen Begriffen der mathematischen Optimierung lohnt es sich, die lesenswerte Arbeit von Dantzig (2002,[64]) zu konsultieren.

Bei einem linearen Programm sind sowohl die Zielfunktion als auch die Beschränkungen in linearer Form gegeben. Der Lösungsraum hat die Form eines Polyeders. Infolge der Konstanz des Gradienten liegt die Lösung in einer der Ecken. Das Simplexverfahren nach Dantzig nutzt diese Eigenschaft aus, indem es von Ecke zu Ecke iterativ die optimale Ecke sucht. In der Sprache der linearen Algebra wird das geometrische Element Ecke durch den Begriff der Basis ersetzt. Die wesentlichen algorithmischen Komponenten des Simplexverfahrens sind der Optimalitätstest, die Wahl einer aufzunehmenden Basisvariablen und die Elimination einer Basisvariablen, sowie der Pivotschritt, der geometrisch die Bewegung von einer Ecke zu einer benachbarten Ecke beschreibt.

In den 50er Jahren sind mit Bellman [31] die ersten Ansätze *dynamischer Programmierung* zu verzeichnen, Ford & Fulkerson (1956,[153]) beschäftigten sich mit *Netzwerkflussproblemen*, Kuhn und Tucker [188] formulierten 1951 die Grundlagen *nichtlinearer Optimierung*. Gomory [112] präsentierte 1958 einen ersten allgemeinen Algorithmus zur Lösung *ganzzahliger Optimierungsprobleme*. Dantzig & Wolfe (1960,[59]) untersuchten *Dekompositionsverfahren* und ihre Anwendung auf größere Probleme. Bereits 1954 wurde von Orchard-Hays von der Rand Corporation das erste kommerzielle Programm zur Lösung linearer Optimierungsprobleme geschrieben.

Heutzutage konkurrieren die besten Simplexverfahren mit den besten Innere-Punkte-Methoden, die auch schon in den 50er und 60er Jahren in der nichtlinearen Optimierung ihre Anfänge haben, deren Entwicklung aber mit Karmakar (1984,[171]) für die lineare Optimierung neue Impulse erhielt. Je nach Problemstruktur ist das eine oder andere Verfahren geeigneter. In manchen Fällen eignet sich eine Mischung beider Algorithmen zur Lösung von Problemen mit mehreren Millionen Unbekannten. Wesentliche Teilverfahren, die zu dieser Verbesserung der Algorithmen führten, sind neuerliche Entwicklungen des *Preprocessings*, des *Pricings* und der Faktorisierung. Diese Verfahren konnten aber auch nur deshalb bedeutend verbessert werden, weil die inzwischen ebenfalls verbesserten Rechnersysteme (Architektur, Speicher und Softwareumgebung) einfache und schnelle Tests zulassen und somit ein sinnvolles Experimentieren ermöglichen. Der Fortschritt innerhalb der linearen Programmierung hat einerseits Konsequenzen für die Größe und Komplexität der Problemstellungen, die damit in Angriff genommen werden können. Andererseits nimmt die lineare Programmierung innerhalb der Optimierung, speziell auch der gemischt-ganzzahligen linearen und nichtlinearen Optimierung eine Schlüsselstellung ein, da hier viele lineare Programme als Teilprobleme gelöst werden müssen.

Einer der grundlegenden Ideen zur Lösung gemischt-ganzzahliger Probleme ist nach wie vor ein Branch&Bound-Konzept, das auf einer LP-Relaxierung aufbaut. Die erzeugte Folge von LP-Problemen wird meist mit Hilfe des dualen Simplexverfahrens gelöst, da dieses die Einbeziehung einer zusätzlichen Nebenbedingung vergleichsweise effizient berücksichtigt. Da bei den Innere-Punkte-Methoden noch die algorithmischen Grundlagen für Warmstarts und Re-Optimierung in den Anfängen stecken, finden sie bei MILP-Problemen nur selten Anwendung. Mit Hilfe eines inzwischen automatisierten Preprocessings für MILP-Probleme lassen sich Redundanzen im Modell eliminieren, Variablen feste Werte zuweisen, falls möglich, Nebenbedingungen verschärfen und weitere Ungleichungen von logischen Relationen zwischen Ungleichungen ableiten, und sich somit schwierige MILP-Probleme, die allein mit Hilfe von Branch&Bound kaum gelöst werden können, in günstigen Fällen nun sogar ohne Branch&Bound lösen. Bei Branch&Cut-Verfahren [222] werden zu einem vorliegenden Teil-Problem während des Branch&Bound-Prozesses zusätzliche, zulässige Ungleichungen addiert, die es für eine vorliegende Variable unmöglich machen, bestimmte, nicht-ganzzahlige Werte anzunehmen. Der Vorteil

dieser Methode ist, dass stets nur lokal und dynamisch die LP-Relaxierung der konvexen Hülle angenähert wird. Mit der Entwicklung paralleler Algorithmen und Software zur Lösung industrieller Planungs- oder Scheduling-Probleme, hat die gemischt-ganzzahlige lineare Optimierung inzwischen Einzug in viele kommerzielle Produkte und professionelle Planungssysteme gefunden.

Die gemischt-ganzzahlige nichtlineare Optimierung ist seit etwa 1985 ein aktives und sich schnell entwickelndes Feld, das besonders von den Aktivitäten und Entwicklungsarbeiten der Chemieverfahrensingenieure um Ignatio Grossmann von der Carnegie Mellon Universität in Pittsburgh (USA) und seinen Schülern profitiert hat. Ein Überblick über ein breites Feld von Anwendungen aus den Bereichen Prozesssynthese und -design[9], Quantenchemie, Standortplanung, Planung und Scheduling, Stromerzeugung und Finanzdienstleistung gibt Floudas (2000,[89], Kapitel 20). Inzwischen existieren auch für gemischt-ganzzahlige nichtlineare Probleme stabile kommerzielle Lösungsalgorithmen, meist allerdings nur für konvexe MINLP-Probleme.

Besondere Forschungsbemühungen gelten daher der Globalen Optimierung, die in der Lage ist, auch bei nichtkonvexen MINLP-Problemen das globale Optimum zu bestimmen. Dem interessierten Leser seien hierzu die Bücher [89], [137], [175], und [136] als Einstiegslektüre empfohlen; das *Journal of Global Optimization* zeigt die neuesten Entwicklungen auf diesem Gebiet auf. Auch hier gibt es mittlerweile gute Software, die von einigen Universitäten entwickelt wurde – insbesondere die von Floudas und seiner Schülerin Adjiman in Princeton (USA) entwickelten Programme $\texttt{SMIN}-\alpha\texttt{BB}$ ([89], Kapitel 21) und $\texttt{GMIN}-\alpha\texttt{BB}$ ([89], Kapitel 22). Neuere Forschungsarbeiten in Princeton – siehe http://titan.princeton.edu – und dem Imperial College in London (England) ermöglichen es, diese Ansätze auch zur Lösung von Parameterschätzproblemen, Optimierungsproblemen, die zusätzlich noch Differentialgleichungen enthalten, und Optimale-Steuerungsproblemen zu verwenden. Ein weiteres Aktivitätszentrum der Globalen Optimierung ist die Arbeitsgruppe um Neumaier an der Universität Wien; hier wurde auch das Programm `GLOBT` [57] entwickelt und hier residiert eine der besten und informativsten Webseiten zum Thema Globale Optimierung und Optimierung überhaupt: http://www.univie.ac.at/~neum – ein Besuch lohnt sich. Effiziente B&B-Verfahren wurden speziell für nichtkonvexe MINLP-Probleme entwickelt und haben in Softwarepaketen wie `GAMS` Einzug gehalten. Allerdings muss bemerkt werden, dass im nichtkonvexen gemischt-ganzzahligen Umfeld die Problemgröße meist bestenfalls auf einige hundert Variablen beschränkt ist; mit den Programmen $\alpha\texttt{BB}$ [7] und `BARON` [101] sind zwar auch schon Probleme mit über 1000 Variablen gelöst worden, aber die meisten dieser Variablen traten nur in linearen Relationen auf.

Einen sehr lesenswerten Überblick über die dem Operations Research zugeordneten Aspekte der gemischt-ganzzahligen Optimierung im Laufe der vergangenen 50 Jahre geben einige Hauptakteure dieser Disziplin anlässlich des 50jährigen Jubiläums (Band 50) der Zeitschrift *Operations Research*.

[9] In der chemischen Verfahrenstechnik treten MINLP- und Globale Optimierungsprobleme in vielen Bereichen auf, z. B. bei Misch- oder Trennungsproblemen, insbesondere bei Destillation und Extraktion, der Analyse der Stabilität von Phasen, oder auch der Parameterschätzung. In der Prozessoptimierung beispielsweise ist es aus Sicherheitsgründen nicht ausreichend, lediglich lokale Minima zu bestimmen, sondern es muss mit Sicherheit festgestellt werden, dass man das globale Minimum erhalten hat.

1.3 Zur Bedeutung von Modellen

Der Begriff der Modellierung bzw. Modellbildung setzt den Begriff des Modells voraus, welcher sich zunächst aus dem lateinischen Wort *modellus* (Maßstab [Diminutiv von Modus, Maß] und dann aus dem im 16. Jahrhundert gebildeten Wort *modello* ableitet und der heutzutage im Alltag und in der Wissenschaftssprache in verschiedener Bedeutung verwendet wird, meist jedoch im Sinne einer vereinfachten, abstrahierten oder strukturierten Abbildung eines interessierenden Realitätsausschnittes. Die Idee und damit die Begrifflichkeit ist aber schon wesentlich älter. Die klassische Geometrie unterscheidet schon mit Pythagoras um 600 v. Chr. ganz bewusst zwischen Rad und Kreis; Acker und Rechteck. Um 1100 n. Chr. wurde ein Holzmodell als Abstraktion des späteren Speyerer Doms gefertigt. Astrolabien und Himmelskugeln wurden als manifestierte Modelle der Bewegungsabläufe am Firmament zur Berechnung von Aufgang und Untergang von Sonne, Mond und Sterne verwendet. Bis zum 19. Jahrhundert wurden mechanische Modelle als Abbild der Wirklichkeit verstanden; sie führten alle Vorgänge auf die Bewegung kleinster Teilchen zurück, die den Prinzipien der klassischen Mechanik folgten. Weitergehend wurde versucht, sämtliche physikalische und auch sonstige Vorgänge auf das mechanistische Modell zu reduzieren. Heutzutage spricht man in der Physik und den mathematischen Wissenschaften von Modellen, wenn man

- aus Gründen der Vereinfachung die Untersuchung auf bestimmte, jeweils als wichtig betrachtete Phänomene in einem Bereich, beschränkt (*Beispiel*: bei der Bewegung der Planeten wird zunächst die räumliche Ausdehnung dieser Körper vernachlässigt),
- aus Gründen didaktischer Veranschaulichung ein auf klassischen Vorstellungen beruhendes Bild für anschaulich nicht zugängliche Phänomene angibt (*Beispiel*: das Planetenmodell zur Veranschaulichung der Verhältnisse im Atomkern),
- die Verhältnisse in einem Bereich in Analogie zu bekannten Verhältnissen in einem anderen Bereich studiert (Analogiemodelle).

Speziell von einem mathematischen Modell eines Prozesses oder Problems spricht man, wenn dieser mit Hilfe der Mathematik eigenen Objekten (Variablen, Terme, Relationen) formuliert wird. Ein (mathematisches) Modell repräsentiert ein reales Problem in der Sprache der Mathematik, d. h. unter Verwendung von mathematischen Symbolen, Variablen, Gleichungen, Ungleichungen und anderen Relationen.

Ein sehr wichtiger Aspekt, der mit der Modellbildung verbunden ist und ihr vorausgeht, ist der *Modellzweck*. Er ergibt sich unmittelbar im Zusammenhang mit der Problemstellung und beeinflusst sehr wesentlich den Prozess der Modellbildung. In den *Naturwissenschaften* Physik, Astronomie, Chemie und Biologie dienen Modelle dazu, ein tieferes Verständnis der in der Natur ablaufenden Prozesse zu erlangen (erkenntnistheoretisches Argument). Der Vergleich von Messungen und Beobachtungen mit den von einem Modell vorausgesagten Ergebnissen erlaubt Aussagen über Zulässigkeitsbereich und Qualität des Modells. Diese Aspekte und Fragen der Etablierung und Weiterentwicklung naturwissenschaftlicher Modelle werden in der philosophischen *Wissenschaftstheorie* diskutiert. Karl R. Popper hat in seinem Werk „*Logik der Forschung*" [236] auf die Asymmetrie von Verifikation und Falsifikation (auf Grund der logischen Unmöglichkeit von Induktionsschlüssen in den empirischen Wissenschaften) hingewiesen. In den Disziplinen „*Wissenschaftliches*

Rechnen" und „*Operations Research*" kommt den Modellen oft eine eher lokale Bedeutung zu; ein spezieller Ausschnitt der Realität[10] wird sehr detailliert abgebildet.

Motivation für die Modellentwicklung sind also auf der einen Seite erkenntnistheoretische, auf der anderen Seite aber auch militärische, pragmatische und kommerzielle Gesichtspunkte, aus denen sich oft operative Handlungsanweisungen ableiten lassen. Im einzelnen finden wir die folgenden, eher dem Verständnisgewinn[11] in einer bestimmten Situation oder eines Systems gewidmeten Modellzwecke:

- Handhabung nicht direkt zugänglicher räumlicher oder zeitlicher Größenordnungen (sehr langsame oder sehr schnelle Prozesse, großräumige Prozesse);
- Vermeidung schwieriger, teurer und/oder gefährlicher Experimente;
- Analyse von Fallbeispielen und „*Was-Wäre-Wenn Szenarien*".

Desweiteren sehen wir Modelle, die gerade im Umfeld der gemischt-ganzzahligen Optimierung in der Industrie entwickelt werden, mit den Zielen

- unerwünschte Nebenprodukte zu vermeiden,
- Kosten zu minimieren,
- Gewinne zu maximieren, oder
- Ressourcen zu schonen

um nur einige typische Ziele zu nennen.

Wie gelangt man nun von einem gegebenen Problem zu einem mathematischem Modell? Der damit verbundene Prozess der Modellbildung ist keineswegs einfach oder eindeutig; er und seine Schwierigkeiten ähneln ein wenig der Lösung von Textaufgaben [235] in der Schule, allerdings hatte man da stets den Vorteil, dass von den Lehrinhalten her bekannt war, welche Mathematik zur Lösung des anstehenden Problems die *richtige* war. In der Praxis weiß man jedoch meist nicht, welche Mathematik die passende ist. Daher ist es wichtig, bei den ersten herantastenden Versuchen, das Problem zu verstehen und zu lösen, sehr offen und kreativ zu sein. Es ist hilfreich, so viele Problemlösungsstrategien und -heuristiken [211] wie möglich zu kennen und sich dieser bedienen zu können, oder daraus neue zu entwickeln. Für die Bildung von Modellen gilt:

- es gibt kein Patentrezept zur Bildung von Modellen,
- Erfahrung und Heuristik kommt große Bedeutung zu,
- es gibt kein Modell, das eindeutig das richtige wäre,

[10] Die Realität an sich ist schon so kompakt wie möglich, um sich selbst verlustfrei darstellen zu können. Folglich ist jede Selbstabbildung auf einen kleinen Ausschnitt (einzelner Mensch) unweigerlich mit Informationsverlust verbunden. Wir prägen uns und unser Weltbild also immer nur durch einen sehr kleinen Realitätsausschnitt, der durch Sinnesorgane und erlerntes Verhalten stets gefiltert wird. Unsere Begriffe und Sprachen überhaupt sind Produkte eingeschränkter Wahrnehmung - und daher höchst subjektiv. Modellbildung geschieht dann als Reduzierung in schon ohnehin beschränkten Weltbildern.

[11] Der Verständnisgewinn mag natürlich wiederum selbst einem höheren Zweck unterworfen sein.

- ein Nebeneinander von Modellen kann zur Untersuchung unterschiedlicher Aspekte berechtigt sein (*Beispiel*: Welle-Teilchen-Dualismus in der Physik). Zu beachten ist, dass unterschiedliche Modelle zur Beschreibung desselben Sachverhaltes unter Umständen auf sehr verschiedene Datenbasen aufbauen können.
- die Modellentwicklung sollte dem Grundsatz „So einfach wie möglich, so kompliziert wie nötig." oder den Worten Antoine de Saint-Exupérys: „Ein Entwurf ist nicht dann fertig, wenn Du nichts mehr hinzufügen kannst, sondern dann, wenn Du nichts mehr weglassen kannst." folgen.

Dennoch gibt es einige Aspekte, die bei der Modellentwicklung beachtet werden sollten:

- *Definition der Problemstellung und des Modellzwecks*: Eine klar formulierte Aufgabenstellung sollte als Grundlage für die Definition des Modellzwecks verwendet werden.
- *Problemabgrenzung*: Es ist klar zu definieren, was zum Problem bzw. Modell gehört und was nicht (*Beispiel*: ein Produktionsplanungssystem für die mittel- und langfristige Planung darf auf Feinplanungsaspekte verzichten).
- *Identifizierung wichtiger Modellobjekte* (*Beispiele*: Produkte, Produktionsstandorte, Lager, Transportverbindungen, etc.) und verbale Erfassung der Relationen zwischen diesen Objekten.
- *Akzeptanz und Nutzer*: Die Modellformulierung sollte der Problemstellung, dem Modellzweck und dem potentiellen Nutzer angepasst sein. Problematisch und der Akzeptanz eines Optimierungsmodell nicht gerade förderlich sind Fälle, in denen der Modellierer nicht alle Zusammenhänge kennt. Routenplanungsmodelle und deren Einsatz können bei den Lastwagenfahrern auf Akzeptanzprobleme stoßen, wenn dem Modellierer vom Auftraggeber, dem Routenplaner, die Minimierung der Fahrtkosten als Ziel genannt wurde, jeder einzelne Lastwagenfahrer aber prozentual am Lieferwert beteiligt ist. Ein Fahrer wird also lieber eine Fahrt mit einem teuren Möbelstück unternehmen, bei der er lediglich dieses Möbelstück aufstellen muss, als eine Tour mit vielen Zielen, an denen nur billige und vielleicht sogar noch zeitintensive Aufstellarbeiten zu leisten sind.

Was leistet ein Modell? Ein sorgfältig formuliertes und dokumentiertes Modell erlaubt *Deduktion*, d. h. aus dem Modell lassen sich Konsequenzen und Ergebnisse ableiten und eindeutig nachvollziehen. Gleichsam bietet ein Modell die Möglichkeit der Interpretation, d. h. eine sinnvolle Übersetzung der Modellergebnisse und möglicherweise Konsequenzen für die Realität. Die Modellbildung führt zu Erkenntnisgewinn; das Modell repräsentiert dokumentiertes und nachvollziehbares Wissen und Verständnis. An seinem Realitätsgrad und seiner Leistungsfähigkeit wird sich die *Modellbewertung* orientieren. Während innerhalb der Mathematik eine strenge Beweistechnik der Maßstab ist, spricht man bei naturwissenschaftlichen Modellen von Falsifikation und Verifikation und bei ökonomischen oder OR-orientierten Modellen lediglich von *Validierung*, wobei die Ordnungsrelation[12]

$$\boxed{\text{Validierung} \leq \text{Falsifikation/Verifikation} \leq \text{Mathematischer Beweis}}$$

[12] Bei der Interpretation dieser Ungleichung ist zu beachten, dass Falsifikation/Verifikation als Überprüfung von Modellergebnis an der Wirklichkeit gedacht ist, während der mathematische Beweis sich auf die vorausgesetzte Modellwelt samt Axiomen und Regeln beschränkt.

erfüllt ist. Man sollte sich jedoch auch bei einem gut validierten Modell stets bewusst sein, dass es sich bei einem Modell möglicherweise um ein beschränktes Abbild der Realität handelt, das Modell somit *weniger* als die Realität ist. Es muss allerdings nicht zwingend weniger sein. So bildet zum Beispiel die Elektrodynamik makroskopische elektromagnetische Phänomene so genau ab, dass man im Rahmen der Messgenauigkeit von Übereinstimmung zwischen Modell und Realität sprechen kann.

1.4 Die Kunst der Modellierung

Zwar kann eine exakte Abbildung zwischen dem ursprünglichen Problem und dem Modell das Ziel der Modellierung sein, aber in manchen praktischen Fällen werden einige Aspekte des Problems zu vereinfachen sein, um ein rechenbares Modell zu erhalten. Diese Vereinfachungen sollten aber nicht nur als ein notwendiges Übel angesehen werden, sondern haben eine eigene Berechtigung, wie die drei folgenden Beispiele zeigen:

1. Zu Beginn der Modellierung kann es sinnvoll sein, mit einer „verkleinerten" Version des Problems zu arbeiten, um ein Gefühl für die Struktur und Abhängigkeiten des Modells zu gewinnen. Auf dieser Basis kann sich insbesondere ein konstruktiver Austausch zwischen dem Kunden und dem Modellierer entwickeln. So kann z. B. ein Fuhrpark mit 100 Fahrzeugen und 12 Depots zunächst vereinfacht mit 10 Fahrzeugen und 2 Depots untersucht werden; die *reale Welt* und die *Modellwelt* können so in einer Folge von Diskussionen aufeinander zuwachsen.
2. In Teil- oder Untermodellen lässt sich ein Verständnis für bestimmte Aspekte des Problems entwickeln, die für die Lösung des Gesamtproblems wieder sinnvoll genutzt werden können.
3. Teile des realen Problems können zu schwierig sein, um sie vollständig und komplett abzubilden. Es bleibt während der Modellierung zu klären, ob diese Teilaspekte vernachlässigt oder approximiert werden können, oder ob sie essentiell sind.

In jedem Falle ist es natürlich wichtig, dass die vorgeschlagenen Vereinfachungen klar verstanden und dokumentiert sind.

Zur Modellierung und Lösung eines Problem mit Hilfe der mathematischen Optimierung sind die drei Hauptobjekte zu identifizieren:

- Variablen [engl.: *variables*];
- Nebenbedingungen [engl.: *constraints*];
- Zielfunktion [engl.: *objective function*].

Diese Objekte basieren in größeren Modellen auf einer vorherigen Klärung der *Datenstruktur* und *der zu verwendenden Indizes*. Zur Betonung der Bedeutung der Datenstruktur und der Indizes und auch im Hinblick auf eine gute Modellierungspraxis sei daher nachstehend tiefer auf diese Aspekte eingegangen.

1.5 Variablen, Indizes und Indexmengen

Die Entscheidungsunterstützung, die von einem mathematischen Modell erwartet wird, wird zum Teil direkt durch die Variablen geleistet, manchmal auch Entscheidungsvariablen[13] [engl.: *decision variables*] genannt, zum anderen durch aus den Variablen abgeleitete Entscheidungen. In einem Modell können wir mit Hilfe von Variablen die Fragen *wie viele*, *welcher Art*, *woher*, *wieviel* usw. modellieren. Im algebraischen Sinne sind die Variablen die *Unbekannten* in einem Modell, und typischerweise werden Symbole wie x, y und z zur Bezeichnung dieser Variablen verwendet.

Zur Klärung des Unterschieds zwischen einer Variablen bzw. einer Entscheidung und einer abgeleiteten Entscheidung sei das folgende Beispiel betrachtet: ein Leiter einer Logistikabteilung muss entscheiden, wie viele LKWs in festen 5-Minutenintervallen losfahren müssen; der erste muss um 8^{30} Uhr das Werk verlassen. Die Anzahl der Fahrzeuge und Abfahrt des letzten LKWs sind also zunächst unbekannt. Sobald aber die Entscheidung hinsichtlich der Anzahl der LKWs oder diejenige nach der letzten Abfahrtszeit getroffen ist, kann die andere Entscheidung abgeleitet werden. Als Variable könnten wir in diesem Falle also die Anzahl der LKWs, aber auch die letzte Abfahrtszeit wählen; hier ist es aber einfacher, die Anzahl der LKWs zu wählen, da diese Information wahrscheinlich in einer Kostenfunktion direkt benötigt wird. Die primäre Entscheidung, eine bestimmte Menge eines Produktes einem Lager zu entnehmen oder zuzuführen und die Variable, die dies misst, ist ein weiteres Beispiel im Zusammenspiel mit der sich aus der entsprechenden Lagerbilanzgleichung ergebenden Lagervariablen, die lediglich misst, wieviel Material im Lager ist und nicht als Entscheidung angesehen wird.

Festzuhalten ist, dass die Identifikation wichtiger Entscheidungen der Schlüssel zur Wahl geeigneter Variablen ist. Während die Entscheidungen eher zur Welt des Problemstellers gehören, fallen die Variablen in den Zuständigkeitsbereich des Modellierers, wobei die folgenden Variablentypen unterschieden werden können:

1. *Kontinuierliche Variablen*, $0 \leq X^- \leq x \leq X^+$; dies sind nichtnegative Variablen, die jeden reellen (wegen der begrenzten Darstellungsmöglichkeit von Zahlen, rationalen) Wert zwischen einer unteren Schranke X^-, in vielen Fällen ist $X^- = 0$, und einer oberen Schranke X^+, hier ist der Wert unendlich möglich, annehmen können und die in diesem Buch mit kleinen Buchstaben bezeichnet werden – sie können z. B. die Menge Öl beschreiben, die je Zeiteinheit durch eine Rohrleitung fließt.

2. *Freie Variablen*, auch unbeschränkte Variablen; dies sind (kontinuierliche) Variablen, die auch negative Werte annehmen dürfen, also nicht im Vorzeichen beschränkt sind. Freie Variablen können Wachstum, Änderung oder einen Kontostand beschreiben und lassen sich äquivalent als Differenz zweier nichtnegativer Variablen $x^+ - x^-$ ausdrücken. Auf Seite 255 werden sie z. B. als Hilfsvariablen bei der Lösung spezieller nichtlinearer Optimierungsprobleme mit Hilfe von SLP-Verfahren benötigt.

3. *Ganzzahlige Variablen*, $\alpha, \beta, \ldots \in \{0, 1, 2, 3, 4, \ldots\}$; auch hier gilt $X^- \leq \alpha \leq X^+$, aber diese Variablen können nur ganzzahlige Werte annehmen, werden im Buch durch kleine griechische Buchstaben bezeichnet und zählen z. B. die Anzahl von Objekten.

[13] Im Umfeld der linearen Programmierung werden die Variablen auch manchmal *Aktivitäten* oder *Spalten* [engl.: *columns*] genannt.

4. *Binäre Variablen* können nur die Werte 0 und 1 annehmen und sind somit ein Spezialfall der ganzzahligen Variablen mit $X^- = 0$ und $X^+ = 1$. Sie eignen sich besonders zur Beschreibung von *ja-nein*-Entscheidungen und damit auch zur Modellierung logischer Zustände.

5. *Diskrete Variablen* können nur bestimmte isolierte bzw. diskrete Werte annehmen, d. h. $\sigma \in \{W_1, W_2, \ldots, W_n\}$. Die Werte $W_1, W_2, \ldots, W_n$ müssen dabei nicht unbedingt ganzzahlig oder binär sein; denkbar ist z. B. $\sigma \in \{-3, 0, +0.25, +1, +17.1\}$. Diskrete Variablen enthalten ganzzahlige Variablen als Spezialfälle und können mit Hilfe binärer Variablen δ_i, der Gleichung $\sigma := \sum_{i=1}^{n} W_i \delta_i$ und der Nebenbedingung $\sum_{i=1}^{n} \delta_i = 1$ oder den nachstehend beschriebenen SOS-1-Mengen beschrieben werden.

6. *Strukturierte Variablenmengen* [engl.: *special ordered sets*]; hierzu werden K nichtnegative, reellwertige Variablen λ_k betrachtet, die eine der folgenden Bedingungen genügen:

   - SOS-1-Mengen: es existiert genau ein Index k mit $\lambda_k > 0$; wie in Abschnitt 5.7 ausführlich dargelegt, eignen sich derartige Variablenmengen zur Selektion von Kapazitäten oder Auswahl einzelner Elemente einer Menge und allgemein zur Beschreibung diskreter Variablen.
   - SOS-2-Mengen: es existieren maximal zwei Indizes k_1, k_2 mit $\lambda_{k_1} > 0$, $\lambda_{k_2} > 0$ und für diese gilt $|k_1 - k_2| = 1$; in Abschnitt 5.7 wird gezeigt, wie mit SOS-2-Variablen spezielle nichtlineare Zusammenhänge modelliert werden können.

   Standardmäßig wird von den SOS-2-, häufig auch von SOS-1-Variablen die Erfüllung der Konvexitätsbedingung

   $$\sum_{k=1}^{K} \lambda_k = 1 \quad , \quad \lambda_k \geq 0$$

   gefordert.

7. Halbstetige bzw. semi-kontinuierliche [engl.: *semi-continuous*] und partiell-ganzzahlige Variablen; *semi-kontinuierliche Variablen* sind durch die Bedingung

   $$x = 0 \ \vee \ X^- \leq x \leq X^+$$

   definiert; der oberen Grenze X^+ kommt dabei oft nur eine mathematisch-technische Bedeutung zu und sie muss nicht notwendigerweise im Modell auftreten. Mit Hilfe halbstetiger Variablen lassen sich effizient Mindestproduktions- oder Mindesttransportmengen beschreiben, z. B. „ein Transport ist nur möglich, wenn mindestens 20 Tonnen verschifft werden können"; siehe auch Abschnitt 10.2.1.3, wo halbstetige Variablen Strömungsraten beschreiben. Eine Verallgemeinerung dieses Variablentyps sind *partiell-ganzzahlige Variablen*, die durch die Bedingung

   $$x = 0, 1, ..., N \ \vee \ N = X^- \leq x \leq X^+$$

   charakterisiert sind. Halbstetige und partiell-ganzzahlige Variablen können äquivalent durch binäre Variablen und passende Ungleichungen ersetzt werden. Es ist jedoch numerisch effizienter, wenn diese Variablenstrukturen, wie in Abschnitt 5.13.4 beschrieben, unmittelbar mit in den Branch&Bound-Prozess einbezogen werden.

In der Regel werden Variablen mit einem einzigen Symbol bezeichnet; damit ist z. B. sichergestellt, dass mit xy wie in der Mathematik üblich das Produkt „x mal y" gemeint ist. Diverse Softwarepakete, die eine Modellierungssprache enthalten, erlauben auch Variablennamen der Form $xcash$, $prod1$, $prod2$; dies kann manchmal hilfreich sein, macht es aber schwieriger, die mathematischen Relationen zu lesen.

In diesem Abschnitt wurden die Variablen soweit als individuelle Objekte betrachtet. In größeren Modellen werden die Variablen jedoch eher in Kollektiven auftreten und von Indizes und Indexmengen abhängen.

Wenn Produktionsmengen etwa in z. B. 12 verschiedenen Zeitintervallen und 5 verschiedenen Standorten modelliert werden sollen, wäre es ziemlich unpraktisch, dafür 60 individuelle Variablen mit Namen $x01, \ldots, x60$ einzuführen und sich zu merken, welcher Wert zu welcher Kombination entspricht. Wesentlich einfacher ist es, die Variable x als generische Variable einzuführen, die von Zeitscheibe und Standort abhängt. Hierzu wird der Index $t = 1, 2, .., 12$ für die Zeit und der Index $s = 1, 2, 3, 4, 5$ für die Standorte verwendet. Genauer wird t als Index und die Menge $\mathcal{T} = \{1, 2, ..., 12\}$ als Indexmenge bezeichnet, mit s und $\mathcal{S} = \{1, 2, ..., 5\}$ wird entsprechend verfahren. Nun lässt sich x_{ts} als die zur Zeit t am Standort s produzierte Menge einführen, also z. B. x_{45} für die Produktionsmenge im Zeitintervall 4 am Standort 5; allgemein

$$x_{ts} \quad , \quad t \in \mathcal{T} \quad , \quad s \in \mathcal{S} \quad .$$

Dies ist eine alternative Formulierung zu

$$x_{ts} \quad ; \quad t = 1, 2, ..., 12 \quad ; \quad s = 1, 2, ..., 5 \quad .$$

Wenn klar ist, dass alle Elemente einer Indexmenge gemeint sind, schreiben wir dies mit dem Allquantor $\forall$ (für alle) und fordern z. B.

$$x_{ts} \geq 10 \quad , \quad \forall t \quad , \quad \forall s$$

bzw.

$$x_{ts} \geq 10 \quad , \quad \forall \{ts\} \quad .$$

Alternativ zu dieser Schreibweise findet sich auch manchmal die etwas umständlichere Konvention $x(t, s)$, die allerdings einer Computerimplementierung näher kommt. Vielleicht setzt sich in der nahen Zukunft die bereits von `Maple`[14] und `MathCAD`[15] verwendete zweidimensionale, kürzere Schreibweise durch, in der Indizes tatsächlich als tiefgestellte Indizes erscheinen. Natürlich kann eine Variable auch von mehr als zwei Indizes abhängen, so bezeichnet z. B. p^P_{srpt} die am Standort s, auf dem Reaktor r produzierte Menge an Produkt p im Zeitinterval t; das hochgestellte P dient als weiteres Attribut, das die Produktionsvariable p^P z. B. von der Einkaufsvariable p^E unterscheidet.

Indexmengen ermöglichen es, leicht Modellinstanzen geringer Größe für Testzwecke zu erzeugen, indem man nur einen Teil der Indizes verwendet, z. B. statt $t = 1, 2, \ldots, 12$ nur $t = 1, 2, 3$. Statt numerischer Indexmengen wie $\mathcal{S} = \{1, 2, ..., 5\}$ können auch alphanumerische Indexmengen, z. B. $\mathcal{S} = \{\text{Athen}, \text{Rom}, \text{Paris}, \text{London}, \text{Bonn}\}$ verwendet werden. Die Verwendung diskreter Zeitscheiben (Tage, Wochen, Monate, Quartale oder Jahre) führt zu sogenannten *zeitdiskreten Modellen*. Insbesondere bei Schedulingproblemen sind auch *zeitkontinuierliche Modellformulierungen*, wie sie z. B. in den Arbeiten [143], [142],

[14] `Maple` ist ein eingetragenes Markenzeichen der Firma Waterloo Maple Inc.
[15] `MathCAD` ist ein eingetragenes Markenzeichen der Firma MathSoft, Inc.

[144], [127], [194] und [196] von Floudas und Mitarbeitern entwickelt werden, vielversprechende Alternativen. In diesen Modellen können die Start- und Endzeitpunkte beliebige Werte auf dem kontinuierlichen Zeitstrahl annehmen.

1.6 Nebenbedingungen, Beschränkungen, Constraints

Der im deutschen Sprachgebrauch häufig verwendete Begriff *Nebenbedingungen* ist leider nicht so griffig und auch nicht so allgemein akzeptiert, wie der in der angelsächsischen Literatur verwendete Begriff *Constraints*. Insbesondere wirft er die Frage nach den *Hauptbedingungen* auf; als solche könnte die Zielfunktion oder möglicherweise mehrere Zielkriterien angesehen werden, allerdings taucht in der Praxis der Begriff Hauptbedingungen nicht häufig auf. Synonym zum Ausdruck Nebenbedingungen findet man in der Literatur auch die Begriffe, *Randbedingungen*, der eigentlich passender für das Umfeld von Differentialgleichungen erscheint, *Zulässigkeitsbedingungen*, *Beschränkungen*, *Restriktionen* oder den englischen Ausdruck *Constraints* selbst. Für die Wahl des Ausdrucks *Zulässigkeitsbedingungen* spricht, dass die *Constraints* implizit die Menge der zulässigen Punkte, kurz *zulässiger Bereich*, [engl.: *feasible set*] des Optimierungsproblems beschreiben.

In der mathematischen Optimierung betrachten wir als Zulässigkeits- oder Nebenbedingungen algebraische Ausdrücke der Form

$$\text{(linke Seite)} \ \circ \ \text{(rechte Seite)}$$

wobei $\circ$ für eine der Relationen $\{\leq, =, \geq\}$ steht; die Relationen $<$ und $>$ müssen, wie auf Seite 185 ausgeführt, mit geeigneten kleinen Parametern auf die Relationen $\leq$ und $\geq$ zurückgeführt werden. In der linearen Optimierung können auf der linken und rechten Seite nur lineare Ausdrücke, d. h. Terme der Form

$$2x + 3.5y - 8.1w \ldots \quad ,$$

auftreten, bei denen jede Variable mit einem konstanten Koeffizienten multipliziert wird. In der nichtlinearen Optimierung sind auch Produkte, Potenzen oder Funktionen der Variablen wie e^x erlaubt. Eine spezielle Form von Nebenbedingungen liegt vor, wenn sie nur eine Variable enthalten, die mit Koeffizienten 1 auftritt, z. B. $x \leq 7$; derartige Bedingungen werden *Schranken* [engl.: *bounds*] genannt.

1.7 Die Zielfunktion

Die Zielfunktion ist die treibende Kraft in einem Optimierungsproblem. Je nach dem, wie diese gewählt wird, kann jeder Punkt eines LP-Problems, der auf dem Rand des zulässigen Bereichs liegt, optimale Lösung werden, bei nichtlinearen Problem sogar jeder Punkt des zulässigen Bereichs; auf Seite 310 wird die Bedeutung sinnvoller und ganzheitlicher Zielfunktionen betont. In praktischen Problemen wird als Zielfunktion häufig Gewinnmaximierung oder Kostenminimierung betrachtet. Manchmal hat ein Probleminhaber aber auch mehrere Zielfunktionen oder besser Zielkriterien im Sinn, von denen einige maximiert, andere minimiert werden sollen; derartige Fragestellungen werden im Abschnitt 6.4 behandelt.

Zwei Zielfunktionen bedürfen der besonderen Aufmerksamkeit: *minimax* und *maximin*. Betrachtet man die Fertigstellungszeiten t_i der Prozesse $i \in \mathcal{I}$ und soll die Gesamtfertigstellungszeit [engl.: *makespan*] für alle Prozesse minimiert werden, so führt dies auf ein typisches *minimax*-Problem

$$\min \left\{ \max_{i \in \mathcal{I}} \{t_i\} \right\} \quad .$$

Diese Zielfunktion kann mit Hilfe der zusätzlichen Ungleichungen $t_i \leq t$, $\forall i \in \mathcal{I}$, und der zu minimierenden Zielfunktion min t als lineares Problem beschrieben werden. In ähnlicher Weise kann

$$\max \left\{ \min_{i \in \mathcal{I}} \{t_i\} \right\}$$

mit Hilfe der Ungleichungen $t \leq t_i$, $\forall i \in \mathcal{I}$, und der Zielfunktion max t als lineares Problem formuliert werden.

1.8 Definition gemischt-ganzzahliger Optimierungsprobleme

Mit den eingeführten Begriffen sind wir nun in der Lage, eine formale Definition eines ganzzahligen bzw. gemischt-ganzzahligen Optimierungsproblems zu geben. Wir Für gegebene Vektoren $\mathbf{x}^{\mathrm{T}} = (x_1, ..., x_{n_c})$ und $\mathbf{y}^{\mathrm{T}} = (y_1, ..., y_{n_d})$, die n_c kontinuierliche und n_d ganzzahlige Variablen repräsentieren, der zusammenhängenden Teilmenge $X \subseteq \mathbb{R}^{n_c}$, der diskreten und beschränkten Teilmenge $V \subseteq \mathbb{Z}^{n_d}$, den zusammengesetzten Vektor $\mathbf{x}_{\oplus}^{\mathrm{T}} = \mathbf{x}^{\mathrm{T}} \oplus \mathbf{y}^{\mathrm{T}}$, eine Zielfunktion $f(\mathbf{x}, \mathbf{y})$, n_e Gleichungsbedingungen $\mathbf{h}(\mathbf{x}, \mathbf{y})$ und n_i Ungleichungsbedingungen $\mathbf{g}(\mathbf{x}, \mathbf{y})$, heißt das Optimierungsproblem[16]

$$\min \quad \left\{ f(\mathbf{x}, \mathbf{y}) \;\middle|\; \begin{array}{l} \mathbf{h}(\mathbf{x}, \mathbf{y}) = 0 \\ \mathbf{g}(\mathbf{x}, \mathbf{y}) \geq 0 \end{array} \quad , \quad \begin{array}{l} \mathbf{h} : X \times V \to \mathbb{R}^{n_e} \\ \mathbf{g} : X \times V \to \mathbb{Z}^{n_i} \end{array} \quad , \quad \begin{array}{l} \mathbf{x} \in X \subseteq \mathbb{R}^{n_c} \\ \mathbf{y} \in V \subseteq \mathbb{Z}^{n_d} \end{array} \right\} \tag{1.8.1}$$

gemischt-ganzzahliges nichtlineares Optimierungsproblem (MINLP), wenn zumindest eine der Funktionen $f(\mathbf{x}, \mathbf{y})$, $\mathbf{g}(\mathbf{x}, \mathbf{y})$ oder $\mathbf{h}(\mathbf{x}, \mathbf{y})$ nichtlinear und zudem $n_c > 0$ und $n_d > 0$ ist; mit $n_c = 0$ und $n_d > 0$ heißt (1.8.1) *ganzzahliges nichtlineares Optimierungsproblem*. Die Vektorungleichung $\mathbf{g}(\mathbf{x}, \mathbf{y}) \geq 0$ soll dabei komponentenweise gelesen werden. Ist statt $\mathbf{y} \in V \subseteq \mathbb{Z}^{n_d}$ der Grundbereich $\mathbf{y} \in Y$ irgendeine Menge mit diskreten Elementen, z. B. $Y = \{-3, 0, +0.25, +1, +17.1\}$, so handelt es sich bei (1.8.1) um ein *diskretes Optimierungsproblem*, das wie auf Seite 11 mit Hilfe von binären Variablen wiederum als ganzzahliges Optimierungsproblem formuliert werden kann. Jeder Vektor $\mathbf{x}_{\oplus}^{\mathrm{T}}$, der den Nebenbedingungen in (1.8.1) genügt, heißt *zulässiger Punkt*[17] von (1.8.1); die Menge aller zulässigen Punkte bildet den *zulässigen Bereich* S. Das Vektorpaar $(\mathbf{x}, \mathbf{y})$ heißt *optimale Lösung*, wenn es zulässig ist und für alle zulässigen Punkte $(\mathbf{x}', \mathbf{y}') \in S$ gilt: $f(\mathbf{x}, \mathbf{y}) \leq f(\mathbf{x}', \mathbf{y}')$. Aus dieser Definition folgt, dass ein Problem möglicherweise mehrere, nicht eindeutig bestimmte optimale Lösungen besitzt. Zudem besteht eine typische

[16] Als Standardform wurde hier ein Minimierungsproblem gewählt; da jedes Maximierungsproblems als Minimierungsproblem formuliert werden kann, ist dies keine Einschränkung.

[17] Häufig werden die zulässigen Punkte auch zulässige Lösung genannt. Dies wollen wir in diesem Buch nicht tun, da das Ziel der Optimierung die Bestimmung des optimalen Punktes, bzw. im Falle von Nicht-Eindeutigkeit auch optimaler Punkte ist, und somit nur solche als Lösung angesehen werden können.

Schwierigkeit im Umfeld der nichtlinearen nichtkonvexen Optimierung darin, dass es meist nur möglich ist, lokale Optima zu bestimmen, es aber viel schwieriger ist, nachzuweisen, dass ein globales Optimum bestimmt wurde. In einfachen Worten ausgedrückt ist ein globales Optimum der Punkt, dessen Zielfunktionswert nicht schlechter ist als der *aller* anderen zulässigen Punkte, während ein lokales Optimum $(\mathbf{x}_*, \mathbf{y}_*)$ diese Eigenschaft nur in einer Nachbarschaft dieses Punktes $(\mathbf{x}_*, \mathbf{y}_*)$ hat. Die formale Definition dafür lautet: In einem gegebenem Minimierungsproblem heißt ein Punkt $(\mathbf{x}_*, \mathbf{y}_*) \in S$ *lokales Minimum bezüglich der Umgebung* U von $(\mathbf{x}_*, \mathbf{y}_*)$ (oder einfach *lokales Minimum*) wenn

$$f(\mathbf{x}_*, \mathbf{y}_*) \leq f(\mathbf{x}, \mathbf{y}), \quad \forall\ (\mathbf{x}, \mathbf{y}) \in U(\mathbf{x}_*, \mathbf{y}_*)$$

gilt. Ist $U(\mathbf{x}_*, \mathbf{y}_*) = S$, so heißt $(\mathbf{x}_*, \mathbf{y}_*)$ *globales Minimum*.

Nun ein Beispiel: Das ganzzahlige Optimierungsproblem in den Variablen y_1 und y_2

$$\min_{y_1, y_2} \left\{ 3y_1 + 2y_2^2 \;\middle|\; \begin{array}{l} y_1^4 - y_2 - 15 = 0 \\ y_1 + y_2 - 3 \geq 0 \end{array} \quad , \quad y_1, y_2 \in V = \mathbb{N}_0 = \{0, 1, 2, 3, ...\} \right\}$$

besitzt z. B. den zulässigen Punkt $(y_1, y_2) = (3, 66)$ und die eindeutige optimale Lösung $\mathbf{y}^* = (y_1, y_2)^* = (2, 1)$ mit $f(\mathbf{y}^*) = 8$.

Spezialfälle der Funktionen $f(\mathbf{x}, \mathbf{y})$, $\mathbf{g}(\mathbf{x}, \mathbf{y})$ und $\mathbf{h}(\mathbf{x}, \mathbf{y})$ sowie der Werte n_c und n_d liefern die unter den folgenden Bezeichnungen bekannten Optimierungsprobleme

Abkürzung	*Optimierungsprobleme*
LP	Lineare Optimierung [engl.: *linear programming*]
MILP	Gemischt-Ganzzahlige Lineare Optimierung [engl.: *mixed integer linear programming*]
MIP	Gemischt-Ganzzahlige Optimierung [engl.: *mixed integer programming*]
NLP	Nichtlineare Optimierung [engl.: non*linear programming*]
MINLP	Gemischt-Ganzzahlige nichtlineare Optimierung [engl.: *mixed integer nonlinear programming*]
GLOBAL	Globale Optimierung für nichtkonvexe NLPs oder MINLPs

.

Diese Optimierungsprobleme sind Spezialfälle von (1.8.1)

Acronym	$f(\mathbf{x}, \mathbf{y})$	$\mathbf{h}(\mathbf{x}, \mathbf{y})$	$\mathbf{g}(\mathbf{x}, \mathbf{y})$	n_d	*zu lösende Unterprobleme*
LP	$\mathbf{c}^{\mathrm{T}}\mathbf{x}$	$\mathsf{A}\mathbf{x} - \mathbf{b}$	$\mathbf{x}$	0	Matrixinversion
QP	$\mathbf{x}^{\mathrm{T}}Q\mathbf{x} + \mathbf{c}^{\mathrm{T}}\mathbf{x}$	$\mathsf{A}\mathbf{x} - \mathbf{b}$	$\mathbf{x}$	0	
MILP	$\mathbf{c}^{\mathrm{T}}\mathbf{x}_{\oplus}$	$\mathsf{A}\mathbf{x}_{\oplus} - \mathbf{b}$	$\mathbf{x}_{\oplus}$	≥ 1	LP-Probleme
NLP	$f(\mathbf{x})$	$F_2(\mathbf{x})$	$F_3(\mathbf{x})$	0	Lösung nichtlinearer Gleichungen
MINLP				≥ 1	MILP- und NLP-Probleme
GLOBAL				≥ 0	NLP, Branch&Bound

mit einer aus m Zeilen und n Spalten bestehenden Matrix $\mathsf{A} \in \mathcal{M}(m \times n, \mathbb{R})$, sowie $\mathbf{b} \in \mathbb{R}^m$, $\mathbf{c} \in \mathbb{R}^n$ und $n = n_c + n_d$. MIPs ohne kontinuierliche Variablen sind ganzzahlige Programme (IPs). Manchmal lässt sich ein Problem aus äquivalent in mehreren Klassen formulieren. Eine Binärvariable δ in einem MILP-Problem kann auch als kontinuierliche Variablen in der Gleichungsbedingung

$$\delta\,(1-\delta) = 0$$

in einem NLP-Problem auftreten. Ein anderes Beispiel finden wir in Rebennack *et al.* (2009,[245]), wo eine exakte Transformation des klassischen, als MILP formulierten *fixed charge network flow*-Problems auf ein kontinuierliches Problem mit Bilineartermen gegeben wird, d. h. die Autoren reformulieren das MILP-Problem als ein QP-Problem. In der LP und MILP schreibt man meist statt $\mathsf{A}\mathbf{x} - \mathbf{b} = 0$ die Konstanten auf der rechten Seite, also $\mathsf{A}\mathbf{x} = \mathbf{b}$. Da einige dieser Optimierungsprobleme als Unterprobleme bei der Lösung anderer auftreten, ist es sehr wichtig, die Lösungsalgorithmen gut zu verstehen und so effizient wie möglich zu implementieren. Während sich LP-Probleme noch relativ leicht lösen lassen (der Aufwand zur Lösung eines LP-Problems mit m Nebenbedingungen und n Variablen wächst linear in m), zeigen MILP- und MINLP-Probleme exponentiell mit n_d wachsende, sehr strukturabhängige Laufzeiten. Lokale Lösungsverfahren für NLP-Problemen arbeiten iterativ; sie werfen Fragen der Konvergenz und Bereitstellung guter Startwerte auf und führen in nichtkonvexen Problemen auf die Problematik lokaler und globaler Optima. Globale Optimierungstechniken [*cf.* Horst & Pardalos (1995,[136]), Floudas (2000,[89]), oder Floudas & Gounaris (2009,[90])] finden sowohl bei nichtkonvexen NLP- als auch bei MINLP-Problemen Anwendung; die Komplexität wächst auch hier exponentiell mit der Anzahl der nichtlinear auftretenden oder ganzzahligen Variablen.

Von einem praktischen Standpunkt aus betrachtet wollen wir Optimierungsprobleme als schwierig bezeichnen, wenn sie nicht optimal oder zumindest nicht bis zu einer vorgegeben Genauigkeit innerhalb einer vernünftigen Zeit gelöst werden können. Problemstruktur, Größe oder beides kann zu schwierigen Problemen führen. Von einem theoretischen Standpunkt aus führen Überlegungen nach den schlechtestem Zeit- oder Speicherverhalten bei MILP- oder MINLP-Problems zur Klassifizierung $\mathcal{NP}$*-schwer*. Probleme in dieser Klasse haben schlechte Skalierungseigenschaften; wir müssen das Schlimmste befürchten. Auch wenn wir eine bestimme Probleminstanz noch gut lösen können, können leicht veränderte Eingabedaten, die das Problem auch nur geringfügig vergrößern dazu führen, dass wir das Problem überhaupt nicht mehr in vernünftiger Zeit lösen können.

1.9 Konventionen und Abkürzungen

Zulässige Indexkombinationen werden der kompakteren Schreibweise und einfacheren Lesbarkeit wegen abgekürzt. Bei Verwendung numerischer Indizes verwenden wir

$$\forall\{st\} := \{(s,t)|s = 1,\ldots,S; t = 1,\ldots,T\}$$

bzw. bei alphanumerischen Indexmengen

$$\forall\{st\} := \{(s,t)|s \in \mathcal{S}; t \in \mathcal{T}\} \quad ,$$

wobei der Operator $\forall$ als *für alle* zu lesen ist. Gelten bestimmte Ausdrücke nur unter bestimmten logischen Voraussetzungen, so wird dies durch das Symbol | ausgedrückt; im folgenden Beispiel

$$\frac{b_e}{A_{ej}} = \min_{i\in\mathcal{I}} \left\{ \frac{b_i}{A_{ij}} \middle| A_{ij} > 0 \right\} \quad , \quad j \in \mathcal{J}$$

werden also nur die Beiträge mit positiven A_{ij} berücksichtigt. Häufig ist die Erzeugung einer Variablen oder einer Nebenbedingung an die Existenz eines bestimmten Tabellenwertes gekoppelt; eine obere Schranke für die Lagervariable s_i wird z. B. nur dann generiert, wenn eine Lagerkapazität S_i existiert, d. h.

$$s_i \leq S_i \quad , \quad \forall\{i|\exists S_i\} \quad ,$$

wobei wir den Existenzoperator $\exists$ verwendet haben.

Vektoren sind fettgedruckt (Beispiel: $\mathbf{x}$ oder $\mathbf{F}$), Matrizen sind z. B. durch A gekennzeichnet. Mit $\mathbb{1}$ ist die Einheitsmatrix passender Dimension bezeichnet. Der Vektor $\mathbf{e}$ ist ein Vektor passender Dimension, dessen Komponenten alle den Wert 1 haben. Bekannte Größen und die Daten eines Modells werden mit großen Buchstaben bezeichnet. Kleine Buchstaben repräsentieren Variablen, kleine griechische Buchstaben binäre, ganzzahlige, halbstetige, partiell-ganzzahlige oder allgemein diskrete Variablen. Indizes werden zur Kennzeichnung der Abhängigkeit von Variablen oder Daten von den Modellobjekten verwendet; hochgestellte Attribute, wie z. B. in C^T_{sd}, werden zur weiteren Spezifikation, hier der Transportkosten, verwendet. Die folgenden Symbole

Symbol	*Menge*
$\mathbb{N}, \mathbb{N}_0$	der natürlichen Zahlen; $\mathbb{N} := \{1, 2, 3, \ldots\}$, $\mathbb{N}_0 := \{0, 1, 2, 3, \ldots\}$
$\mathbb{R}$, $\mathbb{R}_0^+$	der reellen Zahlen, $\mathbb{R}_0^+ := \{x \in \mathbb{R} \mid x \geq 0\}$
$\mathbb{Z}$	der ganzen Zahlen; $\mathbb{Z} := \{\ldots, -3, -2, -1, 0, 1, 2, 3, \ldots\}$

bezeichen diverse Zahlenmengen. Die nachfolgende Tabelle enthält schließlich in alphabetischer Reihenfolge die in diesem Buche verwendeten Abkürzungen:

Abkürzung	*Bedeutung*
CP	Constraint Programming
IPM	Innere-Punkte-Methoden [engl.: *interior point methods*]
KKT	Karush-Kuhn-Tucker
LP	lineare Programmierung / Optimierung
MILP	gemischt-ganzzahlige lineare Programmierung / Optimierung
MIP	gemischt-ganzzahlige Programmierung / Optimierung
MINLP	gemischt-ganzzahlige nichtlineare Programmierung / Optimierung
NLP	nichlineare Programmierung / Optimierung
OA	Äußere Approximation [engl.: *outer approximation*]
PRP	Projekt-Ressourcen-Planer (siehe Abschnitt 7.5)
SLP	sequentielle lineare Programmierung
SQP	sequentielle quadratische Programmierung
u.d.N.	unter den Nebenbedingungen
WAA	Wiederaufbereitungsanlage

2 Einführende motivierende Beispiele

In diesem Kapitel werden einige einführende Beispiele zur Modellbildung linearer, linearer gemischt-ganzzahliger und nichtlinearer kontinuierlicher Optimierungsprobleme gegeben, die den Leser mit den Bausteinen eines Optimierungsmodells, d. h. Daten, Variablen, Nebenbedingungen und Zielfunktion, vertraut werden lassen sollen und ihn in die Lage versetzen sollen, erste Modellformulierung selbständig vorzunehmen. Dabei wird auf die schon in Kapitel 1 gelegten allgemeinen Grundlagen zur Modellbildung Bezug genommen. Bei den noch recht übersichtlichen Problemen, die sich aus echten Problemen ableiten, hier aber dazu dienen, bestimmte Strukturen herauszuarbeiten, ist es noch möglich, die Probleme exakt in einem mathematischen Modell abzubilden. In den später behandelten größeren Problemen wird man bemüht sein, soviel Realität wie nötig und sinnvoll abzubilden, aber doch auch Vereinfachungen vorzunehmen.

2.1 Beispiel: Lineare Optimierung - Verleihung von Booten

In diesem Beispiel werden die grundlegenden Elemente eines Optimierungsmodells illustriert. Die Firma *Sunshine Cruises Ltd.* mit Sitz in England ist darauf spezialisiert, Flussboote an Urlauber zu verleihen. In einer typischen Urlaubswoche erzielt sie für Standard- und Premierboote einen Nettoerlös (Einnahmen minus variable Kosten) von £600/Woche bzw. £800/Woche. Die Firma möchte den alten gepachteten Bootsbestand durch neue Boote ersetzen und möchte den Entscheidungsprozess durch einige quantitative Überlegungen aus dem Bereich der mathematischen Optimierung absichern.

Insgesamt besitzt die Firma eine Anlegekapazität von 350 Booten; allerdings dürfen es nicht mehr als 200 Premierboote sein. Es sollen aber auch nicht weniger Premierboote als Standardboote sein. Jedes Premierboot erfordert etwa 4 Stunden Wartungsaufwand pro Woche; für Standardboote sind es nur 3 Stunden. Die verfügbaren Arbeitskräfte erlauben einen Gesamtwartungsaufwand von höchstens 1400 Stunden pro Woche.

Das Problem erscheint recht einfach und wohl strukturiert; einige kompliziertere Aspekte sind vernachlässigt und lassen es hier einfacher verdaulich erscheinen. Um zu einer Modellformulierung zu gelangen, fragen wir: Welche Freiheitsgrade stehen zur Verfügung und welche Entscheidungen müssen getroffen werden? Anlegekapazitäten und Wartung sind abgeleitete Informationen, nicht aber fundamentale Entscheidungen. Fundamentale Entscheidungen sind: *Wieviele Premier- und Standardboote sollen angemietet werden?* Sobald diese Fragen beantwortet sind, können alle übrigen interessierenden Größen berechnet werden. Wir werden nun das Problem mit Hilfe von Entscheidungsvariablen oder kurz *Variablen* formulieren, die bestimmten Nebenbedingungen unterliegen.

J. Kallrath, *Gemischt-ganzzahlige Optimierung: Modellierung in der Praxis*,
DOI 10.1007/978-3-658-00690-7_2, © Springer Fachmedien Wiesbaden 2013

Zudem wird der Formulierung eine Zielfunktion hinzugefügt, die in unserem Falle zu maximieren ist.

Bezeichne p die Anzahl der Premierboote, s die der Standardboote. Sinnvollerweise erwarten wir, dass die Variablen p und s keine negativen Werte annehmen können und stets ganzzahlig sind. Den Aspekt der Ganzzahligkeit wollen wir hier zunächst vernachlässigen, d. h. wir würden auch eine Lösung mit Wert $p = 18.3$ akzeptieren.

Mit Hilfe der eingeführten Variablen können wir nun die Modellbedingungen formulieren und somit p und s verknüpfen. Die Anlegebeschränkung führt auf die Ungleichung

$$p + s \leq 350 \quad . \tag{2.1.1}$$

Die Bedingung, dass höchstens 200 Premierboote angemietet werden können, wird durch

$$p \leq 200$$

berücksichtigt. Derartige Ungleichungen, die nur eine Variable enthalten, die mit Koeffizient 1 auftritt, werden Schranken genannt und können numerisch effizienter als allgemeine Ungleichungen behandelt werden.

Die Wartungsbedingung erfordert, dass die gesamt erforderliche Wartungszeit $4p$ und $3s$ für die Premier- und Standardboote die verfügbare Stundenzahl von 1400 nicht überschreitet, also

$$4p + 3s \leq 1400 \quad .$$

Weiter muss noch sichergestellt werden, dass nicht weniger Premierboote als Standardboote angemietet werden, d. h.

$$p \geq s \quad \text{bzw.} \quad p - s \geq 0 \quad .$$

Schließlich wird noch gefordert, dass die Variablen keine negativen Werte annehmen, d. h.

$$p \geq 0 \quad , \quad s \geq 0 \quad .$$

Sämtliche Ungleichungen, Schranken und Gleichungen im Modell, die den zulässigen Bereich implizit beschreiben, werden Nebenbedingungen genannt.

Hinsichtlich der Aufstellung der Nebenbedingungen ist zu beachten, dass jede für sich verschiedene Einheiten haben mag, z. B. Stunden oder Anzahl von Booten, die linke und rechte Seite aber stets die gleichen Einheiten haben müssen. Insbesondere ist darauf zu achten, dass nicht versehentlich *kg* und *cm* addiert werden. Vorsicht ist auch gefordert bei der Addition von Größen gleichen Types aber verschiedener Einheit, z. B. *cm* und *km*, oder *Liter* und m^3; in solchen Fällen ist eine Konvertierung der Einheiten erforderlich.

Nachdem nun alle Bedingungen des Problems modelliert wurden, können wir uns der Zielfunktion zuwenden. Hier ist es sinnvoll, den Nettoerlös pro Woche zu maximieren. Jedes Premierboot ergibt £800 pro Woche; bei Anmietung von p Premierbooten ergibt sich also $800p$. Entsprechend folgt für die Standardboote $600s$, und somit die in £ gemessene und zu maximierende Zielfunktion

$$z = z(p, s) = 800p + 600s \quad .$$

Das gesamte Modell kann also wie folgt zusammengefasst werden:

$$\max \quad 800p + 600s \tag{2.1.2}$$

unter den Bedingungen

$$\begin{array}{rcrcr} p & + & s & \leq & 350 \\ p & & & \leq & 200 \\ 4p & + & 3s & \leq & 1400 \\ p & - & s & \geq & 0 \end{array} \quad , \quad p \geq 0 \quad , \quad s \geq 0 \quad . \tag{2.1.3}$$

In der Praxis werden solche Probleme mit Hilfe der in Abschnitt 4.1 beschriebenen Verfahren gelöst. Im vorliegenden Modell mit nur 2 Variablen, ist es jedoch auch möglich, das Problem graphisch zu lösen und damit ein Gefühl für die Lösung zu entwickeln.

Natürlich könnte man sich der Lösung auch mit einem Versuch-und-Irrtum-Verfahren [engl.: *trial and error*] nähern. Versucht man die Werte 50, 100, 150, 200, 250, 300 und 350 für die Variablen, so findet man z. B. das Wertepaar $p = 150$ und $s = 150$ mit dem Nettoerlös £210,000 oder $p = 200$, $s = 100$ und $z = £220,000$. Dieses Verfahren ist jedoch recht aufwendig und bei Auswertung weniger Kombinationen zu ungenau.

Der zulässige Lösungsraum lässt sich graphisch wie in Abbildung 2.1 darstellen: s ist auf der horizontalen, p auf der vertikalen Achse aufgetragen. Da s und p nichtnegative Variablen sind, brauchen wir uns nur um den Teil des Graphen im nichtnegativen Quadranten zu kümmern. In der Abbildung entsprechen die Grenzen des zulässigen Bereichs den aus der Ungleichung

$$p + s \leq 350$$

abgeleiteten Geraden

$$p = -s + 350 \quad .$$

Zulässige Kombinationen der Variablen p und s ergeben sich aus der Geraden $p = -s+350$ und dem Halbraum

$$p < -s + 350$$

bzw. genauer dem Bereich zwischen der Gerade und den Koordinatenachsen; hinsichtlich der anderen Ungleichungen verfährt man entsprechend. Der schattierte Bereich umfasst alle zulässigen Wertepaare der Variablen p und s. Die Zielfunktion bzw. ihre Konturlinien, d. h. die Menge aller Punkte gleichen konstanten Wertes k, sind Geraden der Form

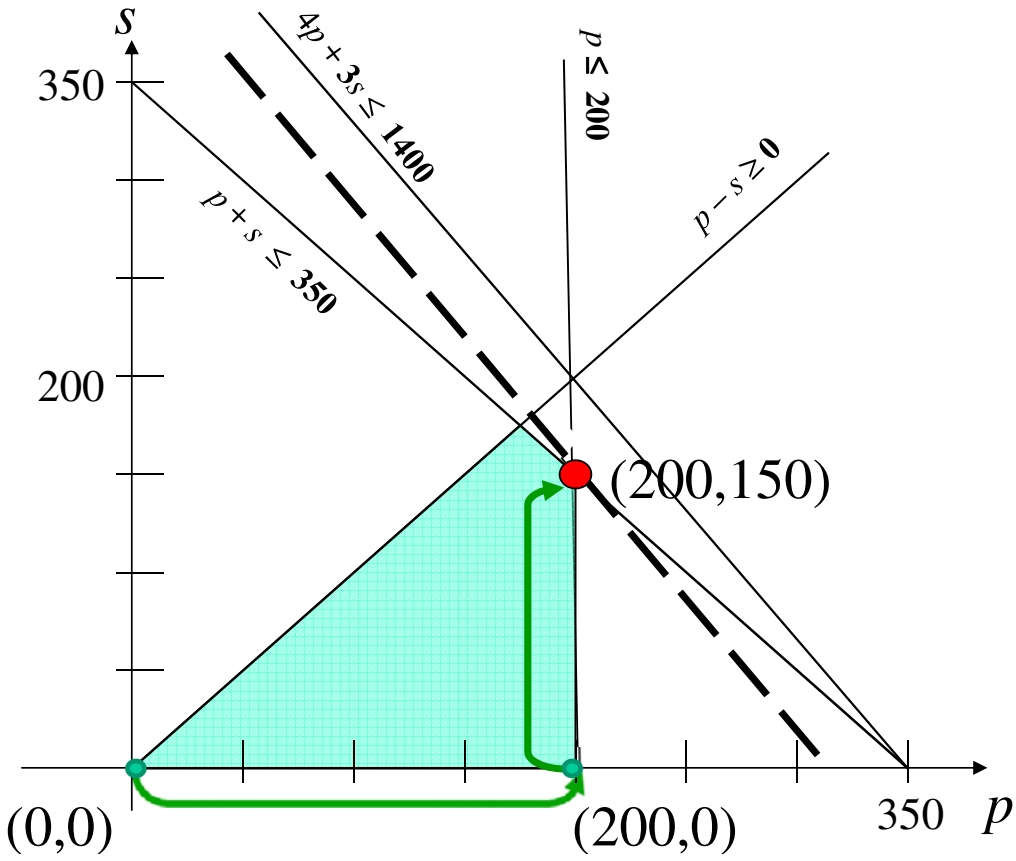
$$800p + 600s = k \quad .$$

Ein passender Wert für k lässt sich ableiten, in dem man typische Werte s und p annimmt, z. B. $s = p = 200$; weiter lässt sich die Steigung $m = -0.75$ der Geraden berechnen.

Da die Zielfunktion maximiert werden soll, liegt es nahe, die Niveaulinien der Zielfunktion so parallel zu verschieben, dass die Konstante k wächst und die Konturlinie noch eine Schnittmenge mit dem zulässigen Bereich hat; die gedachte Parallelverschiebung erfolgt in Richtung des Gradienten $\boldsymbol{\nabla} z = \partial z/\partial(p, s) = (800, 600)^{\mathrm{T}}$ bis die Konturlinie den Punkt $(p, s) = (200, 150)$ erreicht – jede weitere Verschiebung führt aus dem zuverlässigen Bereich hinaus. Die Feststellung, dass die optimale Lösung auf einer Ecke liegt, lässt sich unter bestimmten Voraussetzungen auf jedes LP-Problem übertragen.

In diesem kurzen Beispiel lautet die optimale Lösung also $(p, s) = (200, 150)$ mit Zielfunktionswert $z = £250,000$. In die Sprache der Firma zurückübersetzt bedeutet dies, 200 Premierboote und 150 Standardboote anzumieten; daraus resultiert ein wöchentlicher Nettoerlös von £250,000. Wie erwartet, werden mehr lukrative Premierboote angeboten. Die Beschränkung, höchstens 200 Premierboote anmieten zu können, limitiert die Zahl der anzumietenden Premierboote; die Entscheidungsvariable p liegt gerade an ihrer oberen Schranke. Im optimalen Lösungspunkt $(p_*, s_*) = (200, 150)$ ist wegen $4p_*+3s_* = 1250$

Abbildung 2.1 Graphische Lösung eines LP-Problems mit zwei Variablen. Entlang der horizontalen Achse ist die Anzahl p der Premierboote, entlang der vertikalen Achse die Anzahl s der Standardboote aufgetragen. Die gestrichelte Linie ist die Konturlinie der Zielfunktion, die durch den Lösungspunkt $(p, s) = (200, 150)$ verläuft. Weiter sind hervorgehoben die Initialisierung (0,0) und der nächste Vertex (200,0), der im Simplexverfahren untersucht wird.



die Stundenbeschränkung dagegen nicht limitierend, d. h. im Fachjargon *nicht aktiv*. Zu bemerken ist noch, dass die Lösung ohne besonderes Zutun bereits ganzzahlig ist; dies ist im allgemeinen jedoch nicht so.

2.2 Beispiele: Gemischt-ganzzahlige lineare Optimierung

Im folgenden wird anhand einiger Beispiele verdeutlicht, wie sich *praktische Probleme* mit Hilfe gemischt-ganzzahliger Modelle formulieren lassen. Dabei zeigt sich, dass ganzzahlige Variablen im wesentlichen zur Beschreibung der folgenden Strukturen verwendet werden: Nichtteilbare und zählbare Größen [siehe z. B. die Abschnitte 2.2.1 und 7.4.2],

Minimalgrößen [siehe z. B. Abschnitt 7.4.1], Zustände und *ja/nein*-Entscheidungen und logische Bedingungen [Kapitel 5.1 und Abschnitt 5.2], und schließlich einfache nichtlineare Funktionen [Abschnitte 5.5, 5.6.4 und 5.7].

2.2.1 Beispiel 1: Von Kühen und Schweinen

Ein Landwirt mit Schwerpunkt Viehhaltung (Kühe, Schweine) möchte seinen Tierbestand erweitern. Er hat die Möglichkeit, Kälber und Jungschweine für den Preis von 1 kEuro=1000 Euro zu erwerben; ihm stehen jedoch augenblicklich nur 3.5 kEuro zur Verfügung. Später kann er einen Erlös von 3 kEuro je Kuh bzw. 2 kEuro je Schwein erzielen. Da seine Stallungen begrenzt sind und er diese nicht baulich erweitern will, muss er die Randbedingungen berücksichtigen, dass er höchstens 2 Jungschweine und 2 Kälber kaufen kann. Natürlich möchte der Landwirt seinen zukünftigen Erlös maximieren. Ihm stellt sich nun die Frage, wieviele Kälber und Jungschweine er erwerben soll. Die offensichtliche Lösung zu diesem Problem besteht darin, 2 Kälber und ein Jungschwein zu kaufen. Dieses überschaubare Problem soll nun in eine mathematische Form übersetzt und als ganzzahliges Optimierungsproblem gelöst werden.

Die Aufgabe des Modellierers besteht zunächst darin, das gegebene Problem in eine mathematische Form zu transformieren. Dabei sind zuerst *Variablen* zu identifizieren, mit deren Hilfe sich *Zielfunktion* und *Nebenbedingungen* beschreiben lassen. Die Variablen werden meist so gewählt, dass sich damit gleich die wesentlichen Fragen der Aufgabenstellung beantworten lassen. Im vorliegenden Fall bietet es sich also an, **Variablen** κ und σ mit den Eigenschaften

$$\kappa, \sigma \in \mathbb{N}_0 := \{0, 1, 2, 3, 4, ...\}$$

für die Zahl der zu kaufenden Kälber und Jungschweine einzuführen. Die Wahl der Grundmenge $\mathbb{N}_0$ berücksichtigt, dass lebende Kälber und Schweine nur in ganzen Einheiten gekauft werden können. Mit Hilfe von κ und σ lässt sich nun die **Zielfunktion** einfach schreiben als

$$\max Z \quad , \quad Z = 3\kappa + 2\sigma \quad . \tag{2.2.1}$$

Die **Nebenbedingungen** nehmen die Form

$$0 \leq \kappa, \sigma \leq 2$$

und

$$\kappa + \sigma \leq 3.5 \tag{2.2.2}$$

an. Steht kein Optimierungsverfahren zur Verfügung, so bietet es sich bei diesem überschaulichen Problem noch an, sämtliche Werte, die κ und σ annehmen können, in die Zielfunktion (2.2.1) einzusetzen und das Wertepaar (κ, σ) zu bestimmen, das den maximalen Erlös liefert. Wegen $\kappa, \sigma \leq 2$ brauchen dabei nur sämtliche $2-$Tupel aus $\{0, 1, 2\}$, d. h. insgesamt $3^2 = 9$, ausprobiert werden. Die Ergebnisse dieser Vorgehensweise sind in der nachstehend dargestellten Tabelle zusammengefasst:

κ	0	1	2	0	1	2	0	1	2
σ	0	0	0	1	1	1	2	2	2
Z	0	3	6	2	5	8	4	7	-

.

In der Tat liest man aus dieser Tabelle das erwartete Ergebnis ($\kappa = 2, \sigma = 1$) und $Z = 8$ ab. Das Wertepaar $(2, 2)$ ist unzulässig, da es die Bedingung (2.2.2) verletzt. Bei Problemen niedriger Dimension kann die vorgestellte Methode zum Erfolg führen. In dem Fall, dass der Landwirt jedoch statt nur 2 Kälber bzw. Kühe etwa jeweils 9 Tiere kaufen könnte und statt 3.5 kEuro nun 20 kEuro zur Verfügung hätte, müssten bereits $10^2 = 100$ 2−Tupel aus $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ getestet werden. Noch unübersichtlicher würden die Verhältnisse, wenn nicht nur Kälber und Schweine, sondern vielleicht auch noch Hühner, Puten und anderes Getier beim möglichen Kauf berücksichtigt werden müssten. Aus diesem Grunde ist es das Ziel, ein Optimierungsverfahren einzusetzen, das auch in allgemeinen Fällen die Komplexität beherrschen kann.

2.2.2 Beispiel 2 : Ein Projektplanungssystem

Für die nächsten T Zeitperioden, z. B. T Monate, sollen in einer Firma aus einer möglichen Auswahl von P Projekten solche ausgewählt werden, die einen maximalen Gewinn ermöglichen. In jeder Zeitperiode t steht ein Kapital von B_t Euro zur Verfügung. Wird in der Zeitperiode t das Projekt p bearbeitet, so entstehen durch die Beanspruchung von Personal, Maschinen oder Räumen, Kosten in Höhe von K_{pt} Euro im Laufe der Periode t und ein Erlös in Höhe von E_p Euro am Ende des Planungszeitraumes.

Wir können nun mit Hilfe der binären **Variablen** δ_p

$$\delta_p = \begin{cases} 1, \text{ das Projekt } p \text{ wird bearbeitet} \\ 0, \text{ sonst} \end{cases} \quad , \quad \delta_p \in \{0, 1\} \quad , \quad p = 1, ..., P$$

die **Zielfunktion**

$$\max \; Z = \sum_{p=1}^{P} E_p \delta_p$$

formulieren; die **Nebenbedingung** hat mit diesen Variablen die Gestalt

$$\sum_{p=1}^{P} K_{pt} \delta_p \leq B_t \quad , \quad t = 1, ..., T \quad .$$

Probleme mit dieser Struktur nennt man auch binäre Rucksackprobleme [engl.: *binary knapsack problem*].

2.2.3 Beispiel 3 : Ein Produktionsplanungssystem

Betrachtet werden soll ein Produktionsplanungssystem mit N Produkten. Die mit der Produktion eines Produktes p anfallenden Kosten setzen sich aus Rüst- und Proportionalkosten K_p bzw. C_p zusammen. Zur Herstellung von Produkt p werden A_{rp} kg eines Rohstoffes r benötigt; insgesamt stehen M verschiedene Rohstoffe in beschränkter Menge B_r zur Verfügung. Erwartet wird, dass maximal D_p kg von Produkt p zu einem Erlös von E_p Euro/kg nachgefragt werden.

Nachstehend werden die Produktions- und Identifikations-**Variablen**

$$\delta_p = \begin{cases} 1 \text{ , Produkt } p \text{ wird hergestellt} \\ 0 \text{ , sonst} \end{cases} \quad , \quad p = 1, ..., P$$

$$0 \leq x_p = \text{Menge, die von Produkt } p \text{ produziert wird}$$

eingeführt, mit deren Hilfe die **Zielfunktion**

$$\max\ Z = \sum_{p=1}^{P} E_p x_p - \sum_{p=1}^{P} (K_p \delta_p + C_p x_p)$$

und die **Nebenbedingungen**

$$\sum_{p=1}^{P} A_{rp} x_p \leq B_i \quad , \quad i = 1, ..., I$$

sowie

$$0 \leq x_p \leq D_p \delta_p \quad , \quad \forall p$$

formuliert werden. Zu bemerken ist, dass $x_p > 0$ nur möglich ist, wenn $\delta_p = 1$ gilt; $\delta_p = 0$ impliziert nämlich $x_p = 0$.

2.2.4 Beispiel 4 : Optimale Depotwahl - Ein Standortplanungsmodell

Eine Mineralölgesellschaft möchte in der Zentralschweiz 2 durch Pipelines versorgte Depots eröffnen, die 4 umliegende Tankstellen mit bekannten Nachfragemengen D_1, D_2, D_3 und D_4 versorgen sollen. Infolge der schwierigen Verhältnisse im Gebirge sind die Verbindungen zwischen Depots und Tankstellen eingeschränkt; $L_{ij} \in \{0, 1\}$ beschreibt, ob zwischen Depot i und Tankstelle j eine Verkehrsverbindung besteht, z. B. sollen als mögliche Verbindungen nur

L_{11}, L_{12}, L_{14}	es gibt keine Verbindung von Depot 1 zu Tankstelle 3
$L_{21}, L_{22}, L_{23}, L_{24}$	Depot 2 beliefert alle Tankstellen
L_{32}, L_{33}, L_{34}	es gibt keine Verbindung von Depot 3 zu Tankstelle 1

in Betracht gezogen werden. Reparatur- oder Veränderungsaspekte in den Depots im Laufe der Zeit sollen nicht betrachtet werden. Von den möglichen Positionen O_1, O_2 und O_3 der Depots mit Kapazitäten A_1, A_2 und A_3 sollen 2 so gewählt werden, dass die Summe aus Investitions- K_i, Operations- P_i und Transportkosten C_{ij} für diesen Zeitraum minimal wird.

Zur Beschreibung dieses Problems führen wir die **binären Variablen**

$$\delta_i = \begin{cases} 1 \text{ , Ort } i \text{ wird gewählt} \\ 0 \text{ , sonst} \end{cases} \quad , \quad \delta_i \in \{0, 1\} \quad , \quad i = 1, 2, 3$$

und für die Indexkombinationen (i, j), für die $L_{ij} = 1$, die kontinuierlichen Variablen

$$0 \leq x_{ij} = \text{Menge, die vom Ort } i \text{ zur Tankstelle } j \text{ transportiert wird}$$

ein. Mit diesen Variablen ist die **Zielfunktion**

$$\min\ Z = \sum_{i=1}^{3} K_i \delta_i + \sum_{i=1}^{3} \sum_{j=1 | L_{ij}=1}^{4} (P_i + C_{ij}) L_{ij} x_{ij}$$

unter den **Nebenbedingungen**

$$\sum_{j=1|L_{ij}=1}^{4} L_{ij}x_{ij} \leq A_i\delta_i \quad , \quad i=1,2,3 \quad , \tag{2.2.3}$$

$$\sum_{i=1|L_{ij}=1}^{3} L_{ij}x_{ij} = D_j \quad , \quad j=1,2,3,4 \tag{2.2.4}$$

und

$$\delta_1 + \delta_2 + \delta_3 = 2 \tag{2.2.5}$$

zu minimieren. Eine genauere Betrachtung der Kapazitätsbeschränkungen (2.2.3) zeigt, dass diese die Implikation $\delta_i = 0 \rightarrow x_{ij} = 0$ beinhalten, während (2.2.4) die Forderung nach Erfüllung der Nachfrage beschreibt. Schließlich erzwingt (2.2.5), dass genau 2 Depots gebaut werden.

Die vorliegende Fragestellung stellt ein typisches Standortplanungsproblem [engl.: *facility location problem*] dar. Die Standortplanung legt die Anzahl und Standorte der Knoten – hier die der Depots – und die gesamte Struktur eines Distributionsnetzwerkes fest. Je nach Detaillierungs- und Aggregierungsgrad fließen Transport-, Bestands- und Produktionsplanung mit ein. Spezialfälle der Standortplanung lassen sich als Weber-Probleme behandeln; dem Leser sei hierzu Wesolowsky (1993,[289]) empfohlen.

2.2.5 Beispiel 5 : Optimale Produktverteilung

Angenommen, eine Horde von Studierenden stürmt ein Eiscafé. Nun ist aber nicht soviel Eis da, dass der Eisverkäufer jedem seinen ersten Eis-Wunsch erfüllen kann. Wie kann er eine optimale Zufriedenheit seiner Kunden erreichen, wenn diese jeweils drei Präferenzen angeben können?

Betrachten wir n Produkte $p \in \mathcal{P} = \{p_1, ...,p_n\}$ für m Kunden $k \in \mathcal{K} = \{k_1, ...,k_m\}$. Mit Hilfe der Indextabellen $W^w_{kp} \in \{0,1\}$ indizieren wir, dass Produkt p auf der Wunschliste von k Position $\in \{1,2,3\}$ einnimmt. Als Variablen führen wir

$$\delta_{kp} = \begin{cases} 1 \text{ , der Kunde } k \text{ erhält das Produkt } p \\ 0 \text{ , sonst} \end{cases} \quad , \quad \delta_{kp} \in \{0,1\} \quad , \quad \forall\{kp\}$$

und

$$\tilde{\alpha}_k = \begin{cases} 1 \text{ , der erste Wunsch des Kunden wird verletzt} \\ 0 \text{ , der erste Wunsch des Kunden wird erfüllt} \end{cases} \quad , \quad \forall k \in \mathcal{K}$$

ein. Die Variable $\tilde{\alpha}_k$ wird nur dann erzeugt, wenn der Kunde k das Produkt p als ersten Wunsch angegeben hat, d. h. W^{α}_{kp} existiert und hat z. B. den Wert 1 – die Tabelle indiziert also, ob das Produkt p für Kunden k der erste Wunsch des Kunden ist; ist $\tilde{\alpha}_k = 1$, so bedeutet dies, dass der erste Wunsch des Kunden nicht erfüllt wird. In ähnlicher Weise sind die Variablen

$$\tilde{\beta}_k = \begin{cases} 1 \text{ , der erste und zweite Wunsch des Kunden wird verletzt} \\ 0 \text{ , der zweite Wunsch des Kunden wird erfüllt} \end{cases} \quad \forall k \in \mathcal{K}$$

und

$$\tilde{\gamma}_k = \begin{cases} 1 \text{ , der erste, zweite und dritte Wunsch des Kunden wird verletzt} \\ 0 \text{ , der dritte Wunsch des Kunden wird erfüllt} \end{cases} \quad \forall k$$

definiert. Schließlich muss die Nebenbedingung

$$\sum_p \delta_{kp} = 1$$

erfüllt werden, d. h. jeder Kunde erhält genau ein Produkt.

Um die Zufriedenheit zu beurteilen, werden Straffaktoren S_k^α, S_k^β und S_k^γ eingeführt, die nach dem ersten, zweiten und dritten Wunsch, aber gegebenenfalls auch nach Kundenbedeutung differenziert werden. Je größer die potentielle Zufriedenheit, desto niedriger wird der Straffaktor gewählt. Die Zielfunktion lautet dann:

$$\min \sum_{k \in \mathcal{K}} \left[S_k^\alpha \tilde{\alpha}_k + S_k^\beta \tilde{\beta}_k + S_k^\gamma \tilde{\gamma}_k \right] \quad .$$

Für existierende Verbote, z. B. wegen Allergie oder Diabetes,

$$V_{kp} = \begin{cases} 1 \text{ , } p \text{ ist für } k \text{ verboten,} \\ 0 \text{ , sonst} \end{cases}$$

werden die Binärvariablen fixiert, d. h.

$$\delta_{kp} = 0 \quad , \quad \forall \{k, p | \exists V_{kp}\} \quad .$$

Die Variablen $\tilde{\alpha}_k$, $\tilde{\beta}_k$ und $\tilde{\gamma}_k$ sind mit den δ_{kp}-Variablen in der folgenden Weise verknüpft:

$$\tilde{\alpha}_k = 1 - \delta_{kp} \quad , \quad \tilde{\beta}_k = \tilde{\alpha}_k - \delta_{kp} \quad , \quad \tilde{\gamma}_k = \tilde{\beta}_k - \delta_{kp} \quad , \quad \forall \{kp\} \quad .$$

2.2.6 Beispiel 6 : Modellierung logischer Bedingungen

Mit Hilfe binärer Variablen lassen sich, wie in Kapitel 5.1 ausführlicher dargelegt, logische Bedingungen algebraisch formulieren. Beispielsweise lassen sich die Bedingungen „wenn Entscheidung 1 getroffen wird, dann muss die Variable x größer sein als irgendeine positive Konstante C" und „wenn Entscheidung 1 nicht getroffen wird, dann ist die Variable x gleich 0" durch die Ungleichungen

$$C\delta \leq x \leq X\delta$$

beschreiben. Hierbei stellt X eine obere Schranke von x dar und $\delta = 1$ beschreibt den Fall, dass z. B. die Entscheidung, ob Produkt P gestartet werden soll, positiv ausfällt; fällt die Entscheidung negativ aus, so ist $\delta = 0$.

2.2.7 Beispiel 7 : Optimale Einbruchstrategie - Ein Rucksackproblem

Das *Rucksackproblem* [engl.: *knapsack problem*] ist, wie in Salkin & de Kluyver (1975,[252]) gezeigt, sehr anwendungsrelevant und innerhalb der gemischt-ganzzahligen Optimierung von besonderer Bedeutung, da es als Unterproblem in vielen größeren Problemen enthalten ist und viele Techniken, die zu verbesserten MIP-Formulierungen führen, auf der Analyse von Rucksack-Strukturen beruhen. Das Rucksackproblems lässt sich am einfachsten

mit Hilfe eines Einbrechers veranschaulichen, der in das Haus einer wohlhabenden Familie einbrechen will. Er hat das Haus seit langem beobachtet und weiß, dass es zahlreiche Objekte enthält, die er sehr begehrt. Da er die Beute mit niemanden teilen möchte, wird er den Einbruch alleine durchführen; daher ist das Gewicht der einzelnen Objekte für ihn sehr relevant - ein schwerer Fernsehapparat scheidet als Beute aus, da ihm vor einiger Zeit sein Auto abhanden gekommen ist. Er ist allerdings ein sehr ordentlicher und wohl organisierter Dieb und deshalb hat er alle Informationen während der vergangenen 17 Wochen detailliert dokumentiert. Aus der Liste aller möglichen Objekte interessieren ihn besonders 8 Einzelobjekte. Die nachstehende Tabelle enthält die Daten dieser Objekte: Ihren geschätzten Wert V_i in Einheiten von 1000 Euro und ihr Gewicht W_i in Kilogramm

i	1	2	3	4	5	6	7	8
V_i	15	100	90	60	40	15	10	1
W_i	2	20	20	30	40	30	60	10

.

Zur Vorbereitung hat der potentielle Dieb während der vergangenen 3 Monate einen enormen Aufwand betrieben und viel Zeit in einem Body-Building-Zentrum verbracht, um seinen Körper zu stählen und für diese Aktion fit zu machen. Er glaubt so, zumindest für eine kurze Weile einen Sack mit einem Gesamtgewicht von 102 kg tragen zu können. Realistischerweise schätzt er die Situation so ein, dass er nur einmal einbrechen kann. Da alle Objekte zusammen 212 kg wiegen, kann er, er bedauert dies sehr, nicht alle Objekte mitnehmen. Lange hat er gegrübelt, welche Objekte er mitgehen lassen soll, wenn er den Gesamtwert maximieren möchte. Glücklicherweise hat unser notorischer Dieb bei einem seiner letzten Einbrüche eine Vielzahl von Büchern erbeutet und in einem von diesen Büchern fand er einige Erklärungen über das Rucksackproblem. Nachfolgend das, was er daraus gelernt hat:

Das allgemeine *Rucksackproblem* besteht darin, aus einer Menge von Objekten, deren Gewichte und Werte bekannt sind, eine Untermenge auszuwählen, die das zulässige Gewicht eines Rucksacks [engl.: *knapsack*] nicht überschreitet und den Gesamtwert der ausgewählten Objekte maximiert; die Objekte dürfen wie bei einem Einkauf mit beliebigen Vorrat mehrfach gewählt werden. Bereits bei Dantzig (1957,[62]) findet sich eine Modellformulierung und ein Lösung des Problems, bei dem die wesentliche Entscheidung darin liegt, zu bestimmen, wieviel Objekte von jedem Typ gewählt werden. Damit kann das Problem, das sehr umfassend und didaktisch hervorragend in Martello & Toth (1990,[203]) beschrieben ist, wie folgt formuliert werden.

Sei σ_i die ganzzahlige Variable, die beschreibt, wie viele Objekte vom Typ i ($i \in \{1, 2, ..., n\}$) gewählt werden, W_i das Gewicht vom Typ i, V_i der Wert vom Typ i und C die Kapazität des Rucksacks. Die zu maximierende Zielfunktion lautet somit

$$\max z \quad , \quad z := \sum_{i=1}^{n} V_i \sigma_i$$

und unterliegt der Nebenbedingung, dass die zulässige Kapazität des Rucksacks nicht überschritten werden darf

$$\sum_{i=1}^{n} W_i \sigma_i \leq C \quad , \quad \sigma_i \in \mathbb{N}_0 \quad .$$

Dieses Problem erscheint eigentlich recht einfach und ist wohl auch eines der am einfachsten zu formulierenden MIP-Probleme, da es nur eine Ungleichungsnebenbedingung

besitzt. Dennoch ist es sehr schwierig zu lösen. Die benötigte Rechenzeit zur Lösung des Problems wächst exponentiell in der Zahl n der Objekte. Für große n ist es nicht sinnvoll, das Problem mit einem MILP-Algorithmus zu lösen, sondern es ist besser, dies mit speziellen Verfahren zu tun.

Eine häufig verwendete Variante des Problems besteht wie im Beispiel in der Limitierung, dass jeder Objekttyp nur einmal gewählt werden darf; in diesem binären Rucksackproblem, das unser Dieb zu lösen hat, ist also $\sigma_i \in \{0, 1\}$. Hat er auf einem seiner früheren Beutezüge auch das Buch von Kallrath & Wilson (1997,[167]) erbeutet, so findet er darin auf der beigefügten CD unter dem Namen *burglar* eine Modelldatei seines Problems und z. B. mit `Xpress-MP`[1] die Lösung $\boldsymbol{\sigma} := \{1, 1, 1, 1, 0, 1, 0, 0\}$ und $z = 280$. Sein Rucksack wiegt mit diesen Objekten gerade 102 kg.

2.3 Beispiel: Nichtlineare Optimierung - Ein Mischungsproblem

Dieses übersichtliche Beispiel ist eine vereinfachte Version eines realen Problems, bei der es um die Entsorgung giftiger Schlacke ging, die einige schädliche Komponenten, z. B. Kadmium, enthält. Lieferungen giftiger Schlacke aus verschiedenen Quellen werden in einem Zwischenprozess gemischt (deswegen lassen sich die Komponenten nicht getrennt aufbewahren) und schließlich in Endlagerstellen deponiert. Formal kann das Problem wie folgt beschrieben werden: Schlacke, die Komponenten c aus Quellen i enthält, wird in Kläranlage j gemischt und verarbeitet und schließlich dem Endlager k zugeführt. Die maximale Menge der Komponente c, die im Endlager k aufgenommen werden kann, beträgt M_{ck}. Somit ergeben sich die Indizes des Problems

$$\begin{array}{rcl} c \in \{1, 2, ..., C\} & : & \text{Komponenten} \\ i \in \{1, 2, ..., I\} & : & \text{Quellen} \\ j \in \{1, 2, ..., J\} & : & \text{Kläranlagen} \\ k \in \{1, 2, ..., K\} & : & \text{Endlager} \end{array} .$$

Die Daten sind

$$\begin{array}{lcl} A_i & : & \text{die aus } i \text{ zugeführte Menge} \\ C_{jk} & : & \text{die Entsorgungskosten (Euro/Tonne), wenn Kläranlage } j \text{ nach } k \text{ entsorgt} \\ F_{ci} & : & \text{der relative Anteil der Komponente } c \text{ im Material aus Quelle } i \\ M_{ck} & : & \text{die maximale Menge von Komponente } c, \text{ die } k \text{ zugeführt werden darf} \end{array} .$$

Es bieten sich die folgenden Variablen an:

$$\begin{array}{lcll} q_{cj} & \geq & 0 & : \text{ Durchsatz an Komponente } c \text{ in } j \text{ in Tonnen} \\ t_j & \geq & 0 & : \text{ Gesamtmenge, die durch } j \text{ fließt} \\ x_{ij} & \geq & 0 & : \text{ Menge, die von } i \text{ nach } j \text{ gesendet wird} \\ y_{jk} & \geq & 0 & : \text{ Menge, die von } j \text{ nach } k \text{ entsorgt wird} \\ \lambda_{jc} & \geq & 0 & : \text{ relativer Anteil der Komponente } c \text{ in Kläranlage } j \end{array} .$$

Im vorliegenden Problem sollen die Entsorgungskosten

[1] `Xpress-MP` ist ein eingetragenes Markenzeichen der Firma Dash Optimization (http:/www.dashoptimization.com).

$$\min \quad \sum_{j=1}^{J} \sum_{k=1}^{K} C_{jk} y_{jk}$$

minimiert werden, wobei alle anfallende Schlacke entsorgt werden muss, d. h.

$$\sum_{j=1}^{J} x_{ij} = A_i \quad , \quad \forall i \quad .$$

Für jede Kläranlage j gelten somit die folgenden Massenbilanzen

$$\sum_{i=1}^{I} x_{ij} = t_j = \sum_{k=1}^{K} y_{jk} \quad , \quad \forall j$$

der ein- und abfließenden Mengen. Als nächstes wird die Menge q_{cj} [in Tonnen]

$$q_{cj} = \sum_{i=1}^{I} F_{ci} x_{ij}$$

der in j einfließenden Komponente c berechnet. Um die maximale Aufnahmekapazität für Komponente c in k beachten zu können, müssen wir die absolute Menge [in Tonnen] an Komponente c im Ausgang von j berechnen. Mit Hilfe der Konzentration λ_{cj}, [Anteil von Komponente c im Ausgangsstrom von Kläranlage j nach k] erhalten wir die Ungleichung

$$\sum_{j=1}^{J} \lambda_{cj} y_{jk} \leq M_{ck} \quad , \quad \forall \{ck\} \quad . \tag{2.3.6}$$

Schließlich muss die Variable λ_{cj} noch mit den übrigen Variablen verknüpft werden. Eine solche Relation folgt aus dem folgenden Erhaltungssatz: Der Anteil der Komponente c an der gemischten Schlacke in der Kläranlage j entspricht der Menge aller einfließenden Tonnen an giftigem c, dividiert durch die Menge aller Schlacken, die in j ankommen, d. h.

$$\lambda_{cj} := \phi_*(\mathsf{D}) = \frac{\sum_{i=1}^{I} F_{ci} x_{ij}}{t_j} = \frac{q_{cj}}{t_j} \tag{2.3.7}$$

oder, äquivalent hierzu

$$t_j \lambda_{cj} = q_{cj} \quad . \tag{2.3.8}$$

In beiden Fällen erhalten wir nichtlineare Terme, wobei (2.3.8) aus numerischen Gründen vorzuziehen ist, da hier eine Division durch Null nicht auftreten kann.

Das vorliegende Mischungsproblem wird in Abschnitt 9.1.1 mit Hilfe eines speziellen Verfahrens zur Lösung nichtlinearer Optimierungsprobleme, der *Rekursion*, gelöst.

2.4 Es muss nicht immer Optimierung sein

Die folgende Fallstudie und Analyse einer Kundenanfrage zeigt, dass manchmal bereits eine genaue Situationsanalyse zur Problemlösung führen kann. Es muss dabei nicht in jedem Fall auf ein mathematisches Optimierungsmodell hinauslaufen.

Ein Werksleiter einer Automobilfirma ist für die Produktion der Motoren, Getriebe und Achsen verantwortlich. Es werden 4-, 5-, 6- und 8-Zylindermotoren gefertigt; die 5- und 8-Zylindermotoren sind eher Sondertypen mit geringer Nachfrage und brauchen nicht weiter betrachtet zu werden. Für die 6-Zylindermotoren gibt es eine besonders hohe Nachfrage von 900 Motoren/Tag, die derzeitige Kapazität liegt aber nur bei 600 Motoren/Tag. Bei den 4-Zylindermotoren liegt sie bei 1200 Motoren/Tag, es werden aber nur 900 Motoren/Tag gebraucht. Der Werksleiter wendet sich an die OR-Gruppe des Werkes; sein Ziel ist die Deckung des Bedarfs der 6-Zylindermotoren.

Er gibt auf Anfrage der OR-Gruppe noch die weiteren Informationen. Die Herstellung eines Motors erfordert drei Arbeitsvorgänge:

1	Gießerei	Gießen / Schmieden
2	Transferstraße	Drehen / Frisieren
3	Montage	Motorenmontage

.

Die Arbeitsvorgänge 1 und 3 unterscheiden sich nicht für 4- und 6-Zylindermotoren. Allerdings benötigen 4- und 8-Zylindermotoren verschiedene Transferstraßen, an denen verteilt auf 25 hintereinander angeordneten Arbeitsplätzen, aber parallel ablaufenden Arbeitsschritten Tätigkeiten wie Bohrungen, Fräsungen etc. durchgeführt werden. Diese Transferstraße wurde vor 8 Jahren entwickelt, in 3 Jahren steht – aus technologischen Gründen und unabhängig von der jetzigen Kapazitätssituation – der Kauf eines völlig neuen Systems an. Momentan kostet der Kauf einer neuen Transferstraße der alten Technologie 100 MEuro bei einer Herstellzeit von 1.5 Jahren; wegen der geplanten Einführung der neuen Technologie und Verschrottung der existierenden Anlagen bleibt also nur noch eine Nutzungsdauer von 1.5 Jahren. Bei einem Erlös von 7000 Euro/6ZM wird ein Deckungsbeitrag von 2000 Euro/6ZM erzielt, bei jedem 4-Zylindermotor sind es 5000 Euro Erlös und 1000 Euro Deckungsbeitrag. Naheliegend sind die folgenden Maßnahmen zur Deckung der Nachfrage, die mit dem Werksleiter diskutiert wurden:

- A Neukauf einer zusätzlichen Transferstraße,
- B Preiserhöhung der 6-Zylinder Motoren,
- C Werbung für die 4-Zylinder Motoren, und
- D Modifikation des bestehenden Fertigungsprozesses.

Die obigen Daten legen es nahe, zu untersuchen, ob sich der Neukauf einer Straße lohnt. Bei den gegebenen Deckungsbeiträgen und der verbleibenden Nutzzeit von 1.5 Jahren = 480 Tagen ist ein zusätzlicher Gesamtdeckungsbeitrag von $480 \cdot 300 \cdot 2000$ Euro = 288 MEuro zu erwarten. Unter rein finanziellen Gesichtspunkten scheint diese Maßnahme also rentabel. Der Werksleiter lehnt diese Maßnahme A aber mit dem Hinweis ab, es würde seiner Reputation schaden, für diese kurze Zeit noch eine Investition in eine alte Technologie zu tätigen. Der Vorschlag B, vermöge einer Preiserhöhung der 6- Zylindermotoren finanziell immer noch recht positioniert zu sein bzw. den Verkaufschwerpunkt in Richtung 4-Zylindermotoren zu verschieben, missfällt, da er das Preisgefüge auch im Hinblick auf die Marktkonkurrenz zerstört. Die dritte Möglichkeit C missfällt aus ähnlichen Überlegungen, da auf dem Markt nun einmal die 6-Zylindermotoren gefragt sind und 4-Zylindermotoren nur schwer vermittelt werden können.

Bleibt zu hoffen, dass die Analyse des Fertigungsprozesses Lösungsmöglichkeiten aufzeigt. Die Arbeitsschritte 1 und 3 scheinen unkritisch. Der zweite Arbeitsschritt findet am Fließband statt. Es arbeiten etwa 10 Leute an der Transferstraße. Diese beladen die Transferstraße (20 Motoren im Eingangslager) und entladen sie am Ende der Straße (20 Motoren im Ausgangslager). Zwischendurch läuft bis auf kleine Pannen alles automatisch. Allerdings muss in bestimmten Zeitabständen das Werkzeug an der Taktstraße ausgetauscht werden. Diese Zeitabstände hängen insbesondere auch davon ab, mit welcher Taktfrequenz die Fertigung läuft; höhere Taktgeschwindigkeit hat eine stärkere Maschinenabnutzung zur Folge und erfordert eine frühere Ersetzung der Werkzeuge.

Der Engpass liegt in der Tat beim zweiten Arbeitsschritt; der Takt bzw. Durchsatz liegt dort bei $v = 1$ Motor/Minute, d. h. der langsamte der 25 Arbeitsschritte benötigt 60 Sekunden, die anderen Schritte benötigen weniger Zeit. Weitere Befragungen, ob denn dieser langsamste Schritt änderbar sei, werden negativ beantwortet, da dies Modifikationen der Straße erfordere, die jedoch beim Hersteller der Transferstraßen vorgenommen werden müssten - und dies käme etwa dem Kauf einer neuen Transferstraße alter Technologie gleich. Es fällt jedoch folgendes auf: Bei der aktuellen Frequenz v müsste man 1440 Motoren/Tag herstellen können.

Zu fragen ist: Wieviel Stunden wird denn täglich gearbeitet? Es stellt sich heraus: Es werden täglich zwei 8-Stunden-Schichten gefahren – die erste von 6-14 und die andere von 14-22 Uhr und dies 5 Tage/Woche. Leider lassen sich Samstags oder Sonntagsschichten nicht mit dem Betriebsrat vereinbaren. Damit kommt man jedoch immer noch auf $(2/3) \cdot 1440$, d. h. 960 Motoren/Tag, d. h. es fehlen noch 360 Motoren/Tag bzw. bezogen auf die Nachfrage von 960 M/Tag genau 37.5%. Auf die Frage an den Werksleiter, ob es Stillstand und Ausfallzeiten gibt, wird folgende Liste gegeben, wobei sich die Prozentzahlen bzw. Arbeitsstunden auf eine Woche mit 80 Arbeitsstunden beziehen:

Reinigung durch externe Putzfirma (Freitagnachmittags)	$\approx$	5.00%	=	$4.0h$
Störfalle:	$\approx$	13.25% 6.25%	=	$16.0h$
Mitarbeiter kommen zu spät wegen Stau	$\approx$	3.00%	=	$2.4h$
Wartung der Straße zu Beginn jeder Schicht	$\approx$	10.00%	=	$8.0h$

Maßnahmen: Die Reinigung durch die externe Firma kann man auf Samstags verlegen; Ersparnis bereits 5% bzw. 4 Stunden. Die Wartung der Straße – hierfür steht anderes Personal zur Verfügung – nach jeder Schicht kann man in einer Pause durchführen lassen, indem man die zweite Schicht nicht um 14^{00}, sondern um 15^{00} beginnen lässt; gewonnen werden nochmals 10% bzw. 8 Stunden. In dieser Pause kann man bei Bedarf auch schon den Werkzeugwechsel vorgezogen durchführen lassen oder, wenn sich gerade eine stochastische Panne ereignet hat, eine Reparatur ausführen; hiermit ergeben sich im Mittel weitere 13.25% bzw. 10.6 Stunden. Zwar kann die Anzahl der stochastischen Ausfälle durch bessere Wartung gesenkt werden, der quantitative Zusammenhang ist aber nicht direkt ersichtlich; man kann zumindest aber dafür sorgen, dass wichtige Ersatzteile schneller verfügbar sind, oder ein Expertensystem frühzeitig warnt oder schnell diagnostiziert. Mitarbeiter, die zu spät kommen, kann man durch Mitarbeiter aus der 4-Zylinderproduktion ersetzen lassen. Dann wird zwar dort etwas weniger produziert, was aber infolge der geringen Nachfrage keine weiteren Konsequenzen hat. Es lassen sich also auf diese Weise allein aus der Flexibilisierung 31.25% zusätzliche Kapazität gewinnen, ohne das dabei weitere Ressourcen eingebracht werden müssen.

Das Problem besitzt darüber hinaus noch eine nichtlineare Komponente. Mit den 31.25% eingesparten Stillstandzeiten, dies entspricht weiteren 300 Motoren/Tag, erzielt

man eine Produktionsrate von 900 Motoren/Tag. Gleichzeitig erhält man durch die Flexibilisierung eine regelmäßigere und höhere Wartungsbereitsschaft, so dass man die Taktstraße vielleicht noch ein wenig schneller laufen lassen kann; eine Reduzierung des Taktzyklus um nur 3 Sekunden bewirkt eine Kapazitätserhöhung von 960 Motoren/Tag auf 1010 Motoren/Tag. Andererseits führt die bessere Wartung auch zu einer Reduzierung der stochastischen Störfälle. Damit ist das vorgelegte Ziel erreicht.

3 Optimierung in der Praxis

In diesem Kapitel wird beschrieben, in welcher Weise die Modellbildung im gesamten Projektverlauf in den Lösungsprozess eines realen Problems eingebettet ist. Dabei werden einige Aspekte aufgegriffen, die in der Praxis hilfreich sein können, wenn mathematische Optimierung eingesetzt werden soll. Hierzu zählt, das Problem zu erfassen, die Kommunikation und rege Interaktion mit dem Kunden, wobei auch die Verfügbarkeit und Strukturierung der Daten, die Datenhaltung sowie Fragen nach einer möglichen graphischen Benutzeroberfläche geklärt werden müssen. Diese Aspekte sollten vor allem in der Vor- oder Frühphase eines Projektes in die Strukturierung des Projektes einfließen. Im Umfeld der Ergebnisdiskussion werden einige Begriffe wie z. B. *Schattenpreise* und *reduzierte Kosten* konzeptionell und ohne mathematischen Formalismus eingeführt.

3.1 Vorteile durch den Einsatz Mathematischer Optimierung

Mit der Öffnung der Märkte und Grenzen und einer zunehmenden Globalisierung wird die Anzahl und Komplexität von Optimierungsmodellen zur Lösung von Produktionsplanungs- und Distributionsproblemen zunehmen. In der Lage zu sein, vollständige, genaue und optimale Lösungen großer Planungsprobleme zu berechnen, bietet das Potential, erhebliche Kosten zu sparen, die Effizienz zu erhöhen, und sorgfältig mit limitierten Ressourcen umzugehen. Erfahrungen mit dem Einsatz der mathematischen Optimierung führen zu vorsichtigen Ersparnisschätzungen von etwa 3-4%. Insbesondere ist es möglich, in komplizierten Verbundproduktionsnetzwerken Synergien auszunutzen [166].

Die mathematische Optimierung identifiziert ein Problem als unzulässig oder sichert die Zulässigkeit und die Optimalität der Lösung. Sollte der Nachweis der Optimalität zu zeitaufwendig sein, so erhält man aber immerhin eine garantierte und bewertete Lösungsqualität. Auch die Information, dass man mit einer Entscheidung schlimmstenfalls 2% vom bestmöglichen Wert entfernt ist, kann sehr wichtig und beruhigend sein. Mit Hilfe exakter Optimierungsverfahren lässt sich aber nicht nur beweisen, dass es besser als ein bestimmter Maximumswert nicht geht, sondern auch, dass es bei den gegebenen Randbedingungen überhaupt nicht geht – ein wichtiges Argument in mancher Diskussion.

Mit Hilfe der Optimierung lassen sich komplexe Entscheidungsprozesse z. B. über mehrere Standorte hinweg unterstützen. Verschiedene Abteilungen eines Unternehmens, die andernfalls lediglich ihre eigenen Ziele maximieren, können sich mit Hilfe von übergreifenden Optimierungsmodellen konstruktiv auf eine für das Gesamtwohl des Unternehmens passende Zielfunktion einigen. Die für die Optimierung erforderliche ständige Datenpflege erhöht zu dem die Konsistenz der zugrundeliegenden Datenbasis; dies allein

J. Kallrath, *Gemischt-ganzzahlige Optimierung: Modellierung in der Praxis*,
DOI 10.1007/978-3-658-00690-7_3, © Springer Fachmedien Wiesbaden 2013

ist schon ein großer Wert.

Zusammenfassend lassen sich drei signifikante Vorteile der Optimierung festhalten. Zunächst führt die Entwicklung eines Optimierungsmodells zu einem tieferen Problemverständnis; allein dieser Prozess kann für den Kunden schon bedeutsam sein. Zweitens liefert sie ein Werkzeug, das Entscheidungsprozesse quantitativ unterstützt. Drittens erlaubt das Modell die Durchführung von Experimenten. Das in Abschnitt 10.2 besprochene Modell erlaubt, Ideen zukünftiger Planungen zu testen, die am aktuellen Prozess nicht zu überprüfen wären. Die Möglichkeit, einige Prozent der Kosten einzusparen, kann die Entwicklung neuer Geschäftsprozesse beflügeln; Subramanian *et al.* (1994,[268]) berichten in einem Fluglinienmodell von erheblichen Kosteneinsparungen.

3.2 Optimierung ist nicht genug

In diesem Abschnitt[1] sollen einige Aspekte dargestellt werden, weshalb der Einzug mathematischer Methoden in die Praxis von zum Beispiel Supply Chain und Transportlogistik weit langsamer verläuft als die Möglichkeiten es erwünschen lassen. Es werden Vorschläge unterbreitet, wie diese Schwierigkeiten überwunden werden können, darunter einige *eigentlich* altbekannte: Manche mögen es *Interdiszplinarität* nennen, andere einfach nur sagen: *Die Beteiligten müssen miteinander reden* – oder *Wir brauchen Teams von Spezialisten*. Unser Fokus – und damit auch Ziel dieses Abschnitts – ist zu betrachten, was benötigt wird, um ein gegebenes Problem wirklich zu *lösen*. Daher geht es hier weniger um mathematische Optimierung oder Projektsicht, sondern wirklich darum, das Problem im guten Sinne und nachhaltig zu lösen. Exemplarisch wird hier die Logistik zur Verdeutlichung herangezogen, da sie mit ihren beiden Schwerpunkten *Supply Chain Logistik* und *Transportlogistik* einen großen Anteil von Standard-Optimierungsanwendungen abgedeckt.

3.2.1 Hintergrund

Das Finden optimaler Lösungen ist eine der Haupt- und Lieblingsbeschäftigungen bzw. Besprechungsinhalte in vielen Bereichen: Für Mathematik, Operations Research (OR), Betriebswirtschaftslehre und Logistik trifft das sicher zu.

Zweifelsfrei sind die mathematischen und informatischen Möglichkeiten in den letzten Jahren gewaltig gestiegen. Die weiter deutlich abzusehende Zunahme der Rechnerleistungen und die sinkenden Kosten hierfür werden diese Entwicklung fortsetzen.

Während einige Unternehmen wie *Amazon*, *facebook* oder *GOOGLE* sich all diese Entwicklungen sehr erfolgreich zunutze machen, scheinen weite Bereiche in Wirtschaft und Verwaltung hiervon vollkommen ausgeschlossen zu sein. Natürlich gibt es kaum noch Unternehmen, in denen auf den Rechnereinsatz verzichtet wird. Vielfach werden diese aber genutzt, um den herkömmlichen Bleistift durch eine Maus zu ersetzen. Nicht einmal das gute alte Papier ist aus der Mode gekommen. Sowohl die Disposition von Fahrzeugen in Speditionen als auch die Erstellungen von Stundenplänen in Hochschulen oder

[1] Dieser Abschnitt basiert in wesentlichen Teilen auf Jetzke & Kallrath (2012,[150]) Die Integration in dieses Buch erfolgt mit freundlicher Genehmigung von Siegfried Jetzte (Ostfalia Hochschule für angwandte Wissenschaften, Salzgitter) und Dr. Joachim Minnemann (Herausgeber der OR News).

Personaleinsatzplänen in Krankenhäusern werden zwar am Rechner, aber immer noch händisch gemacht. Insbesondere für die Logistik scheinen aber all die aus der mathematischen Optimierung bekannten Methoden doch bestens geeignet. Ob nun Handlungsreisendenproblem oder Packungsalgorithmen, exakt das, was die Logistik braucht.

Trotz aller Fortschritte ist es erschreckend, dass in der öffentlichen Diskussion als ultimative Lösung der Herausforderungen der Logistik stets neue Hardware und Infrastruktur gefordert wird: Mehr Straßen, größere LKWs und folglich mehr Planungsabteilungen, um diese größeren LKWs in der Tourenplanung und im Rahmen weiterer Gesetzesvorgaben adäquat abzubilden. All dieses muss finanziert werden durch geringere Kosten für zu verrichtende Arbeiten und höhere Preise. Ist es wirklich nötig, dieses Geld auszugeben? Könnten nicht auch die vorhanden Mittel reichen?

Es kann möglicherweise erstrebenswert sein, die besten Straßenbauer der Welt in den eigenen Reihen zu haben, allerdings scheint dies angesichts des bereits dichten Straßennetzes und der aktuellen politischen Situation nur begrenzt erfolgversprechend. Sicher ist es erstrebenswert, gute Maschinen für den Straßenbau zu entwickeln. Dieses schafft einige Arbeitsplätze in Deutschland, doch auch hier scheint das Entwicklungspotential durchaus überschaubar. Einige der Länder, die Straßen benötigen, werden sicher auch bald in der Lage sein, die benötigten Maschinen bauen zu können. Wie das Beispiel *Volvo* zeigt, lässt sich das (technische) Erfahrungswissen einschließlich der Infrastruktur mit hinreichendem Kleingeld recht einfach erwerben.

Eine allgemeine Diskussion hierüber soll nicht geführt werden, sondern es soll dargestellt werden, wie wir die Forderung „*Das was wir teurer sind, müssen wir besser sein.*“mit *brain ware* statt *hard ware* verbinden können.

Da sowohl in der wissenschaftlichen Literatur als auch in populärwissenschaftlichen Zeitschriften zur Logistik oder anderen Anwendungsgebieten ausschließlich über erfolgreich abgeschlossene oder erfolgreiche abzuschließende Projekte geschrieben wird, ist es nahezu unmöglich, Gründe für das Scheitern durch Zitate zu belegen. Deshalb sind die folgenden Ausführungen auch als persönliche Meinung und nicht als wissenschaftlich fundierte Abhandlung zu sehen.

Einige Beispiele sind realen Projekten im Umfeld der Supply Chain und Transportlogistik entlehnt. Sie wurden soweit verfremdet, dass sie den Sinn deutlich, aber jeden Rückschluss auf beteiligte Personen unmöglich machen.

3.2.2 Probleme und Lösungen

Eines der grundlegenden Probleme ist in dem letzten Absatz bereits angesprochen: Wann ist ein Projekt erfolgreich abgeschlossen? Aber beginnen wir vorne. Wir werden anhand folgender Punkte die beobachteten Probleme und Lösungsansätze vorstellen:

1. Akzeptanz
2. Transparenz
3. Datenqualität
4. Optimale Lösung
5. Zieldefinitionen
6. Konsistenz von Zielen

7. Bewertung von Alternativen
8. Monitoring – Was muss wann beobachtet werden?
9. Projektfähigkeit
10. Problemlösung und Projekte

Im Folgenden sollen nun diese Punkte im Detail beleuchtet werden.

3.2.2.1 Akzeptanz

Das erste Problem ist einerseits die Akzeptanz der Theoretiker durch die Praktiker und der Praktiker durch die Theoretiker (auf dieses erste Problem wollen wir hier nicht weiter eingehen), das zweite die Akzeptanz der Lösungen.

Lösungen werden nur dann akzeptiert werden, wenn sie gut sind; gäbe es nur eine einfache Möglichkeit zu klassifizieren was *gut* ist. Aber selbst hervorragende Lösungen, die hin und wieder nicht funktionieren und in deren Folge Aufträge unbearbeitet bleiben, werden verworfen werden. Es mag sein, dass für eine Filialbelieferung ohne Optimierung dreißig LKWs benötigt werden und nach einer Optimierung fünf weniger. Wenn dann aber einmal je Woche zehn Filialen nicht beliefert werden können, wird die nichtoptimale Lösung sicher den Vorrang erhalten.

Dieses kann nur verhindert werden, wenn wirklich alle zu beachtenden relevanten (wer bestimmt was relevant ist?) Einflussfaktoren berücksichtigt wurden. Zusätzlich muss der Praktiker verstehen, dass eine wirklich optimale Lösung auf jeglichen Puffer verzichtet. Entweder muss ein solcher Puffer von Beginn an berücksichtigt werden oder nachträglich: Werden fünf LKWs eingespart, könnte ein LKW inklusive Fahrer als Reserve auf dem Hof stehen bleiben. *Als Reserve* bedeutet, dass er bei einer Störung verfügbar ist, d. h. er muss stehen und nichts tun. Dieses *nichts Tun* erscheint dem modernen Zeitgeist unerträglich. Die heutigen Reserven sind als Wartezeiten oder nicht ausgelastete LKWs nicht sichtbar, die neuen hingegen mit bloßem Auge. Hier würde ein Umdenken helfen: Nicht jede Bewegung ist sinnvolle Arbeit; lädt jemand eine Palette auf einen Anhänger, um sie sofort danach wieder abzuladen, wäre faules Schauen effizienter gewesen. Lässt er fünf Minuten verstreichen, damit es nicht so offensichtlich ist, ändert das nichts an der Effizienz.

3.2.2.2 Transparenz

Objektiv gesehen bietet Transparenz viele Vorteile, sie hat aber auch Schattenseiten, die erfolgsgefährdend sein können. Das Definieren von Zielen und Randbedingungen kommt einer Zurschaustellung und Preisgabe von sorgfältigst behütetem Wissen gleich: Die Probleminhaber müssen Fremden ihre Ziele darstellen oder offenbaren, dass diese gar nicht richtig formuliert sind, öffentliche aber auch versteckte Randbedingungen konkretisieren. Das Heft des Handelns liegt hierbei in den Händen der Fragenden. Deren Unwissenheit führt auch zu Fragen, die eigentlich nur Kleinkinder stellen – jede Mutter und jeder Vater weiß, wie peinlich die sein können.

Kurzum, die Situation ist alles andere als angenehm für die Befragten. Das Ergebnis ist dann etwas, was eine Maschine mit ein paar Zeilen Programmcode ausgeworfen hat. Jegliche Freiheit geht verloren, keinerlei kreative Morgenrunde im Alltagsgeschäft, in der für all die nun nicht mehr existenten Probleme des Vortages Verantwortliche gesucht, Auswege diskutiert und Entscheidungen getroffen werden können. Der Entscheider kann nicht mehr durch Entscheidungsstärke glänzen, sein behütetes Wissen für ihn selbst optimal einsetzen und seine glänzende Improvisationsstärke verkümmert. Nicht nur, dass seine unterstützenden Vorschläge überflüssig werden, noch schlimmer, manche stellen sich als kontraproduktiv heraus. Macht und Einfluss über den eigenen Wirkungsbereich nehmen ab zugunsten einer Maschine und wenigen Bits und Bytes und den Herrschern über diese Maschine. Entscheider müssten ihre Rolle neu definieren: Sie könnten sich auf strategische Fragen oder innovative Produkte und Dienstleistungen fokussieren.

3.2.2.3 Datenqualität

Jeder, der schon einmal an irgendeinem Projekt zur Verbesserung gearbeitet hat, wird die mangelhafte Qualität der Daten, die verfügbar sind, festgestellt haben: Daten fehlen bzw. sind nicht auffindbar, sie sind schlicht und ergreifend fehlerhaft oder ihre Quelle ist völlig unbekannt. Wurden Daten aus anderen Daten berechnet, so sind oftmals weder die Grunddaten bekannt noch gibt es Aussagen darüber, wie gerechnet wurde. Für Ergebnisse statistischer Auswertungen existieren möglicherweise noch Angaben über Stichprobenumfang und Varianzen, aber Art und Weise, wie die Stichprobe erstellt wurde, sind Fehlanzeige. Die mit Rechnungen stets verbundene Fortpflanzung von Unsicherheiten scheint gänzlich unbekannt, denn schließlich haben viele im Rechenunterricht der Schule die Grundrechenarten als etwas Exaktes kennengelernt. Moderne Möglichkeiten haben den Glauben an die Exaktheit praktischer Größen noch verstärkt. Überhaupt scheint es, als würden wenige Schlüsselindikatoren in Exceltabellen als vollständiges Abbild der Realität angesehen.

Beispiel 1 *Es soll eine Fahrt von Kiel nach Hannover geplant werden. Die Fahrstrecke beträgt ungefähr* 240 km. *Ein erfahrener Disponent weiß, wie er den Einfluss des Elbtunnels berücksichtigen muss und morgens sicherlich eine andere Zeit einplanen als nachts. Fahren wir von Hannover weiter nach Wuppertal, nochmals ungefähr* 240 km, *verdoppelt sich dann die Fahrzeit? Wenn ja, welche? Die durchschnittliche Fahrzeit?*

Das Betrachten von Durchschnittsgeschwindigkeiten führt sicher nicht zu einem zuverlässigen Ergebnis. Die künstliche Intelligenz bietet vielfältige Möglichkeiten des Lernens, die hier zum Einsatz kommen könnten.

Wir müssen lernen, die mögliche Qualität der Daten und der zu erzielenden Ergebnisse besser zu beurteilen und die Auswirkungen unterschiedlicher Qualitäten deutlich kommunizieren zu können. Die heutzutage verfügbaren Möglichkeiten der Datenerfassung bieten dafür ausreichend Potential.

3.2.2.4 Optimale Lösungen und Randbedingungen

Definition 3.1 Eine optimale Lösung ist die beste Lösung, die innerhalb einer gegebenen Zeitspanne gewonnen werden kann.

Mit dieser Definition ist die optimale Lösung oder das Optimum nicht unbedingt das eingangs erwähnte Maximum oder Minimum. Es sollte zu Projektbeginn von jeder Person, die am Projekt teilnimmt, deren spezifisches Verständnis über das, was eine optimale Lösung ist, abgefragt werden.

Auch wenn wir noch so gute Modelle und Algorithmen einsetzen, wird es sicher mehrere berechtigte Einwände gegen die Lösung geben, z. B., dass nicht alle Zeitfenster bei der Belieferung von Kunden eingehalten werden. Der engagierte Optimierer wird versuchen, diese Einwände zu entkräften, in dem er auch die letzten Vorgaben bedenkt, möglicherweise nicht ahnend, dass er einige Randbedingungen nicht richtig berücksichtigt hat.

Beispiel 2 *Bei der Routenplanung für ein Müllentsorgungsunternehmen war die erzielte Lösung trotz intensiver Bemühungen um mehr als* 10 *% schlechter, gemeint war die Weglänge, als die aus der Praxis bekannte. Nach Durchsicht weiterer, für unwichtig erachteter Unterlagen, wurde die Ursache klar: Die Fahrzeuge kamen bei der Deponie mit einer durchschnittliche Beladung von* 115 *% an. Sie sammelten auf dem Weg – en passant – noch zusätzlichen Müll ein und ersparten sich dadurch zusätzliche Fahrten.*

Es gibt eine Vielzahl von Randbedingungen, die kein Mitarbeiter eines Unternehmens klar und ehrlich öffentlich mitteilen wird. Theoretiker werden es auch nicht schaffen, alle Praktiker dazu zu bewegen, alle Vorgaben bis ins Detail einzuhalten: Soll ein LKW an einem Freitag 5 km vor dem Werktor stehen bleiben, nur weil sein Arbeitszeitlimit erreicht ist und das Entladen auf den kommenden Montag verschieben?

Wir müssen einerseits Möglichkeiten suchen, wie wir gegebene Randbedingungen auf Plausibilität untersuchen und andererseits das Überschreiten von gesetzlichen Vorgaben zulassen. Soll aber der Anbieter einer Optimierungssoftware in seinen Prospekt schreiben, dass – illegale Überschreitungen der Lenkzeit – von seiner Software berücksichtigt werden?

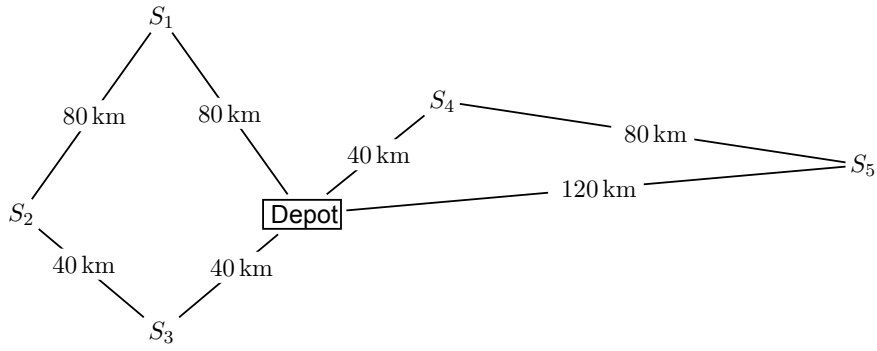
3.2.2.5 Zieldefinitionen

Höchstwahrscheinlich in jedem Management-Lehrbuch ist zu finden, dass die Definition oder Vorgabe von Zielen unverzichtbar für jedes Projekt ist. Dem soll nicht widersprochen werden. Wir brauchen aber *sinnvolle* und *akzeptierte* Ziele. So könnte man sich als Ziel setzen, bei den kommenden olympischen Spielen als Erster und Zweiter die Ziellinie beim Marathon zu überqueren. Dieses ist von der gleichen Qualität, wie zu fordern, dass die durchschnittliche Auslastung eines Lagers oder eines LKWs 100 % betragen soll.

Beispiel 3 *Betrachten wir die Abbildung 3.1. Bei einer Spedition geht eine Anfrage ein, eine Fracht von* 8 *t zum Punkt* S_5 *zu bringen. Dem Spediteur steht ein Fahrzeug mit einer Zuladung von* 12 *t zur Verfügung. Soll er diesen Auftrag annehmen, obwohl das Fahrzeug nur zu ungefähr* 67 *% beladen wäre? Rechnet er die Rückfahrt hinzu, kommt er auf eine Auslastung von einem Drittel.*

Als Königsweg zur Erhöhung der Auslastung wird in jüngster Zeit immer wieder das Einführen sogenannter *milk runs* propagiert. Diese gibt es seit langem in der Landwirtschaft, wo Tankwagen von Hof zu Hof fahren, und dort die frische Milch abholen. Da

Abbildung 3.1 Probleme konkurrierender Ziele beim Einsatz von zwei Fahrzeugen. Die Fahrzeuge schaffen jeweils 40 km je h. Diese Abbildung wurde mit freundlicher Genehmigung von S. Jetzke und J. Minnemann, dem Editor des Magazins OR-News, dem Aufsatz von Jetzke & Kallrath (2012,[150]) entnommen.



das Fahrzeug leer losfährt und im Laufe der Fahrt schrittweise befüllt wird, beträgt auch hier die Auslastung ungefähr 50 %. Die Auslastung ließe sich dann auf 100 % steigern, könnten sie voll losfahren und an jedem Punkt genauso so viel laden wie entladen. Für wirkliche *milk runs* sollten wir uns das Ergebnis besser nicht vorstellen, zu solchen mit anderen Gütern kommen wir später nochmals. Eine andere Quelle der *milk runs* sind amerikanische Filme, in denen jugendliche Fahradfahrer morgens Milchflaschen in Vorgärten werfen: Aber, auch deren Auslastung ist 50 %. Betrachten wir noch einmal die Abbildung. Mit den Punkten $S_1 \to S_2 \to S_3$ könnte ein solcher *milk run* realisiert werden: Wir könnten Waren in vollen Behältern anliefern und leere im Gegenzug mitnehmen, genau so viele wie mitgebracht werden. Bezogen auf das Volumen wäre damit eine gleichmäßige Auslastung gegeben, nicht aber bezüglich des Gewichtes. Was ist denn nun die entscheidende Größe? Volumen oder Gewicht, oder mal so mal so.

Eine tolle Idee, gäbe es da nicht noch die von den Praktikern immer wieder gerne ins Spiel gebrachten Kunden. Diese könnten Vorgaben bezüglich der Zeitfenster machen. Nehmen wir an, dass Fahrzeuge jeweils $40\frac{\text{km}}{\text{h}}$ fahren. Die Zeitfenster von S_2 und S_3 seien gleichzeitig. Nun gibt es einen weiteren Kunden S_2', direkt neben S_2 gelegen, der aber nur halb so viel erhält wie S_2, dafür aber zeitlich in die Tour passt. Soll dieser angefahren werden, oder aber auf die Tour verzichtet werden, weil die Auslastung zu niedrig ist.

Das Thema Ziele verlangt eine weitere sehr intensive Klärung. Ein Grundprinzip, das durchgängig angewendet werden kann, könnte das von Paul Riebel eingeführte *Prinzip der entscheidungsrelevanten Kosten* sein, modifiziert zum *Prinzip des entscheidungsrelevanten Nutzens*. Unternehmenszweck ist selten, Kosten zu senken, sondern Gewinne zu maximieren. Die Anwendung dieses Prinzips ist sowohl einfacher als auch transparenter als die ansonsten aus der BWL bekannten Kostenrechnungsarten. Bei Projekten haben Mitarbeiter verschiedener Hierarchieebenen eventuell unterschiedliche Ziele; die Missachtung dieser explizit oder häufig verdeckten Ziele führt leicht zu Akzeptanzproblemen.

3.2.2.6 Konsistenz

Oftmals sollen mehrere Ziele gleichzeitig verfolgt werden.

Beispiel 4 *Ein Unternehmen liefert Möbel aus. Es sollen optimale Routen bestimmt werden und die Fahrzeuge jeweils* 8 h *unterwegs sein. Das Routenplanungsprogramm ließ die Autos Schleifen fahren.*

Bei optimierter Streckenführung reichten die Aufträge nicht aus, um die Fahrzeuge zeitlich auszulasten. Die Fahrer hatten das vorher geschickt ausgeglichen, z. B. durch Pausen an schwer einsehbaren Plätzen. Das Unternehmen kam dann auf die sehr innovative Idee, die Fahrzeuge mittels *GPS* verfolgen zu lassen. Fahrer wurden erfinderisch. Ein Fahrer erhielt später den Spitznamen *Jonny, der Staufahrer*. Da die Staulängen offensichtlich nicht exakt kalkulierbar waren, kam es schon mal zu - bezahlten - Überstunden.

Kommen wir kurz zurück zu dem oben diskutierten Beispiel des *milk run*. Ein Fahrer hat 8 volle Behälter für S_1. Dort stehen aber nur 6 leere. Soll er jetzt warten, bis zwei weitere verfügbar sind oder weiterfahren, um beim nächsten Kunden pünktlich zu sein. Der Fahrer muss wissen, welches Ziel verfolgt wird: Erst Auslastung und dann Pünktlichkeit oder umgekehrt. Lexikographisches *goal programming* unterstützt multikriterielle Optimierungsprobleme, erfordert aber eine klare Zielhierarchie.

Hier muss klar zwischen unterschiedlichen Ebenen unterschieden werden.

Werden fremde Personen im Rahmen von outsourcing-Projekten eingesetzt, ist es unmöglich, alle Beteiligten über die Ziele zu informieren. Wie soll der Fahrer eines Kleinlieferwagens wissen, dass der Auftraggeber A seinen Kunden einen besonders guten Service anbieten und Auftraggeber B einfach nur die Pakete schnell abliefern möchte. Kann von einem Mitarbeiter eines Dienstleisters erwartet werden, dass er sich auf beide unterschiedlichen Anforderungen einstellen kann? Sollte sein Lohn genauso hoch sein, wie der eines anderen Fahrers, der tagtäglich mit einem Auftraggeber und einer bestimmten Kundenklientel zu tun hat?

Auch das Problem konkurrierender Ziele kann mit dem Prinzip der entscheidungsrelevanten Kosten behoben werden, eine deutliche Nutzenanalyse der Anwender vorausgesetzt.

3.2.2.7 Monitoring – Was muss wann beobachtet werden?

Bisher wurden nur Aspekte betrachtet, die während der Planungsphase relevant sind. Ohne eine Kontrolle der Ergebnisse und ihrer Nachhaltigkeit erscheint der Sinn der Projekte eher fragwürdig. Es wird nur selten gelingen, sofort umsetzbare Ergebnisse zu liefern. Selbst wenn, wird es passieren, dass sich Randbedingungen ändern. Wurde eine für einen Satz von Daten optimale Lösung gefunden, kann jede noch so kleine Änderung diese zerstören. Anwender beginnen, diese Lösung manuell nachzubearbeiten. Die Folge ist eine nicht optimale Lösung. Es wird nach einiger Zeit festgestellt werden, dass die *optimale Lösung* genau so schlecht ist wie die ursprüngliche. Hier sollten all die, die an Optimierungen arbeiten, dazu beitragen, dass optimale Lösungen auch stets besser bleiben als gewöhnliche. Hierzu müssen Veränderungen festgestellt werden, d. h. das laufende System ist zu beobachten. Triviale Kennzahlen, wie durchschnittliche Auslastungen, gefahrene Kilometer, geben sicher keine brauchbare Antwort. Auch das Aufstellen eines Beobachtungsschemas *Versuchsplanung* ist ein Optimierungsproblem, mindestens genauso spannend wie das Problem des Handlungsreisenden oder Packprobleme.

3.2.2.8 Projektfähigkeit

Wie eingangs betont, geht es um die Problemlösung, nicht um Projekte als solche. Da jedoch viele Probleme in Form von Projekten angegangen werden, wollen wir doch darauf einige Gedanken verwenden. Der Autor begleitet Projekte im Praxisumfeld meist aus Sicht der Spezialisten. Eine in den letzten 10 Jahre häufig gemachte Erfahrung ist, dass die mitarbeitenden Persönlichkeiten aus den Anwendungsbereichen die Projektarbeit, die sich leicht über mehrere Monaten bis hin zu einem Jahr erstrecken kann, zusätzlich und neben ihrem Tagesgeschäft, welches sie allein schon zu über 100% auslastet, erledigen müssen – das erhöht die Aussichten auf ein erfolgreiches Projekt nicht wirklich. Projektfähigkeit bedeutet hier, dass der Probleminhaber auch ausreichende Personalresourcen über den gesamten Projektverlauf zur Verfügung stellt. Zur Projektfähigkeit gehört aber auch die Bereitschaft, ehrlich über alle Aspekte zu reden. Stellt sich nur die Frage, wie dies über mehrere Hierarchieebenen hergestellt werden kann. Wird ein Abteilungsleiter dem Planer im Betrieb ehrlich seine Rationalisierungsziele nennen? Wird der Planer ehrlich und ohne negative Konsequenzen sagen dürfen, dass er sich an der internen Datenhaltung vorbei noch einige geheime Tanks hält, mit denen er Fluktuationen in der Produktion besser abfedern kann? Die Projektfähigkeit wird sicherlich erhöht durch eine von allen akzeptierte Verteilung des Gewinns. Ein beeindruckendes Beispiel einer Belohnung durch den Abteilungsleiter nach einem erfolgreichen Projekt für einen besonders engagierten und ehrlichen Planer im Betrieb war die Zuteilung von Sonderurlaub im Äquivalent mehrerer Wochen angehäufter Überstunden; diese Maßnahme erhält Loyalität auf viele Jahrzehnte.

3.2.2.9 Problemlösung und Projekte

Zur Lösung von Aufgaben werden in der Regel Projekte definiert. Zwei der wesentlichen Eigenschaften von Projekten sind deren zeitliche und monetäre Begrenztheit. Im Idealfall wird mit Abschluss des Projektes eine hervorragende Lösung gefunden. Möglicherweise wird die Anlaufphase auch noch innerhab eines Projektes begleitet. Sobald aber die Routine einsetzt, ist das Projekt beendet, sämtliche Unterlagen werden archiviert, Mitglieder des Projektteams gehen ihrer Wege. Es gibt kaum Möglichkeiten, kontinuierlich zu überprüfen, ob die gefundene Lösung noch immer so umgesetzt wird, wie innerhalb des Projektes geplant oder ob der alltägliche Schlendrian viele gute Ergebnisse zunichte macht. Nach einiger Zeit werden sich wieder Probleme zeigen. Was passiert? Das Budget des alten Projektes ist aufgebraucht, das Team verstreut. Also muss ein neues Projekt initiiert werden: Wieder Beantragung von Mitteln, Aufstellen eines neuen Projektteams, neue Ideen und neues Sammeln von Daten. Diese unterscheiden sich von denen, die zum Finden der ursprünglichen Lösung beitrugen. Als Folge werden neue Überlegungen angestellt, die nun wieder genau zu dem führen könnten, was ursprünglich verworfen wurde: So wurden Rundläufe in der Logistik in weiten Teilen durch Direktverkehre ersetzt. Nachdem festgestellt wurde, dass Direktverkehre auch nicht ideal sind, werden *milk runs* eingeführt – Rundläufe mit neuem Namen.

Eine kontinuierliche Begleitung der implementierten Lösung ist unverzichtbar.

3.2.3 Zusammenfassung

Es sind noch eine Vielzahl von kleineren *Problemchen* zu lösen, um Mathematik-basierte Optimierung in die Praxis bringen zu können. Hierzu bedarf es sowohl Mathematiker als auch Informatiker, die sich mit den Praktikern und Probleminhabern zusammensetzen. Auf der Seite der Praktiker und Probleminhaber müssen sowohl diejenigen, die die Abläufe verstehen teilnehmen als auch diejenigen, die Aufwände und Nutzen beurteilen können. Diese müssen unterstützt werden von denjenigen im Unternehmen, die für sich in Anspruch nehmen, die Unternehmensziele klar definieren zu können; Ziele und Wünsche der unteren Hierarchieebenen sollten dabei ernst genommen werden. Die Worthülse „*Wir wollen Beste werden.*“klingt nett, ist aber unbrauchbar. Um dieses zu erreichen, müssen sich die erfahrenen Fachkundigen aus den verschiedenen Bereichen (alteingesessene manuelle Planer im Betrieb, einige Generalisten, die das ganze überblicken, und Spezialisten) zusammensetzen und offene Fragen tabulos klären.

Es könnte wie in einem Orchester sein: Dieses klingt nur dann gut, wenn an jedem Instrument ein Mensch sitzt, der genau dieses beherrscht, alle die selben Noten haben und nach gemeinsamem Start das gleiche Tempo nach Vorgabe des Dirigenten gespielt wird. Bei einem Orchester wird sehr auf die Qualität der Musiker geachtet; diese durchlaufen bis zur Aufnahme in ein A-Orchester eine sehr strenge Auslese. Betreffs der Logistik findet man aber den Glauben vor, dem Kostendruck durch preisgünstigere, sicherlich nicht überqualifizierte Mitarbeiter begegnen zu können. Zum anderen sind Dirigenten meist sehr erfahrene Führungspersönlichkeiten, die in der Regel keinen Vorgesetzen über sich wissen, aber ein sehr tiefes Verständnis der Materie haben. Lässt sich das Konzertbeispiel in die Logistikabteilungen großer Unternehmen übertragen? Oder wird die Logistik dort eher als ein notwendiges Übel und schon gar nicht als Kerngeschäft angesehen? Könnte es nicht ein Wettbewerbsvorteil sein, die *besten Köpfe* nicht nur in Forschung, Entwicklung und Produktion, sondern auch in der Logistik zu haben?

3.3 Die Struktur von Optimierungsprojekten

Im Abschnitt 3.2 wurde auf den Unterschied zwischen Problemlösung und Projekt hingewiesen. Da viele Problemlösungen in Form von Projekten angegangen werden, soll hier nun auf die Struktur von Optimierungsprojekten eingegangen werden. Dabei ist vorab zu bemerken, dass Optimierungsprojekte sich wesentlich von IT-Projekten unterscheiden. Bei einem Optimierungsprojekt liegt das Hauptrisiko zu Beginn des Projektes. Während dieser Phase entscheidet sich meist, ob das Projekt zu einem Erfolg führen kann. Zudem kommt der Autor nach mehr als 23 Jahren zu dem Eregbnis, dass man ein Problem erst wirklich dann verstanden hat, wenn man es gelöst hat. Dies erschwert natürlich die Abschätzung von Zeitaufwand und Kosten erheblich. IT-Projekte im Gegensatz dazu verlaufen viel linearer.

3.3.1 Kontraktierungsphase und Trainingsphase

Das Projekt beginnt mit einer Analysephase des Problems, in der ein strukturierendes Verständnis des Problems erarbeitet wird, das dann Basis eines Angebots werden kann.

Hierzu gehört auch der Aufbau eines geeigneten Projektteams. Nach Annahme des Angebots erweist es sich als vorteilhaft, mit diesem Projektteam – soweit es für die Beiträge und die Rollen der Einzelpersonen im Projekt sinnvoll ist – einen einführenden Schnellkurs in die Denkweise der mathematischen Optimierung durchzuführen.

3.3.2 Strukturierung und Beschaffung der Daten

Bei umfangreicheren Problemen ist es erforderlich, die Quellen für die in ein zu entwickelndes Modell einfließenden Daten zu identifizieren und die Daten zu strukturieren. Dieser Prozess läuft Hand in Hand zu dem Vorgang der Identifikation der in Abschnitt 1.3 eingeführten Modellobjekte. Soweit in Kapitel 1 und 2 und bisher über Daten gesprochen wurde, wurde stets angenommen, dass die in einem Modell verwendeten Daten verfügbar, genau und zuverlässig sind. Stammen die Daten aus einer Datenbank, so mag es möglich sein, ihren Ursprung zu verfolgen und sich zu vergewissern, dass sie aus guten Quellen stammen; dennoch sollte man Daten gegenüber stets kritisch sein und man sollte sich nach Mitchell (1993,[215]) anhand der folgenden Kriterien ein Bild über die Qualität der Daten machen:

1. Der Ursprung der Daten sollte wenn möglich, ihre Eignung für das zugrundeliegende Optimierungsproblem dagegen immer überprüft werden.
2. Es sollte klar präzisiert werden, wie genau die Daten sind und ob die Genauigkeit für die vorliegende Aufgabenstellung hinreichend ist.
3. Unzulänglichkeiten oder begrenzte Genauigkeit der Daten sollten dokumentiert sein; identifiziert man Unzulänglichkeiten, so ist zu überlegen, was in einem solchen Fall zu tun ist.
4. Die Annahmen, unter denen die Daten gewonnen wurden, sollten bekannt gemacht und mit den Anforderungen des Optimierungsmodells verglichen werden. Basieren z. B. die Nachfrageprognosen auf extrapolierten historischen Werten? Wurden Sie anhand eines bestimmten Schema erzeugt? Sind die Produktionskapazitäten geschätzte Werte oder sind sehr genau und zuverlässig? Woher stammen die Zahlen, die den Sicherheitsbestand im Lager quantifizieren? Sind es firmenpolitische Vorgaben oder stammen sie aus Reichweiteabschätzungen?
5. Die Aktualität der Daten sollte von Zeit zu Zeit überprüft werden.
6. Sollten Teile der Daten nur geschätzt sein, so ist auch dies wichtig zu wissen.

Über diese Prüfungen hinaus sollte von Modellierer und Auftraggeber gemeinsam geklärt werden, ob für das Modell noch weitere Daten benötigt werden, und wie in der Zukunft weiter verfahren werden soll:

1. Gemeinsam mit dem Auftraggeber sollten klare Definitionen über die einfließenden Daten, insbesondere betreffs Kosten und Erlöse vereinbart werden.
2. Der Auftraggeber sollte klar spezifizieren, mit welcher Genauigkeit er die Lösungsergebnisse zurück erwartet. Für den Modellierer kann es zu diesem Zeitpunkt noch zu schwierig sein, abzuschätzen, mit welcher Genauigkeit die Ergebnisse erwartet werden können.

3. Aggregationsstufen der Daten und im Modell vorgenommene Vereinfachungen sollten schriftlich festgehalten und dokumentiert werden.
4. Die Verantwortlichkeiten hinsichtlich der Datenpflege nach Beendigung des Projektes sollten geklärt werden, damit das Modell auch in der Zukunft sinngemäß und nutzbringend verwendet wird.
5. Insbesondere ist es hilfreich, klare Regeln zu definieren, wie in der Zukunft die Daten für das Modell aufbereitet werden sollen, und zu klären, wer das Werkzeug einsetzen soll, wie häufig, mit welcher Frequenz und über welchen Zeitraum.

3.3.3 Modellformulierung und Modellierungssprachen

In der Modellbildungsphase wird das Problem in Objekte, Variablen, Zielfunktion und Nebenbedingungen strukturiert. Die Identifizierung der Modellobjekte führt meist direkt zu den Indizes des Modells; hiervon ist wiederum die Datenstruktur betroffen wie in Abschnitt 3.3.2 angesprochen. Sobald die Indizes strukturiert und eingeführt sind, folgen die Variablen, Nebenbedingungen und die Zielfunktion nahezu automatisch. Danach stellt sich die Frage, wie man das Modell einem Rechner zugänglich macht.

Hier finden sich in der Praxis mehrere Vorgehensweisen:

- *Unstrukturierte Ansätze*, bei denen in einer Programmiersprache wie z. B. `Fortran`, `C` oder `C++` das Modell, die Daten und meist auch der Lösungsalgorithmus leider in Form eines Suppeneintopfes vermischt programmiert sind – diesen Ansatz wollen wir wegen der erheblichen Nachteile (Verständnis, Wartbarkeit, Verknüpfung mit Betriebssystemen oder Compilern) und da es kaum möglich ist, hier selbst mathematisch einzugreifen, nicht weiter beleuchten.
- *Tabellarische Ansätze*, bei denen die mathematische Struktur des Optimierungsproblems aus vorgefertigten Tabellen extrahiert wird - dieser eher historische und nicht sehr flexible Ansatz ist in einer Reihe von Softwarepaketen realisiert, die in der Prozessindustrie oder in Raffinerien verwendet werden, allerdings sind diese Ansätze meist auf lineare Optimierungsprobleme oder solche mit festen nichtlinearen Strukturen beschränkt und sollen daher auch nicht weiter betrachtet werden.
- *Algebraische Modellierungssprachen*, die erlauben, ein Optimierungsproblem recht allgemein, flexibel und nahe an der mathematischen Formulierung zu kodieren – derartige Modellierungssprachen sind sicherlich der Königsweg und sollen nachstehend etwas ausführlicher betrachtet werden; ein Beispiel für die Formulierung eines Optimierungsproblem in dieser Form ist auf Seite 257 gegeben.

Einige häufig verwendete algebraische Modellierungssprachen sind in alphabetischer Reihenfolge: `AMPL`[2] [94], `GAMS`[3] [siehe z. B. Broocke *et al.*, 1992, [45])], `LINGO`[4] [260], `MINOPT`[5]

[2] `AMPL` – der Name steht für *A Modeling Language for Mathematical Programming* – wurde in den Bell Laboratories entwickelt und ist nun ein eingetragenes Markenzeichen der Firma Lucent Technologies Inc., 600 Mountain Ave., Murray Hill, NJ, 07974, (http://www.ampl.com/ampl/).

[3] `GAMS` ist ein eingetragenes Markenzeichen der Firma GAMS Development Inc., Washington D.C. (http://www.gams.com).

[4] `LINGO` ist ein eingetragenes Markenzeichen der Firma LINDO Systems, Inc. 1415 North Dayton Street Chicago, IL 60622 (http://www.lindo.com).

[5] `MINOPT` wurde von C. A. Schweiger und C. A. Floudas an der University

[264], MPL[6], NOP[7] [223], NUMERICA[8] [279], Xpress-MP[9] [siehe z. B. Ashford & Daniel (1987, [20]) oder Kallrath & Wilson (1997,[167])]. Wichtig bei diesen Modellierungssprachen ist die klare Trennung zwischen Daten, Modell und Lösungsalgorithmus. Damit ist es auch wie z. B. bei GAMS – der Name steht für *General Algebraic Modeling System* – möglich, verschiedene Lösungsalgorithmen auf ein Problem anzusetzen; ein Besuch der Webseite http://www.gams.com/ zeigt das breite Spektrum der angebotenen Verfahren. Darüber hinaus bieten die meisten Modellierungssprachen Schnittstellen zu Tabellen- oder Datenhaltungssystemen. Neuerlich kommen hinzu OPL[10] [278] und Mosel[11] [131]. Die Modellierungssprache Mosel leitet eine neue Qualität von Modellierungssprachen ein. Sie kann erweitert werden, um spezielle Problemtypen, z. B. nichtlineare Optimierungsprobleme, zu formulieren. Mosel bietet eine offene Schnittstelle, die u.a. als Schnittstelle zu Constraint Programming-Lösungstechniken, Heuristiken oder beliebigen Algorithmen verwendet kann, enthält sowohl Elemente der algebraischen Modellierungssprachen und erlaubt darüber hinaus eigene Programmierung in einer zu Pascal ähnlichen strukturierten Programmiersprache. Die sehr offene Struktur von Mosel bietet elegant die Möglichkeit, selbst eigene Module und Lösungsalgorithmen einzubinden, z. B. die von Opttek Inc. (http://www.opttek.com/) entwickelte, kommerzielle Implementierung von Metaheuristiken darunter die Tabusuche.

3.3.4 Problemgröße, Komplexität und algorithmische Aspekte

Zu Beginn eines Optimierungsprojektes wird häufig ein Prototyp entwickelt, der meist wesentlich kleiner als das vollständige Problem ist, einer ersten Validierung dient und möglicherweise nicht alle strukturellen Merkmale des gesamten Modells enthält.

Kleinere Optimierungs- oder Prototypprobleme beinhalten meist nicht alle strukturellen Eigenschaften, die in praktischen tatsächlich Problemen auftreten; dem Gebot

of Princeton (USA) entwickelt. Die Rechte gehören der Princeton University (http://titan.princeton.edu/MINOPT/minopt.html). Bemerkenswert ist, dass sich mit MINOPT nicht nur MINLP-Probleme, sondern auch gemischt-ganzzahlige dynamische Optimierungsprobleme, die Differentialgleichungen und insbesondere auch Differential-Algebraische Systeme enthalten, formulieren und lösen lassen.

[6] MPL ist ein eingetragenes Markenzeichen der Firma Maximal Software Ltd (http://www.maximalsoftware.co.uk).

[7] NOP wurde von Arnold Neumaier zur Formulierung und Lösung von Problemen entwickelt, die mit Verfahren der Globalen Optimierung gelöst werden sollen (http://www.univie.ac.at/~neum/globt). Diese Modellierungssprache ist eng an die mathematische Struktur des Problems angelegt und verwendet eine Nomenklatur, die der Indizierung dünn besetzter Matrizen entlehnt ist.

[8] NUMERICA wurde von Van Hentenryck zur Formulierung und Lösung von Problemen entwickelt, die mit Verfahren der Globalen Optimierung gelöst werden sollen. In der kommerziellen Version ist NUMERICA eingetragenes Markenzeichen der Firma ILOG Inc. (http://www.ilog.com).

[9] Xpress-MP ist ein eingetragenes Markenzeichen der Firma Dash Optimization (http:/www.dashoptimization.com).

[10] OPL ist ein eingetragenes Markenzeichen der Firma ILOG Inc und basiert bzw integriert Lösungssoftware der Firma ILOG u.a. auch Constraint-Programming Techniken.

[11] Mosel ist ein eingetragenes Markenzeichen der Firma Dash Optimization Ltd. (http:/www.dashoptimization.com).

Du sollst nicht extrapolieren folgend sollte man sehr vorsichtig sein, von dem Prototypproblem auf das eigentliche Problem zu schließen. Bereits in der Modellbildungsphase hat der Modellierer meist schon eine Vorstellung, welche Strukturen und somit welche Algorithmen zum Einsatz kommen könnten. Dabei hilft sein Wissen, dass bestimmte Optimierungsmodelle, insbesondere LP- oder MILP-Probleme meist sehr groß sein und einige zigtausende wenn nicht gar hunderttausende von Variablen und Nebenbedingungen enthalten können, sich dennoch oft in wenigen Minuten lösen lassen - Größe allein ist es eben nicht; Struktur zählt. Der Grund hierfür ist, dass praktische Probleme meist wesentlich mehr Variablen als Nebenbedingungen haben; manchmal liegt hier ein Verhältnis von 10 zu 1 vor. Dennoch stellt sich die Frage, warum LP-Probleme leicht so groß werden und sich dennoch gut lösen lassen.

Zunächst sei geklärt, wie es zu derartig extremen Problemgrößen kommen kann. Ein typisches Modell beinhaltet mehrere Modellobjekte, z. B. in einem Produktionsplanungsproblem Produkte, Anlagen und Zeitscheiben. Seien z. B. $N^P = 50$ Produkte, $N^A = 5$ Anlagen und $N^T = 10$ Perioden zu betrachten, so mag die Entscheidungsfrage lauten: *Wieviel von Produkt P soll auf Anlage A in Periode 4 produziert werden?* Sind alle Kombinationen der Modellobjekte möglich, so ergibt sich die Anzahl der Variablen aus dem Produkt $N^P N^A N^T$ und ist im vorliegenden Fall 2500. Die Anzahl der benötigten Variablen und auch die der Nebenbedingungen wächst also sehr schnell.

Als nächstes wenden wir uns der *Komplexität des Lösungsalgorithmus* zu. Betrachten wir z. B. speziell das in Kapitel 4 näher betrachtete Simplexverfahren zur Lösung von LP-Problemen, so gibt es vier spezielle, mit der algebraischen Struktur von LP-Problemen verknüpfte Gründe, warum dieses Verfahren sehr effizient sein kann.

1. Bei zunehmender, z. B. durch die Anzahl der Variablen oder Nebenbedingungen gemessenen Größe der LP-Probleme, ist zu erwarten, dass die Rechenzeit zunimmt. Die Komplexitätstheorie beschäftigt sich wie auf Seite 303 weiter ausgeführt damit, wie die Anzahl der Rechenoperationen oder die Laufzeit als Funktion der Problemgröße wächst. Dabei ist, wie auf Seite 303 näher erläutert, vor allem die Klasse der $\mathcal{NP}$*-vollständigen* [engl.: $\mathcal{NP}$*-complete*] und $\mathcal{NP}$*-schweren* [engl.: $\mathcal{NP}$*-hard*] Probleme Gegenstand der Betrachtung; hier nimmt die Anzahl der Operationen bzw. die erforderliche Rechenzeit exponentiell in der Problemgröße zu. Während MILP dieser Klasse angehört, wurde von Khachian (1979,[176]) und später Karmarkar (1984,[171]) gezeigt, dass LP-Probleme in polynomialer Zeit gelöst werden können. Das Simplexverfahren[12] kann zwar in extrem konstruierten Fällen exponentielles Wachstum in der Summe $m + n$ aus Anzahl der Nebenbedingungen und Variablen zeigen, aber in der Praxis beobachtet man eher ein lineares Verhalten in $m + n$. Daher ist es nicht überraschend, dass das Simplexverfahren von Dantzig (1963,[63]) in seinen revidierten Formen allen Stürmen der Zeit widerstanden hat und immer noch das am häufigsten verwendete Verfahren in der linearen Optimierung ist.

2. Die meisten LP-Probleme sind *dünn besetzt* [engl.: *sparse, sparsity*], d. h. die Matrix A der Nebenbedingungen $\mathsf{Ax} = \mathbf{b}$ besitzt relativ zu ihrer Größe mit $m \times n$ Einträgen nur wenige von Null verschiedene Koeffizienten; dies tritt bei den kleinen Beispielen

[12] Es ist zu beachten, dass die Zuordnung eines Problems zu einer Problemklasse nur vom Problem, nicht von seiner Darstellung oder der Wahl eines Lösungsalgorithmus abhängt. Für einen Algorithmus, der auf ein Problem angewendet wird, lässt sich dann spezifisch zeigen, ob er polynomiales oder exponentielles Wachstum zeigt.

in diesem Buch nicht hervor. In praktischen Problemen tritt eine Variable beispielsweise nur in etwa 5-10 Nebenbedingungen auf und eine Nebenbedingung enthält im Durchschnitt nur 5-10 Variablen. Lediglich Nebenbedingungen, die eine finanzielle Basis haben, enthalten viele Variablen; alle übrigen nur wenige lokale Variablen. Warum hilft diese Struktur nun aber für das Simplexverfahren? Betrachtet man das Produktionsplanungsbeispiel auf S. 48, so werden viele Nebenbedingungen spezifisch für ein bestimmtes Produkt, eine bestimmte Anlage und Zeitperiode sein; diese Nebenbedingungen repräsentieren als Block ein Untermodell und stehen nicht in Interaktion mit anderen Blöcken. Beispielsweise wird eine Variable, die sich auf London im Juni bezieht, nicht unbedingt gemeinsam mit einer Amsterdam im Mai beschreibenden Variablen auftreten.

3. Treten in einem Problem nicht nur wenige von Null verschiedene Koeffizienten auf, sondern kommen wenige spezielle Koeffizienten, z. B. 1, 2, 2.5, 8, im Vergleich zu anderen sehr häufig vor, so spricht man von *Super-Dünnbesetztheit*.unnbesetztheit

4. Schließlich erlaubt das Simplexverfahren, suksessiv zusammenhängende Probleme mit Hilfe vorheriger optimaler Lösungen als Startlösungen zu lösen.

3.3.5 Ergebnisse der Optimierung und ihre Interpretation

Zu den direkten Ergebnissen eines Optimierungsproblems gehören eine Statusmeldung, mit der der Lösungsalgorithmus terminiert (unbeschränkt, unzulässig, optimal, optimal innerhalb einer bestimmten Genauigkeit), die benötigte Rechenzeit, die Anzahl der Iterationen und andere algorithmusspezifische Informationen. Beispiele hierfür sind in einem B&B-Verfahren

- die Anzahl der benötigten Knoten,
- die Werte der Variablen,
- der Wert der Zielfunktion,
- die *reduzierten Kosten* [engl.: *reduced costs*], diese sind eine Art Ableitung der Zielfunktion nach den Variablen, die den Wert einer oberen oder unteren Schranke annehmen, sowie
- die *Schattenpreise* [engl.: *shadow prices*], die Aussagen über die mit den Nebenbedingungen verbundenen Kosten machen und in einem gewissen Sinne als Ableitung der Zielfunktion nach der rechten Seite der Nebenbedingungen im optimalen Lösungspunkt aufgefasst werden können.

Endet ein Optimierungslauf mit dem Status „*unzulässig*" [engl.: *infeasible*], so wird die Anzahl der unzulässigen Bedingungen bzw. Hinweise auf die Ursache nützlich sein. Im einfachsten Falle ist vielleicht im Modell irrtümlich nur die Richtung ($\leq$,$\geq$) einer Ungleichung vertauscht worden. Schwieriger sind Fälle, in denen Daten falsch eingegeben wurden. Die Meldung „*unbeschränkt*" [engl.: *unbounded*] kann ähnliche Ursachen wie die Meldung „*unzulässig*" haben, oder darauf hindeuten, dass einige Beschränkungen im Modell nicht berücksichtigt wurden. Wurde im gutartigen Falle die optimale Lösung berechnet, so ist zu beachten, dass die Schattenpreise für Ungleichungen, die als Gleichungen realisiert wurden, und die reduzierten Kosten nur bei kontinuierlichen Problemen, nicht aber bei den gemischt-ganzzahligen Problemen Aussagekraft haben.

3.3.5.1 Duale Variablen und Schattenpreise

Mit den Schattenpreisen identisch sind die Werte der *dualen Variablen*. Ist die linke Seite einer $\leq$ Ungleichung z. B. echt kleiner als ihre rechte Seite, so ist die durch diese Ungleichung beschriebene Ressource nicht limitierend und es gibt keinen Wertzuwachs durch eine mögliche Erhöhung der Ressource. Der Schattenpreis, oder der duale Wert einer Nebenbedingung ist in einem Maximierungsproblem der Wertzuwachs der Zielfunktion, der mit einer Einheitserhöhung der rechten Seite der Nebenbedingung einhergeht. Für eine $\geq$ Ungleichung in einem Minimierungsproblem (Maximierungsproblem) gilt sinngemäß entsprechendes und ein positiver dualer Wert bedeutet einen Zuwachs der Zielfunktion (also eine Verschlechterung) bei jeder Einheitserhöhung des Wertes der rechten Seite der Nebenbedingung. Es ist zu beachten, dass es sich bei den Schattenpreisen um ein lokales Konzept handelt. Die durch die dualen Variablen ausgewiesene Änderung der Zielfunktion wird möglicherweise nicht vollständig erreicht, da andere Nebenbedingungen beschränkend wirken. Die dualen Werte lassen sich, wie in Young & Baumol (1960,[300]) ausführlicher diskutiert, konsistent als die Werte der Ressourcen interpretieren.

3.3.5.2 Reduzierte Kosten

Nimmt eine Variable in einem Maximierungsproblem einen positiven Wert an und geht mit einem positiven Koeffizienten in die Zielfunktion ein, so kann sie offensichtlich einen positiven Beitrag zur Zielfunktion leisten; nimmt sie den Wert Null an, so trägt sie nichts bei und ist in einem bestimmten Sinne nicht attraktiv. Allerdings informieren uns die in Abschnitt 4.2.4 näher erläuterten reduzierten Kosten [engl.: *reduced costs*], um welchen additiven Betrag der zugehörige Zielfunktionskoeffizient erhöht werden muss, bevor diese Variable einen positiven Wert annimmt. Alternativ kann man diese reduzierten Kosten als Maß dafür ansehen, um wieviel im Vergleich zu anderen Variablen in einem Maximierungsproblem eine Variable *unterpreist* oder in einem Minimierungsproblem *überpreist* ist; in einem Produktionsplanungsproblem bedeutet unterpreist z. B., dass der Erlös kleiner als die Produktionskosten ist. Die reduzierten Kosten stehen ähnlich wie die Schattenpreise des vorherigen Abschnittes in enger Relation zu den Nebenbedingungen des Problems. Weitere interessante Aspekte im Zusammenhang mit der Betrachtung und Interpretation der Lösung von LP-Problemen finden sich in den vier Artikeln von Greenberg (1993a[117],b[118],c[119];1994,[120]).

3.3.6 Implementierung und Validierung – Vom Modell zum Einsatz

Ist die mathematische Formulierung des Modells abgeschlossen und produziert es zum ersten Mal Lösungen und Ergebnisse, so muss – manchmal hier schon gemeinsam mit dem Probleminhaber – überprüft werden, ob das Modell sich so verhält, wie man es erwarten würde und ob es das ursprüngliche Problem richtig darstellt. Dieser Prozess der Validierung, siehe z. B. auch Abschnitt 10.2.3, sei anhand des folgenden Beispiels erläutert: In einem Produktionsplanungsproblem, bei dem es um die Herstellung von Bier geht, soll abgebildet sein, dass die Herstellung einer Mengeneinheit Bier acht Mengeneinheiten Wasser erfordern. Hier könnte man irrtümlicherweise die inverse Relation (8 Einheiten Bier erfordern 1 Einheit Wasser) implementiert haben. Das Modell wird dann zwar auch arbeiten, aber die Ergebnisse wären wahrscheinlich nicht plausibel.

Daher sind sinnvollerweise zwei Arten von Validierungstests nötig: Zum einen die Klärung, ob der Modellierer das Problem des Auftragsgebers richtig verstanden hat, und zum anderen, ob das richtig verstandene Problem dann auch korrekt umgesetzt wurde. In beiden Fällen hilft eine klare Strukturierung in Unterprobleme oder Untermodelle, die dann einzeln sinnvoll validiert werden können.

Weiterhin ist eine beständige Kommunikation mit dem Auftraggeber wichtig für die Modellvalidierung. Irgendwann geht der Validierungsprozess mehr vom Modellbilder zum Auftraggeber über, wobei der Modellierer noch in engem Kontakt zum Auftraggeber bleibt. Auf Basis von Testdaten wird die Robustheit des Modells getestet und überprüft, ob das Modell sinnvolle Vorschläge unterbreitet, die noch in einer bestimmten Weise mit existierenden Plänen oder realisierten Zuständen kompatibel sind. Wichtig ist auch, eine klare Vorstellung davon zu haben, welche Annahmen in das Modell einfließen, wo seine Grenzen liegen und auch was für eine Lebensdauer es haben soll. Ständige begleitende Überwachung des Modells und sinnvolle Interpretation der Ergebnisse sind notwendig.

3.3.7 Benutzerinterface - Tabellenkalkulation und Datenbanken

Soll der Endnutzer das Planungswerkzeug häufig selbst benutzen, so wird nach einer ersten Implementierungs- und Validierungsphase das Modell und der Lösungsalgorithmus in eine für den Anwender sinnvolle Benutzeroberfläche eingebunden, die es ermöglicht, bestimmte Aspekte der Lösung auch graphisch darzustellen. In vielen Fällen verwenden Benutzer gerne die Tabellenkalkulationssoftware `EXCEL`[13]; mathematische Optimierungsmodelle, die in einer algebraischen Modellierungssprache implementiert wurden, können in `EXCEL` eingebettet – und so in einem gewissen Sinne den Blicken des möglicherweise mathematik-aversiven Benutzers entzogen – werden. Attraktive und aussagekräftige Berichte erleichtern die Interpretation der Lösung und erhöhen die Akzeptanz des Optimierungsmodells. Die Informationen sollten möglichst in der Sprache des Anwenders formuliert werden; *erhöhe die Reaktortemperatur auf 1000*°C ist für den Anwender einfacher verständlich als die Ausgabe `REACT=1000`. Zum Teil unterstützt kommerzielle Optimierungssoftware die Erzeugung von Ergebnisberichten, zum Teil erfordert dies aber auch zusätzlichen Programmieraufwand. Dabei kann es vorteilhaft sein, geeignete Tabellenkalkulationsprogramme, die zum Teil auch limitierte Lösungsroutinen für kleinere Optimierungsprobleme bieten, zu nutzen. Das Programm `What'sBest?`[14] erlaubt Optimierungsprobleme mit Hilfe von Tabellen zu formulieren und verwendet die Optimierungssoftware `LINDO`[15]. `EXCEL`[16] ermöglicht sogar die Lösung kleiner nichtlinearer und gemischt-ganzzahliger Optimierungsprobleme. Derartige Ansätze enthalten nur sehr einfache Algorithmen und sind wegen der Größe der Probleme, die noch sinnvoll behandelt werden können, nur bedingt empfehlenswert.

Alternativ hierzu bieten kommerzielle Optimierungswerkzeuge, so sie denn eine Modellierungssprache enthalten, darunter z. B. `Xpress-MP`, meist eine direkte Kommunikation mit Tabellenkalkulationsprogrammen, z. B. `LOTUS 1-2-3`[17], oder `EXCEL`[18] oder

[13] `MS-EXCEL` ist ein eingetragenes Markenzeichen der Firma Microsoft Corp.
[14] `What's Best?` ist ein eingetragenes Markenzeichen der Firma Lindo Systems Inc.
[15] `LINDO` ist ein eingetragenes Markenzeichen der Firma LINDO Systems, Inc. 1415 North Dayton Street Chicago, IL 60622 (http://www.lindo.com).
[16] `MS-EXCEL` ist ein eingetragenes Markenzeichen der Firma Microsoft Corp.
[17] `LOTUS 1-2-3` ist ein eingetragenes Markenzeichen der Firma Lotus Development Corp.
[18] `MS-EXCEL` ist ein eingetragenes Markenzeichen der Firma Microsoft Corp.

relationalen Datenbanken, z. B. ACCESS[19], dBASE II, III, IV[20], Foxpro[21], Clipper[22] und xBase[23], d. h. insbesondere sind sie in der Lage, die erforderlichen Daten mit Hilfe des ODBC (Open Data Base Connectivity) Standards aus derartigen Programmen zu extrahieren bzw. die Ergebnisse dorthin zu transferieren. Diese Kopplung von Tabellenkalkulation oder relationer Datenbank, oder auch einer Mischung aus beiden Systemen, und der eigentlichen Optimierungssoftware kann zu einer fruchtbaren Symbiose führen, da sie dem Benutzer erlaubt, in seiner vertrauten Umgebung zu arbeiten. Daten und Modell können wohl strukturiert getrennt gehalten und die gewonnenen Ergebnisse graphisch und tabellarisch aufbereitet werden.

Neben der direkten ODBC-Kommunikation ist es gängige Praxis, die Daten aus den genannten Systemen der Optimierungssoftware in Form von ASCII-Dateien zur Verfügung zu stellen und die Ergebnisse umgekehrt als ASCII-Dateien einzulesen. Wichtig ist dabei im Hinblick auf eine gute Wartbarkeit der Software, dass eine strikte Trennung zwischen Daten, Modell und Optimierungsverfahren gewährleistet ist.

Schließlich lassen sich spezielle Optimierungsmodelle und Lösungsansätze auch in die in Abschnitt 3.5 beschriebenen professionellen Planungssysteme einbinden.

3.3.8 Kommunikation mit dem Auftraggeber

Nach Fertigstellung und einer ersten Validierung des Modells, wird die Lösung dem Kunden vorgestellt. Hierbei können unerwartete Ergebnisse auftreten, die für einige Beteiligte negative, für andere positive Überraschungen bringen können. Für den Modellbilder stellt sich die Aufgabe, den Auftraggeber davon zu überzeugen, dass das Modell sich gemäß der vorher vereinbarten Relationen verhält, und gleichzeitig sich davon zu überzeugen, dass keine Informationen verloren oder falsch interpretiert wurden. In diesem Prozess sollte zwischen Auftraggeber und Modellbilder eine enge Vertrautheit erwachsen.

3.3.9 Die Lebensdauer eines Modells

Mehrere Gründe können die Lebensdauer eines Optimierungsmodells beschränken. Zunächst gibt es da alle typischen Probleme jeder komplexeren Software. Der Anwender folgt möglicherweise nicht dem ursprünglichen Sinn des Entwicklers und verwendet Daten, die nicht adäquat sind, und für die das Modell nicht entwickelt wurde. Dies kommt häufig vor, wenn Anwender ohne geeignete Einarbeitung wechseln oder das Modell auf verschiedene Problemfälle des Unternehmens anwenden, die durch das Modell nicht abgedeckt sind. Das Modell wurde möglicherweise für einige Abteilungen getestet, aber für andere, nicht getestete Abteilungen können Überraschungen (unbeschränkte oder unzulässige Probleme) oder Abweichungen von den erwarteten Resultaten auftreten. In einer strengen Validierungsphase [siehe Kapitel 2] ist es Aufgabe des Modellierers, klar die Grenzen des Modells und seine zugrundeliegenden Annahmen zu benennen. Ist das Modell im Einsatz, so kann es schwierig sein, zu überprüfen, ob der Anwender das Modell

[19] MS-ACCESS ist ein eingetragenes Markenzeichen der Firma Microsoft Corp.
[20] dBASE II,III,IV ist ein eingetragenes Markenzeichen der Firma Borland International Inc.
[21] Foxpro ist ein eingetragenes Markenzeichen der Firma Microsoft Corp.
[22] Clipper ist ein eingetragenes Markenzeichen der Firma Computer Associates (CA).
[23] xBase ist ein Standard für Datenbanksprachen mit einem zu dBase III kompatiblen Dateiformat.

sinngemäß nutzt. Was bestenfalls getan werden kann, ist eine genaue Dokumentation zu erstellen und Hilfe anzubieten, wenn Probleme auftreten. Zudem sollten von Zeit zu Zeit neue Anwender mit der Datenstruktur und dem Modell vertraut gemacht werden.

Ein zweiter Punkt betrifft die Ergebnisdaten, ihre Interpretation und die abgeleiteten Konsequenzen. Insbesondere, wenn die Ergebnisse zu automatischen Entscheidungen und Abläufen führen, bedarf es einer ständigen Rückkopplung zu den Modellannahmen. Betrachten wir den Fall, dass zu irgendeinem Zeitpunkt, zu dem schon alle Mitarbeiter, die ursprünglich involviert waren, die Firma längst verlassen haben und eine wahre *black box* Maschine am Werk ist, die z. B. sämtliche Lagerbestandsaspekte kontrolliert, die aber niemand mehr nachvollziehen kann. Dabei kann sich zeigen, dass z. B. die im Modell integrierte Zeitdiskretisierung, z. B. Monatszeiträume, gar nicht mehr adäquat für die praktischen Gegebenheiten ist.

Als dritter Punkt ist zu beachten, dass neue Mitarbeiter, die in dem ursprünglichen Projekt nicht involviert waren, keine oder eine ganz andere Beziehung zu dem Planungswerkzeug haben, die tieferen Annahmen nicht kennen, kein Vertrauen dazu haben und das Werkzeug möglicherweise komplett ablehnen.

Die aufgeführten Punkten legen nahe, dass selbst wenn das Modell von einer externen Unternehmung erstellt und realisiert wurde, eine regelmäßige begleitende Betreuung und überprüfende Validierung des Modells und Planungswerkzeugs sinnvoll ist, auch wenn es schon über Jahre sinnvolle Resultate produziert.

3.4 Algebraische Modellierungssprachen

Wichtig ist bei Algebraischen Modellierungssprachen (AMLs) die klare Trennung zwischen Daten, Modell und Lösungsalgorithmus. Damit ist es auch wie z. B. bei `GAMS` möglich, verschiedene Lösungsalgorithmen auf ein Problem anzusetzen; ein Besuch der Webseite http://www.gams.com/ zeigt das breite Spektrum der angebotenen Verfahren. Darüber hinaus bieten die meisten Modellierungssprachen Schnittstellen zu Tabellen- oder Datenhaltungssystemen.

Modellierungssprachen zur Implementierung mathematischer Optimierungsmodelle müssen die in Welt der mathematischen Optimierung verwendeten Ausdrücke und Symbole unterstützen. Daher ist es natürlich, das AMLs die Entitäten *Daten* (Was ist gegeben?), *Variablen* (Was möchten wir wissen?), *Constraints* (Restriktionen, Schranken, etc.) und *Zielfunktion* (Was möchten wir maximieren oder minimieren?). Daten und Variablen sind nicht nur durch die arithmetischen Operationen (+,-,·,/), sondern auch durch nichtlineare funktionale Relationen verknüpft. AMLs – die ersten, `GAMS`, `LINGO` und `mp-model`, erschienen in den späten 1970er und frühen 1980er – sind deklarative Sprachen zur Implementierung von Optimierungsproblemen. Sie nehmen die Relationen (Gleichungen und Ungleichungen) und Restriktionen zwischen den Variablen auf und verknüpfen Daten und mathematische Objekte und führen diese einem *Solver*[24] zu. Sie enthalten aber keine Informationen darüber, *wie* ein Optimierungsproblem gelöst wird.

[24] In der Gemeinschaft der mathematischen Optimierer, insbesondere unter den Verwendern algebraischer Modellierungssprachen, hat sich der Begriff Solver etabliert als häufig verwendete Kurzbezeichnung für den Lösungsalgorithmus, der als Software zur Lösung eines Optimierungsproblems aufgerufen wird, bzw. für das Programm, in welches der Lösungsalgorithmus einer bestimmten Firma implementiert wurde.

Seit den frühen 1980er spielen AMLs eine wichtige Rolle in der Welt der mathematischen Optimierung und insbesondere beim Einsatz mathematischer Optimierung in der Industrie. In den 1950er und 1960er wurden die meisten in Assembler und Fortran kodierten LP-Modelle durch den Matrixgenerator MPS von IBM ersetzt – dies etablierte einen *de facto* Industriestandard, denn Modelle in jenen Tagen waren LP-Modelle, die mit dem IBM-Solver MPSX gelöst wurden. Zu jener Zeit gab es keinen Markt für AMLs. Aber es gab keine Unterstützung für den Umgang mit NLPs – dies war eine Niche für AMLs, die es dem Anwender ermöglichten, NLP-Probleme zu formulieren und die mit Hilfe *automatischer Differentiation* erste und zweite Ableitungsinformationen bereitstellten. Eine andere Entwicklungslinie wurde durch einen neuen Emporkömmling geschaffen: Den Personalcomputers (PCs). Dash Optimization mit ihrem Solver XPRESS-OPTIMIZER und ihrer AML `mp-model` stellten ein Werkzeug für PC-Benutzer zur Verfügung, ansonsten gab es nur die großen Mainframe-Computer. Nach einer Weile gewannen die AMLs das Rennen mit dem Dinosaurier MPS und LP – nur wenig später auch MILP – wurden in AMLs implementiert.

Heutzutage werden akademische Forschungsmodelle, von Wissenschaftlern entwickelt, verwendet, um Solver zu entwickeln und zu testen oder um effizientere Formulierungen zu finden. Anwendungsmodelle, von Analysten, Beratern und Praktikern entwickelt, werden innerhalb von Beratungsprojekten verwendet. Schließlich findet ein Endbenutzer tief verborgen hinter einem – hoffentlich ansprechenden und gut bedienbaren – graphischen Benutzerinterface für ihn ein Black-Box-Modell, das ihn bei seiner täglichen operativen Arbeit unterstützt. AMLs garantieren Robustheit, Stabilität und unterstützen automatische Datenkonsistenzprüfungen.

Darüber hinaus beschleunigen AMLs die Entwicklung und Verbesserung von Solvern über das gesamt Spektrum von LP- Solver hin zu MINLP-Solvers bis hin zu globalen Optimierungstechniken. Hat ein Nutzer ein NLP-Problem in einer AML implementiert und verwendet einen lokalen NLP-Solver zur Berechnung lokaler Optima, so kostet nur wenige Minuten, stattdessen einen Globalsolver wie BARON oder LINDOGLOBAL zu verwenden. Daher wird das Entwicklungsrisiko für den Nutzer erheblich gesenkt. Auf der anderen Seite können nun die Algorithmenentwickler auf einen viel größeren Markt setzen, wenn ihr Solver in einer AML implementiert ist. Die Solver sind in einem gewissen Sinne eine allgemein und für den Benutzer leicht austauschbare Software geworden – vielleicht zum Leidwesen der Solverentwickler, denn inzwischen stehen auch freie, d. h. kostenlose Coin-OR Solver zur Verfügung Die Implementierung polylithischer Modellierungs- und Lösungsansätze (siehe Kapitel 8) ist in AMLs ohne großen Aufwand möglich.

Bedeutsam ist auch, dass die Entwicklung von Microsoft Windows und verbesserte Hardwaretechnologie zu den Graphischen Benutzer Interfaces wie Xpress-IVE für Mosel, GAMSIDE in `GAMS`, oder Systemen wie `AIMMS` und `MPL` geführt hat. Dies erhöht die Effizienz beim Arbeiten mit AMLs, die sich eher zu Algebraischen Modellierungssystemen entwickelt haben, erheblich und reduziert Projektzeit, Wartungs- und Erhaltungsaufwand und erhöht die Lebensdauer von Optimierungssoftware.

Werfen wir nun einen kurzen Blick auf die heutzutage am häufigsten verwendeten AMLs sind in alphabetischer Reihenfolge:

- `AIMMS`: Historisch betrachtet ist `AIMMS` ein Derivat von `GAMS`, das sich sehr früh für eine graphische Benutzeroberfläche und einer Hinwendung zu Objektorientierter Programmierung entschieden hat. Die Firma Paragon, der `AIMMS` nun gehört, gewann im Jahre 2011 den Edelmann-Preis zusammen mit den Firmen Midwest ISO und Alstom Grid.

- `AMPL`[25]: Robert Fourer und David Gay sind die Eigner und Entwickler von `AMPL`; *cf.* Fourer *et al.* (1993,[94]). Es hat eine dominante Position an Universitäten, wird oft zur Entwicklung von Prototypen verwendet und überrascht immer wieder mit innovativen Ideen.
- `Cplex-Studio`: Früher firmierte dies unter der Bezeichnung *OPL-Studio*, welches von Pascal van Hentenryk entwickelt und dann Teil von ILOG wurde; ILOG und damit auch OPL-Studio ist nun Teil von IBM.
- `GAMS`[26] hat seine Wurzeln in der Weltbank, bei der in den späten 1970er Alexander Meeraus und Jan Bisschop die erste Version entwickelten, bevor dann `GAMS` mit *GAMS Development Corp.* im Jahre 1982 auf dem Markt verfügbar wurde (*cf.* Broocke *et al.* 1992, [45]),
- `LINGO`[27] mit allgemeinen globalen Optimierungsalgorithmen, die nichtkonvexe nichtlineare Modelle in eine Anzahl kleiner konvexer Modelle zerlegt und mit einem B&B-Verfahren verknüpft, um das globale Optimum zu bestimmen, ist insbesondere bei Studierenden sehr beliebt (Schrage 1997,[260]; Cunnigham & Schrage 2004,[54]),
- `LPL`: Alleinig entwickelt von Tony Hürliman (Fribourg University, Fribourg, Schweiz) hat eine Reihe besonderer Eigenschaften.
- `MPL`[28], in den 1980er von Bjarni Kristjansson (Kristjansson & Lee 2004,[185]) hat seine Stärken in der Skalierung, Speicherverwaltung und anderen IT-Aspekten, d. h., Geschwindigkeit, Datenverwaltung und die Applikationsentwicklung von Optimierung.
- `Mosel` ist die Nachfolgesprache von `mp-model`, die von Robert C. Daniel und Robert Ashford entwickelt wurde und von 1983 bis 2001 die mit Xpress-MP (am Anfang stand der LP-Solver LP-OPT, kurz danach wurde er durch den MILP-Solver, MP-OPT erweitert) von Dash Optimization verkauft wurde. Seit 2001 ist Mosel (entworfen und entwickelt von Yves Colombani) die Modellierungsumgebung von Xpress, *cf.* Heipcke (2002,[131]). Die Modellierungssprache `Mosel` leitet eine neue Qualität von Modellierungssprachen ein. Sie kann erweitert werden, um spezielle Problemtypen, z. B. nichtlineare Optimierungsprobleme, zu formulieren. `Mosel` bietet eine offene Schnittstelle, die u.a. als Schnittstelle zu Constraint Programming-Lösungstechniken, Heuristiken oder beliebigen Algorithmen verwendet kann, enthält sowohl Elemente der algebraischen Modellierungssprachen und erlaubt darüber hinaus eigene Programmierung in einer zu Pascal ähnlichen strukturierten Programmiersprache. Die sehr offene Struktur von `Mosel` bietet elegant die Möglichkeit, selbst eigene Module und Lösungsalgorithmen einzubinden, z. B. die von Opttek Inc.

[25] `AMPL` – der Name steht für *A Modeling Language for Mathematical Programming* – wurde in den Bell Laboratories entwickelt und ist nun ein eingetragenes Markenzeichen der Firma Lucent Technologies Inc., 600 Mountain Ave., Murray Hill, NJ, 07974, (http://www.ampl.com/ampl/).

[26] `GAMS` ist ein eingetragenes Markenzeichen der Firma GAMS Development Inc., Washington D.C. (http://www.gams.com).

[27] `LINGO` ist ein eingetragenes Markenzeichen der Firma LINDO Systems, Inc. 1415 North Dayton Street Chicago, IL 60622 (http://www.lindo.com).

[28] `MPL` ist ein eingetragenes Markenzeichen der Firma Maximal Software Ltd (http://www.maximalsoftware.co.uk).

(http://www.opttek.com/) entwickelte, kommerzielle Implementierung von Metaheuristiken darunter die Tabusuche.

- `ZIMPL` ist eine Thorsten Koch am Konrad-Zuse Zentrum in Berlin entwickelte Sprache. Neben `LPL` ist sie die einzige Sprache, die an einer akademischen Einrichtung entwickelt wurde.

Die Leserschaft möge mir verzeihen, wenn hier eine Sprache fehlt. Es ist nicht so, dass heutzutage noch so viele neue entwickelt würden, aber es gibt auf diesem Planeten einfach eine Vielfalt von Modellierungssprachen – und einige von Ihnen habe ich möglicherweise einfach nicht als AML gezählt.

3.4.1 Modelle und Gegensatz zwischen Deklarativ und Prozedural

In Abschnitt 1.3 haben wir die Bedeutung von Modellen beleuchtet. Dabei kamen insbesondere der Zweck eines Modells und der Weg hin zu einem Modell zur Sprache. Hier wollen wir auf den inneren Charakter von Modellen eingehen, wobei wir den Schwerpunkt auf eine für die mathematische Optimierung wichtige Eigenschaft, den deklarativen Charakter, legen.

Ein deklaratives Modell in der mathematischen Optimierung oder Operations Research enthält (algebraische) Constraints, mit deren Hilfe der zulässige Bereich S eines Optimierungsproblems implizit dargestellt und somit die Variablen implizit beschränkt werden. Ein deklaratives Modell sagt uns nichts darüber aus, wie das so repräsentierte Optimierungsproblem gelöst wird. Der deklarative Charakter ist besonders nützlich, da es sich bei S meist um Teilmengen des n-dimensionalen Vektorraumes $\mathbb{R}^n$ handelt, was es ziemlich schwierig macht, alle zulässigen Punkte explizit aufzulisten. Um dies zu illustrieren, betrachten wir das folgende Beispiel: Die Ungleichung $x^2 + y^2 \leq R^2$ in Verbindung mit den Schranken $-R \leq x \leq R$ und $0 \leq y \leq R$ beschreibt als zulässigen Bereich S den oberen Halbkreis mit Radius R, dessen Ursprung mit dem Mittelpunkt des Koordinatensystem der Ebene zusammenfällt. Soll die Zielfunktion $f(x, y) := R/3 - x$ maximiert werden, so müssen wir den Punkt $(x, y) \in S$ zu finden, der $f(x, y)$ maximiert. In diesem einfachen Fall mit nur von x abhängiger Zielfunktion ist es möglich, die Lösung

$$(x, y)_* = \frac{1}{6} R \left(1 - \sqrt{17}, 3 + \frac{1}{3}\sqrt{17} \right)$$

analytisch zu berechnen.

AMLs und ihr deklarativer Charakter eignen sich ideal, um die in Abschnitt 1.8 gelisteten gemischt-ganzzahligen Optimierungsprobleme zu implementieren. Dazu gesellen sich noch speziellere mathematische Optimierungsprobleme wie *Cone Programming*, gemischt-ganzzahlige Komplementaritätsprobleme oder mathematische Optimierungsprobleme mit Gleichgewichtsbedingungen (MPEC).

Imperative *Sprachen* – manchmal auch *prozedurale Sprachen* oder *algorithmische Sprachen*, genannt – mit den prominenten Programmiersprachen Algol, C, Fortran und Pascal repräsentieren ein Modell eines Problems sehr nahe und gerade zu identisch zur Vorgehensweise des Algorithmus, mit dem dieses Problem gelöst wird; *cf.* Hürlimann (2004,[141]). Ein Computerprogramm, welches die Reflektion des Lichtes auf ein Auto mit Hilfe von in die Physik von Strahlungstransport und Fresnelgesetz eingebetteten Monte Carlo Techniken berechnet, könnte daher als Implementierung eines physikalischen Modells für die Reflektion von Licht auf Autolacken angesehen werden.

In ihren Anfängen unterstützen die meisten AMLs lediglich algebraische Gleichungen und Ungleichungen – sie waren vollständig deklarativ. Häufig ist es jedoch bei der Entwicklung mathematischer Optimierungsmodelle, insbesondere wenn man sich mit polylithischen Modellierungs- und Lösungsansätzen (siehe Kapitel 8 oder Kallrath 2011,[162]) beschäftigt, sehr hilfreich, einige prozedurale Sprachelemente zur Verfügung zu haben. Der Umgang mit prozeduralen Strukturen unterscheidet verschiedene AMLs und reflektiert deren innere Architektur. `MPL` und `AMPL` behalten ihren rein-deklarativen Charakter und verwenden die Bibliothek *Optimax* bzw. *Skripttechniken* zur Abbildung prozeduraler Funktionalitäten. Den entgegengesetzten Ansatz sieht man in `GAMS`, `AIMMS` und `LINGO` mit prozeduralen Kommandos wie *loop*, *for*, *if-else*, *while* und anderen, die den deklarativen Befehlen hinzugefügt wurden. `OPL-Studio` (`Cplex-Studio`) und `Mosel` konstruieren ihre Modelle im Wesentlichen prozedural, d. h. das System der Constraints wird prozedural erzeugt.

3.4.2 Ein Beispiel: Deklarativ versus Prozedural

Um die verschiedenen Denkweisen bei deklarativen und prozeduralen Ansätze zu illustrieren, betrachten wir das folgende Beispiel aus der deutschen Fussballbundesliga. Für jedes gewonnene Spiel erhält siegreiche Mannschaft drei Punkte, bei einem Unentschieden bekommen beide Teams und der Verlierer geht mit null Punkten leer aus. Nach G Spielen hat eine Mannschaft P Punkte gesammelt. Mit w, d und l bezeichnen wir die Anzahl gewonnener, unentschiedener und verlorener Spiele. Damit ergeben sich die Relationen

$$\begin{aligned} w+d+l &= G && (3.4.1)\\ 3w+d &= P && (3.4.2)\\ w,d,l &\in N_0 := \{0,1,2,\ldots\} && (3.4.3) \end{aligned}$$

Nun ist es unsere Aufgaben herauszufinden:

1. alle möglichen Kombinationen gewonnener, unentschiedener und verlorener Spiele,
2. die Kombination mit der größten Anzahl gewonnener Spiele und
3. die Kombination mit der kleinsten Anzahl gewonnener Spiele.

Neben der allgemeinen Lösung interessieren wir uns für eine spezifische Probleminstanz mit $G = 14$ und $P = 28$. Lassen Sie uns nun in einer gewollt humoristischen Weise verschiedene Lösungsansätze untersuchen, wie sie Angehörige verschiedener Disziplinen erarbeitet haben könnten:

1. Ein Vertreter der Disziplin *Diskrete Mathematik* würde das Problem auf eine diophantische Gleichung zurückführen und parametrisiert die Lösung durch $k \in Z$ und $(w,d,l) = (w,d,l)(k)$. Eliminierung von d aus (3.4.1) und (3.4.2) führt auf die diophantische Gleichung
$$2w - l = P - G \quad . \qquad (3.4.4)$$
Für $G = 14$ und $P = 28$, (3.4.4) ergeben sich die Lösungen
$$l = 2k \quad , \quad w = 7 + k \quad ; \quad k \in \{0,1,2\} \quad . \qquad (3.4.5)$$
Infolge der Nicht-Negativitätsbedingungen erhalten wir somit die Lösungen

k	$w = 7 + k$	$d = 7 - 3k$	$l = 2k$
0	7	7	0
1	8	4	2
2	9	1	4

(3.4.6)

mit 7 bis 9 gewonnenen Spielen. Elegant – und fertig! Hierfür brauchen wir kein Computerprogramm.

2. Ein Informatikstudent in den Anfängersemestern ohne Mathematikhintergrund könnte dazu tendieren, ein eher langsames, *brute force* Computerprogramm mit drei ineinander geschachtelten Schleifen in w, d und l zu schreiben, wo nacheinander alle Kombinationen darauf hin überprüft werden, ob sie G Spiele und P Punkte ergeben.

3. Studierende der Mathematik würden mit ein wenig Algebra die zusätzlichen Relationen

$$\begin{aligned} w &= (P - G + l)/2 \\ d &= G - w - l = \frac{1}{2}(3G - P - 3l) \end{aligned} \tag{3.4.7}$$

ableiten und wahrscheinlich ein kleines Computerprogramm mit einem effizienten Enumerierungsschema schreiben, welches eine Schleife über l realisiert und die zulässigen Lösungen abgreift. Für $G = 14$ und $P = 28$, führen nur die Werte $l \in \{0, 2, 4\}$ zu zulässigen Kombinationen.

4. Ein Nutzer algebraischer Modellierungssprachen würden wahrscheinlich ein ganzzahliges Optimierungsproblem formulieren mit der Zielfunktion $\max w$ (bzw. $\min w$) und den Constraints (3.4.1) und (3.4.2) sowie der Schranke $w \geq \lceil (P - G)/2 \rceil$. Dieses Problem kann leicht mit MILP-Solvern gelöst werden. Die Solver CPLEX, XPRESS-OPTIMIZER oder BARON erlauben dem Nutzer zudem, alle ganzzahlig zulässigen Lösungen zu bestimmen, was in unserem Problem *alle Kombinationen* ergibt.

Natürlich sind für dieses kleine Beispiel alle vorgestellten Ansätze und Lösungswege geeignet. Für große Werte P und G sieht die Sache aber schon anders aus – hier zahlt sich eine intelligente Lösung hinsichtlich des Laufzeitverhaltens aus.

3.4.3 Modellierung und Studium

In Abschnitt 3.4.2 wurde versucht, die Idee zu vermitteln, dass der Lösungsweg stark vom persönlichen Hintergrund abhängt. Dieser Hintergrund ist geprägt von Ausbildung und Studium, Nähe zu bestimmten Techniken und Erfahrung. In den Ingenieursstudiengängen ist zum Beispiel MATLAB ein fast fester Bestandteil des Lehrplans; daher ist es sehr natürlich, dass Ingenieure in ihrem Berufsleben schnell und leicht zu MATLAB greifen. Mathematiker und Physiker wachsen im Studium mit Fortran oder C und kodieren oft eigene Algorithmen in diesen Sprachen, um z. B. Differentialgleichungen zu lösen. Besondern in den amerikanischen Chemical Engineering Schools, aber auch hierzulande an einigen OR, Agrar- oder volkswirtschaftlichen Studiengängen werden die Studierenden mit `GAMS` vertraut gemacht, wodurch sie auch Kenntnisse und Umgang des deklarativen Ansatzes lernen. In den meisten betriebswirtschaftlichen Studiengängen ist wahrscheinlich MS Excel die Software der Wahl.

Problematisch bei Lehrveranstaltungen mit Inhalt *Modellierung* ist, dass auch hier der Schwerpunkt weniger auf der Modellierung, sondern eher auf der algorithmischen Lösung des Problems liegt. Modellierung, *d. h.* der Prozess der Transformation eines praktischen Problems auf eine mathematische Struktur, wird selten mathematisch gelehrt, weil es nicht Teil des Curriculums ist – und dies ist häufig so, weil diejenigen, die einen Studiengang Mathematik oder Wirtschaft konzipieren, dies nicht für so wichtig erachten. Mathematiker sehen die Modellierung meistens sogar nicht einmal als Teil der Mathematik an – was Friedrich C. Gauß wohl dazu gesagt hätte?

Zur Kunst der Modellierung gibt es aber wenigstens einige empfehlenswerte Bücher: Williams (1993,[291]), Kallrath & Wilson (1997,[167]), Bisschop (1999,[38]), Castillo *et al.* (2002,[50]) und Heipcke *et al.* (2002,[122]). Ich empfehle daher eindringlichst, AMLs in universitären Kursen anzubieten, da dies hilft, den Schwerpunkt auf die Abbildung praktischer Probleme auf mathematische Modelle zu legen, ohne dabei zu viel Zeit bei der Implementierung in prozedurale Sprachen oder Details der Tableau-Simplex-Iterationen zu verlieren; moderne LP-Solver arbeiten ohnehin sehr verschieden von den Textbuchbeschreibungen des Simplex Algorithmus.

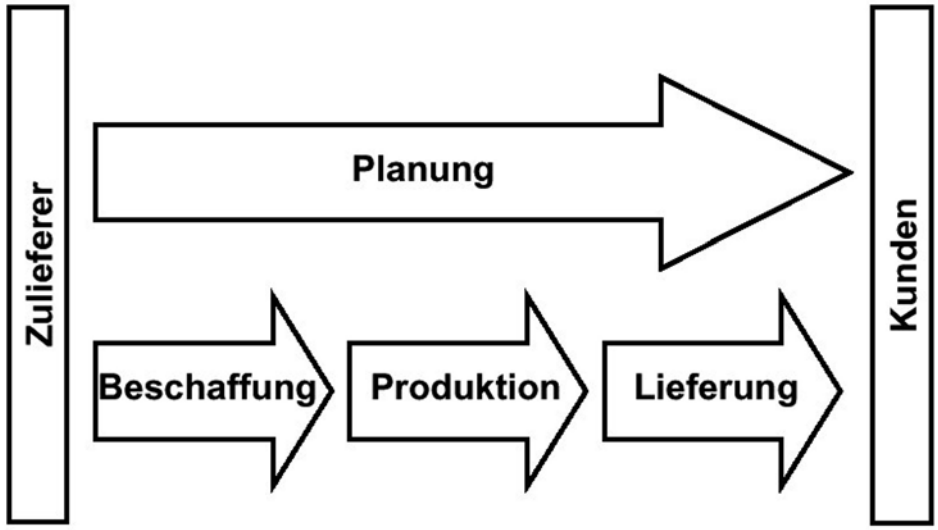
Hinzukommt noch, dass jede AML ihre eigene Anwenderschaft zu haben scheint, die attraktiv auf neue Modellierer ausstrahlt. Dieser *community spirit* hilft enorm: So empfehlen z. B. Agrarwissenschaftler oder Verfahrensingenieure in der Chemie ihren Studierenden gerne die Sprache `GAMS`, da sie selbst diese verwenden. Industrieprojekte mit Beteiligung von Paragon oder IBM verwenden häufig – wenig überraschend – die Sprachen `AIMMS` und `OPL Studio` (`CPLEX Studio`), die von den jeweiligen Firmen betrieben werden. `AMPL` wird an vielen Universitären verwendet, und bei Studierenden ist `LINGO` sehr beliebt, weil es recht einfach benutzbar und sehr preiswert zu erhalten ist. Schließlich gibt es auch Individuen, die mehrere Sprachen je nach Kundenbedürfnissen und Problemeignung verwenden.

3.5 Supply Chain Management und Optimierung

Im vorherigen Abschnitt wurde ein Lösungsweg für Probleme skizziert, der häufig in kleineren oder mittleren Anwendungen von individuellen Entwicklern in Abteilungen größerer Firmen verfolgt wird. Darüber hinaus finden sich gerade für strukturierte Aufgabenstellungen im Supply Chain Management (SCM), etwa in der Produktionsplanung oder Logistik, professionelle Planungssysteme wie z. B. `SAP APO`[29] oder `i2`[30], die mit benutzerfreundlichen Oberflächen die benötigten Daten häufig in Echtzeit aus den Datenbanken von firmeninternen oder -externen Transaktionssystemen oder Enterprise Resource Planning (ERP) Systemen (häufig `SAP R/3`[31]) beziehen. Neben einer Reihe von Funktionalitäten, die nicht direkt mit Optimierung zu tun haben, bieten diese Planungssysteme auch die Möglichkeit zur Supply Chain Optimierung. Da SCM ein großes Feld für komplexe Optimierungsanwendungen bietet, ist dieser Thematik hier ein eigener Raum gewidmet. Daraus soll aber nicht der Schluss gezogen werden, dass dies der Hauptanwendungsfokus der Optimierung oder dieses Buches wäre – viele Themen bezüglich der Rolle von Optimierung im SCM werden hier lediglich überblicksartig angerissen. Für eine

[29] `SAP APO` ist ein eingetragenes Markenzeichen der Firma SAP AG (Walldorf, Deutschland).
[30] `i2` ist ein eingetragenes Markenzeichen der Firma i2.
[31] `SAP R/3` ist ein eingetragenes Markenzeichen der Firma SAP AG (Walldorf, Deutschland).

Abbildung 3.2 SCOR-Modell Ebene 1



tiefere Behandlung dieses Themas sei auf Kallrath & Maindl (2006,[163]) verwiesen.

3.5.1 Supply Chain Management und Supply Chain Optimierung

Als Supply Chain Management, ein Begriff, der 1982 von den Unternehmensberatern Oliver & Webber (1992,[227]) geprägt wurde, kann die Gesamtheit der Logistikplanung eines Unternehmens bezeichnet werden. Sowohl Warenflüsse als auch Informationsflüsse in der gesamten Logistikkette inklusive Zulieferer [engl.: *supplier*], Lohnbearbeiter [engl.: *contract manufacturer*], Distributoren [engl.: *distributor*] und Kunden [engl.: *customer*] werden betrachtet. Die eigentlichen Produktionsprozesse sind ebenso Gegenstand des SCM wie Vereinbarungen mit den Geschäftspartnern. Da der Begriff des Supply Chain Management aus der Betriebswirtschaft stammt, erfolgen Klassifizierungen der Prozesse, Aufgaben und Lösungsansätze auch eher aus betriebswirtschaftlicher und weniger aus mathematisch-naturwissenschaftlicher Sicht. Insbesondere was Optimierungsanwendungen betrifft, ist es eine schwierige und steinige Aufgabe, eine Brücke zwischen diesen so unterschiedlichen Sichtweisen zu schlagen. Während es mathematisch-naturwissenschaftliche Tradition ist, Begriffe mit hoher Präzision zu definieren und zu verwenden, zeigt die Erfahrung mit Projektteams, in der die meisten Projektmitglieder einen betriebswirtschaftlichen oder kaufmännischen Hintergrund haben, dass hier oft kein klarer Konsens über die Verwendung der Begriffe herrscht.

Eine weit verbreitete standardisierte Repräsentation des SCM erfolgt durch das internationale Konsortium Supply Chain Council (SCC) im sogenannten SCOR-Referenzmodell [engl.: *Supply Chain Operations Reference*] (Meyr *et al.* (2000,[210]). Das in unter-

schiedliche Detaillierungsebenen eingeteilte SCOR-Modell standardisiert die Terminologie für SCM-Prozesse, nicht aber die Lösungsmethoden wie zum Beispiel die mathematischen Optimierungsalgorithmen. Abbildung 3.2 zeigt die Elemente der ersten Ebene des SCOR-Modells, das die Prozesstypen Planung, Beschaffung, Produktion und Lieferung [engl.: *Plan, Source, Make, and Deliver*] in ihrem gegenseitigen Verhältnis und in Beziehung zu den Geschäftspartnern darstellt. In jedem dieser Prozesstypen besteht Potential für Optimierungsanwendungen. Ohne Anspruch auf Vollständigkeit seien hier einige Beispiele angeführt:

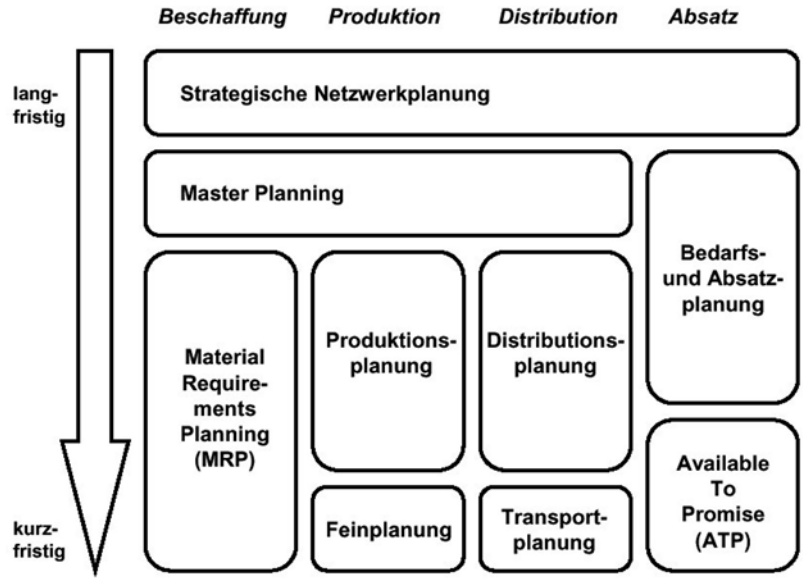
- Planung
  - Langfristige Kapazitäts- und Ressourcenplanung
  - Optimierung des Produktportfolios
- Beschaffung
  - Optimale Wahl der Zulieferer
  - Quotierungsoptimierung
- Produktion
  - Optimale Lokationswahl unter Berücksichtigung von z. B. marktspezifischen und geographischen Merkmalen und Transportwegen, Lohnkosten [engl.: *Supply Chain Design*]
  - Optimale Produktions- und Feinplanung
- Lieferung
  - Bestandsoptimierung in Auslieferungslagern
  - Transportwegoptimierung/Routenplanung [engl.: *Vehicle Routing*]; siehe hierzu Abschnitt 7.6
  - Optimale Ladungskonsolidierung

3.5.2 APS – Advanced Planning Systems

Wie an den Beispielen oben zu sehen ist, umspannen die potentiellen Einsatzgebiete der Optimierung im SCM eine große Bandbreite: Von strategischen Entscheidungen (etwa dem Supply Chain Design), dessen Resultate aufwendig zu implementieren sind und die daher auch über viele Jahre Gültigkeit haben, bis hin zu rein operativen Aufgaben wie der Feinplanung, in der gegebenenfalls in Echtzeit die Auslastung von Produktionslinien und sogar einzelner Maschinen optimiert werden muss. Kommerzielle Softwareprodukte für SCM-Aufgaben dieser Art werden als Advanced Planning Systems (APS) bezeichnet und enthalten meist einige Optimierungsroutinen. In Ergänzung zu den Anbietern reiner Programmbibliotheken oder Optimierungssoftware wie `Xpress-MP` oder `CPLEX`[32] [39] zur

[32] `CPLEX`, jetzt umbenannt in `IBM CPLEX Optimizer`, ist ein eingetragenes Markenzeichen der Firma IBM Inc. (http:/www.ibm.com); vormals ILOG.

Abbildung 3.3 Die Supply Chain Planungsmatrix.



Modellierung und Lösung von Optimierungsproblemen bieten APS ein oder mehrere komplette auf Optimierungssoftware basierende Applikationsmodule, deren Funktionalität in eine Funktionalitätsmatrix, die Supply Chain Planungsmatrix [engl.: *Supply Chain Planning Matrix*, *SCP-Matrix*], eingeordnet werden kann [Rohde *et al.* (2000,[250])]; siehe Abb. 3.3). Im Laufe der 90er Jahre haben sich mehrere größere APS-Anbieter herauskristallisiert, z. B. i2, J. D. Edwards, SAP – siehe z. B. Meyr *et al.* (2000,[249]).

Innerhalb von APS kommt Optimierung zur Anwendung in den Bereichen

- *Strategische Netzwerkplanung* [engl.: *Strategic Network Planning*], bei der es gilt, das Logistiknetzwerk aus Zulieferern, Produktionsstandorten, Distributionszentren und Endkunden optimal zu konfigurieren.
- *Master Planning* [auch *Supply Network Planning*], wobei über das gesamte Netzwerk hinweg die vorhandenen Kapazitäten und daraus resultierende Produktionsmengen optimal abgeglichen werden müssen, um den prognostizierten und/oder bereits verbuchten Kundenbedarf zu befriedigen
- *Produktions- und Feinplanung* [engl.: *Production Planning and (Detailed) Scheduling*], wobei dezentral der aus dem Master Planning kommende Netzwerkplan im Rahmen der Produktionsplanung für die einzelnen Produktionsstandorte und innerhalb der Standorte im Rahmen der Feinplanung auf Linien oder Arbeitsplatzebene in machbare [engl.: *feasible*] kostenoptimale Pläne umgesetzt wird.
- *Distributions- und Transportplanung* [engl.: *Distribution and Transportation Planning*], wo es darum geht, einen optimalen Weg zu finden, die Distributions- und Lagerkosten bei gleichzeitiger Aufrechterhaltung des gewünschten Kundenservices

zu minimieren. Typische Teilaufgaben umfassen Transportwegoptimierung und optimale Ladungskonsolidierung.

- *Verkaufs- und Produktionsplanung* [engl.: *Sales and Operations Planning, SOP*] mit dem Ziel, Nachfragen durch Verkauf zu decken und dabei die eigenen Produktionsmittel so zu nutzen, dass dabei der Deckungsbeitrag maximal wird. Wie in Kallrath (2002,[157]) gezeigt, kann dies eine Kundenportfolioanalyse einschließen.

Die übrigen Aufgaben von APS fallen im Wesentlichen in die Bereiche der Statistik, wie die Bedarfs- und Absatzplanung [engl.: *Demand Planning*], der regelbasierten Algorithmen, wie das Zusagen von Lieferterminen [engl.: *Available To Promise*, *ATP*][33], oder der transaktionalen und/oder regelbasierten Vorverarbeitung für das Master Planning [engl.: *Material Requirements Planning, MRP*].

Bei Anwendung eines APS für Optimierungsaufgaben ist stets zu beachten, dass es sich hierbei um von den Softwareanbietern vorformulierte[34] Modelle handelt, die Module enthalten, die den einzelnen Elementen der SCP-Matrix entsprechen; die Modelle müssen den jeweiligen Anforderungen angepasst werden. Um unangenehmen Überraschungen in der Implementierungsphase vorzubeugen ist es unerlässlich, zwar nicht die Modellformulierung in allen Details zu kennen, aber alle das Modell und die Zielfunktion definierenden Stammdaten und deren Verwendung und Bedeutung im Modell zu identifizieren und die Modellannahmen zu kennen. Im Zuge der Implementierung dieser Systeme kann dabei offenbar werden, dass das Optimierungsmodell nicht oder nur in einem sehr begrenzten Umfang an die spezielle Aufgabe angepasst werden kann. Die Hauptarbeit besteht in der Anbindung an das datenhaltende Transaktionssystem und in der Identifizierung der das Modell bestimmenden Stammdaten.

In der Entscheidungsphase, ob für ein bestimmtes Einsatzgebiet ein APS oder eine komplett aufgabenspezifisch entwickelte Optimierungsanwendung eingesetzt werden soll, sind mehrere Aspekte zu berücksichtigen:

1. Strategisch, taktisch operatives Einsatzgebiet,
2. Lebensdauer des Projekts/Modells und
3. Benutzerschnittstelle.

Diese Aspekte sollen nun nachfolgend ausführlich betrachtet werden.

3.5.2.1 Strategisch, taktisch operatives Einsatzgebiet

Je nach Frequenz der Benutzung der Applikation ergeben sich unterschiedliche Anforderungen an die Integration in bestehende DV-Systeme und Funktionalität und Performanz der Optimierung. So müssen etwa strategische Modelle zur Validierung von Produktionsstandorten und geographischen Lokationen von Distributionszentren sehr genau auf die individuellen Anforderungen der Industriesparte oder des betreffenden Unternehmensbereichs ausgerichtet sein. Derartige Distributionsnetzwerkoptimierung führen häufig auf

[33] Hier kann im Rahmen des CTP [engl.: *Capable To Promise*] allerdings auch ein Produktionsplanungslauf angestoßen werden, der wiederum Optimierung enthalten kann.

[34] In diesem Sinne kann APS-Software als Standardsoftware angesehen werden und die Realität muss zeigen, ob und wieweit das Modell adäquat ist und Problem damit gelöst werden kann, oder ob es besser ist, ein indivuelles Modell zu entwickeln.

MILP-Probleme. Die Daten, die in ein solches Modell fließen, sind meist nicht direkt aus den Datenbanken der ERP-Systeme abrufbar, sondern müssen mit mehr oder weniger Aufwand aggregiert und bereinigt werden, um die Aufgabenstellung zu unterstützen, indem etwa die geplante Schließung von Standorten oder erwartete Strukturänderungen der Märkte reflektiert werden. Andererseits wird für strategische Aufgaben die Echtzeitdatenintegration ins Transaktionssystem wenig bis gar keine Rolle spielen, da für jede strategische Studie, deren Resultate meist über Jahre Gültigkeit behalten, die relevanten Daten in Qualität und Quantität neu definiert werden müssen. Für eine solche Aufgabe ist sehr genau zu prüfen, ob die Funktionalität der ins Auge gefassten APS genügend Flexibilität zulässt, den eigenen Fall abzubilden und auch aussagekräftige Ergebnisse zu produzieren. Zudem spielt bei diesen langfristigen Planung die Lösungsstabilität eine wichtige Rolle; siehe hierzu auch die Bemerkungen zur Sensitivitätsanalyse in den Abschnitten 6.6.2 und 12.3.2.

Operative Anwendungen hingegen bedingen eine enge Integration mit den Datenbeständen des angeschlossenen Transaktionssystems. So stützt sich etwa die mittelfristige standortübergreifende Produktionsplanung (Master Planning in der SCP-Matrix) einerseits auf prognostizierte Kundenbedarf aus dem Absatzplanungsmodul, andererseits auf tatsächliche Kundenaufträge, die nach bestimmten Regeln mit den Prognosedaten verrechnet werden müssen. Da derartige Planungsläufe typischerweise in wöchentlichem Rhythmus ausgeführt werden, ist eine effiziente Schnittstelle zum Transaktionssystem essentiell. Allerdings sind hier mehr Details als bei der strategischen Planung zu berücksichtigen, die den vorgegebenen Modellrahmen möglichweise sprengen. Im Bereich der standortzentrierten Produktions- und Feinplanung muss auf Auftragsänderungen in Echtzeit reagiert werden, was noch größere Ansprüche an Qualität und Stabilität der Schnittstelle stellt. Im Unterschied zu strategischen Aufgaben wird die Optimierung außer an der Qualität der Resultate auch an der Laufzeit gemessen. Insbesondere in der Feinplanung, für die MILP nicht immer gut geeignet ist, existieren daher neben auf LP oder MILP basierenden Ansätze auch solche, die auf Constraint Programmierung, Heuristiken oder genetischen Algorithmen basieren. Ein entscheidender Vorteil der APS sind die meist mitgelieferten Integrationsszenarien, die die Implementierung beschleunigen. Allerdings ist es schwierig und, wenn überhaupt, oft nur mit Entwicklungsaufwand des APS-Herstellers möglich, diese Szenarien zu erweitern oder abzuändern; zudem wird die Formulierung des Optimierungsmodells von den Anbietern nicht im Detail zugänglich gemacht.

3.5.2.2 Lebensdauer des Optimierungsmodells

Dieser Punkt ist eng mit dem Wartungsaspekt [engl.: *support*] der implementierten Lösung verbunden. Gewisse Aufgaben, vor allem strategischer Natur, haben den Charakter von projektgebundenen Einmallösungen, die qualitativ hochwertige Ergebnisse liefern müssen, die dann im Laufe mehrerer Jahre umgesetzt werden. Themen wie Wartbarkeit der Software oder garantiertes Funktionieren bei entsprechenden Aktualisierungszyklen der Transaktions- und Datenbanksysteme spielen eine untergeordnete Rolle, in erster Linie sind die Resultate der sich über einen festgelegten Zeitraum erstreckenden Studie relevant. Die Vorteile der Anwendung von APS über Eigen- oder Auftragsentwicklungen sind hier im vorformulierten Modell zu sehen, das – wenn es zu den Anforderungen passt – Formulierungsaufwand spart, dessen Resultate aber möglicherweise mit teuer eingekauften Beratern der Softwarehersteller interpretiert werden müssen. Anpassungen oder Erweiterungen des Modells sind (fast) nicht möglich, lassen sich in manchen Fällen aber

durch mehr oder weniger umständliche Manipulation an den Quelldaten umgehen.

Grundlegend anders sind die Anforderungen an operative Systeme, etwa für die Produktionsplanung. Neben der oben erwähnten, für Feinplanungssysteme relevanten engen Echtzeitintegration in die Transaktionssysteme müssen operative Anwendungen über viele Jahre hinweg verlässlich funktionieren und entsprechende Aktualisierungszyklen [engl.: *upgrade*] der angeschlossenen Systeme mit minimalem Aufwand mitmachen. Vor allem wenn die Systeme von unterschiedlichen Herstellern stammen, wenn also eine Eigen- oder Spezialentwicklung für die Optimierungsanwendung ausgewählt wird, birgt das Risiken, die weitgehend vermieden werden können, wenn das APS und Transaktionssystem vom selben Anbieter stammen oder eine von beiden Seiten standardisierte Schnittstelle existiert. Kontinuierliche Produktwartung, ein wichtiger Punkt bei der Entscheidung für einen APS-Anbieter oder eine Eigen-/Auftragsentwicklung, ist dann kein Problem, wenn der Anbieter lange genug auf dem Markt ist, diese auch zu gewährleisten.

3.5.2.3 Benutzerschnittstelle

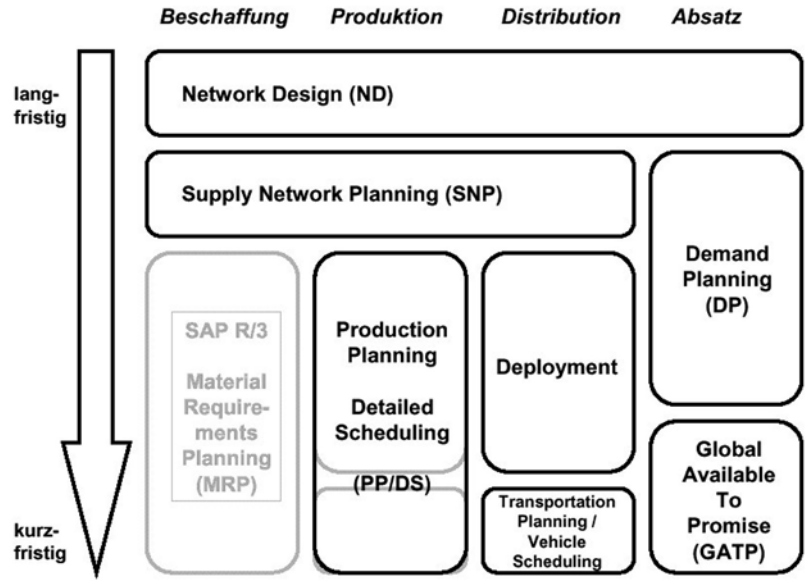
Es ist keine leichte Aufgabe, die Prioritäten Präsentation, Benutzbarkeit für den Anwender [engl.: *usability*] und Funktionalität in einer Lösung optimal zu vereinen. Selbst wenn eine speziell entwickelte Benutzerschnittstelle [engl.: *User Interface*, *UI*, oder *Graphical User Interface*, *GUI*] auf Standardprodukten wie Microsoft Office beruht, entsteht doch ein bisweilen beträchtlicher Programmieraufwand. Kommerzielle APS hingegen werden bereits mit einer Benutzerschnittstelle geliefert, die andererseits nicht notwendigerweise auf die spezielle Anwendung abgestimmt ist. Besonders bei operativen Anwendungen kann es von Vorteil sein und Einführung und Akzeptanz der neuen Lösung beschleunigen, wenn APS und ERP-System für den Anwender ein auf den gleichen Prinzipien beruhendes GUI aufweisen. Das ist meist der Fall, wenn die Systeme vom gleichen Hersteller stammen. Hier sollte man sich aber vor Modeerscheinungen hütten. Wichtiger ist es, die Zielanwender im Auge zu behalten und z. B. zu differenzieren zwischen eher passiven Nutzern und Anwendern oder hochqualifizierten Programmierern? Die gewünschte Einheitlichkeit gibt es bei den verbreiteten GUI-Bibliotheken, z. B. unter Linux vor allem `gnome` und `qt`. Ein zu starke Fokussierung auf Microsoft-Produkte ist also nicht nötig.

3.5.3 SAP APO als Beispiel für ein APS

Von der SAP AG, einem der führenden Anbieter von ERP Software (`SAP R/3`), wird seit 1998 der SAP Advanced Planner and Optimizer (`SAP APO`[35]) angeboten. `SAP APO` wird gemeinsam mit Modulen aus `SAP R/3` und anderen SAP Produkten als `mySAP SCM` Komplettlösung vermarktet, die insbesondere das SCOR-Modell unterstützt und alle Elemente der SCP-Matrix abdeckt. Die SAP Produkte weisen sich durch ein uniformes Design der Benutzerschnittstelle (`SAP GUI`[36]) aus; also präsentieren sich für den Anwender insbesondere das ERP-System `SAP R/3` und `SAP APO` im praktisch selben Erscheinungsbild [engl.: *look and feel*]. In diesem Abschnitt soll, um den Fokus des vorliegenden Buches nicht allzu sehr zu verschieben, `SAP APO` in aller Kürze vorgestellt werden; für weiterführende Behandlungen wird auf Kallrath & Maindl (2006, [163]) verwiesen.

[35] `SAP APO` ist ein eingetragenes Markenzeichen der Firma SAP AG (Walldorf, Deutschland).
[36] `SAP GUI` ist ein eingetragenes Markenzeichen der Firma SAP AG (Walldorf, Deutschland).

Abbildung 3.4 SAP APO Module in der SCP-Matrix



SAP APO besteht aus mehreren Modulen, die Elemente der SCP-Matrix abdecken. Abb. 3.4 zeigt die entsprechenden Module in SAP Terminologie (reines MRP wird von SAP R/3 übernommen). Da eine detaillierte Darlegung der einzelnen Funktionalitäten den Fokus dieses Buches verfehlen würde, erfolgt hier nur eine kurze Aufzählung der SAP APO-Module, in denen *Optimierung*[37] Anwendung findet, mit einer Nennung des mathematischen Lösungsansatzes, der in Ergänzung zu Heuristiken verwendet wird:

- Netzwerk-Design (ND): Optimale Wahl von Standorten im Logistiknetzwerk
  - MILP
- Supply-Netzwerk-Planung [engl.: *supply network planning; SNP*]: Mittelfristige, standortübergreifende Produktions- und Distributionsplanung und Materialnachschub [engl.: *deployment*]: Kurzzeitige Distributionsplanung

[37] Dabei ist wiederum zu beachten, dass der Begriff *Optimierung* hier leider nicht mathematisch exakt und zu dem auch noch etymologisch falsch, sondern wie im betriebswirtschaftlichen Umfeld und insbesondere in der APS Softwareindustrie üblich für jede Art der Anwendung eines (mathematischen) Verfahrens zur Erlangung eines akzeptabel-machbaren oder zur Verbesserung eines vorhandenen Ergebnisses verwendet wird – aus Sicht redlicher abendländischer Kultur eine Unart, die jegliches etymologische Feingefühl vermissen lässt und einen Etikettenschwindel übelster Art darstellt, wie er sich leider im ausgehenden 20sten Jahrhundert mit dem neuen Wertegefüge *appearance statt substance* entwickelt hat. Wie es zu dieser Dominanz angelsächsischer, betriebswirtschaftlicher Strömungen gegenüber alten tradionell abendländischen, philosophisch, naturwissenschaftlich begründeten Denk- und Verhaltenweise kam, wird spannend von De Beuckelaer (2001,[66]) beschrieben.

  - LP und MILP
  - Regelbasierte[38] Algorithmen: Regelfortschreibung

- Produktionsplanung / Feinplanung [engl.: *production planning / detailed scheduling; PP/DS*]: Standortbezogene Produktions- und Feinplanung.
  - Regelbasierte Algorithmen: Regelfortschreibung [engl.: *constraint propagation*]
  - Genetische Algorithmen und CP

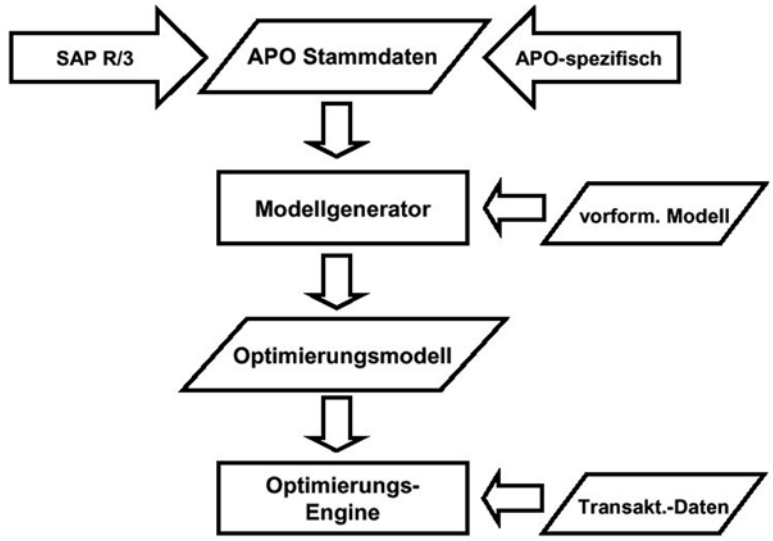
- Routenplanung und Ladungskonsolidierung [engl.: *Transportation Planning / Vehicle Scheduling; TP/VS*] – hier werden nur Heuristiken eingesetzt.

`SAP APO` ist ein selbständiges Softwarepaket, in dem einige betriebswirtschaftliche Optimierungsprobleme, die im SCM auftreten, formuliert und auch gelöst werden können; es bietet allerdings bei weitem nicht die allgemeinen und flexiblen Möglichkeiten wie die in Abschnitt 3.3.3 genannten algebraischen Modellierungssprachen. Bestimmte betriebswirtschaftliche Aufgabenstellungen erfordern allerdings die Integration in weitere Systeme (Auftragserfassung, Kollaborationssysteme mit Zulieferern und Kunden, Transaktionssysteme, etc.), was mittels Standardschnittstellen [engl.: *Business Application Programming Interface, BAPI*] erfolgen kann. Im Spezialfall der Integration mit `SAP R/3` kann das sogenannte Core Interface (CIF) verwendet werden, das bei entsprechender Konfiguration sicherstellt, dass das ERP-System (in diesem Beispiel `SAP R/3`) und das APS (`SAP APO`) in Echtzeit verbunden sind. Wird also etwa in `SAP R/3` ein Verkaufsauftrag erfasst, ist dieser unmittelbar in `SAP APO` sichtbar und fließt in entsprechende Planungsläufe ein; andererseits sind Produktionsaufträge, die in `SAP APO` durch einen Planungslauf erstellt werden, unmittelbar auch in `SAP R/3` verfügbar.

Wie für APS üblich, sind die Optimierungsmodelle in `SAP APO` vorformuliert und die Modellformulierung selbst leider nicht allgemein zugänglich. Außer zur Wahrung von Betriebsgeheimnissen dient dies zur Schaffung von mehr Spielraum für zukünftige Modellerweiterungen ohne bestehende Anwendungen ändern zu müssen, da hierdurch die Anwendung des Optimierungsmodells auf offiziell dokumentierte Eigenschaften beschränkt wird. Ein Beispiel stellt die erst kürzlich hinzugekommene Kampagnenplanung in SNP-PP/DS dar, die aus Anforderungen der chemischen Industrie entstanden ist. Prinzipiell sind allerdings (fast) sämtliche Stammdaten aus `SAP APO` mittels standardisierter Schnittstellen (BAPI, siehe oben) auch in Drittsysteme übertragbar, sodass für spezielle Anforderungen auch ein Benutzen der transaktionalen und regelbasierten Eigenschaften von `SAP APO` mit einem eigenentwickelten Optimierungsverfahren möglich ist. Für die Anwendungsbereiche PP/DS und TP/VS steht auch eine weitergehende Schnittstelle zur Verfügung, die eine Einbindung von Funktionalität und Benutzerschnittstelle von Drittanbieter-Optimierungsanwendungen in `SAP APO` ermöglicht [engl.: *APO Optimization Extension Workbench, APX*]. Beide dieser Möglichkeiten erfordern zwar signifikanten Programmieraufwand, aber bei komplexen Problemstellungen spricht einiges dafür, einen dieser Wege (Drittsysteme, oder APX) zu wählen, denn es ist selbst für einen erfahrenen Modellierer kaum möglich, mit einer *black-box* zu arbeiten. Denkbar wäre natürlich auch, bei Problemen auf `SAP APO`-Entwickler oder -Berater zurückzugreifen, aber diese

[38] Regelbasiert ist hier nicht im Sinne von Expertensystemen gemeint, sondern bezeichnet einfache heuristische Regeln, mit denen eine einfache Planlösung erzeugt werden kann.

Abbildung 3.5 Schematischer Aufbau der Optimierer in SAP APO.



sind nicht nur teuer, sondern dieses Vorgehen bringt auch zusätzliche zeitliche Verzögerungen mit sich. Je komplexer die Fragegestellungen, desto positiver wirken sich sowohl die Vorteile, aber auch die Erfordernisse der mathematischen Optimierung (Verständnis von Modell und Algorithmen, Transparenz, Kompetenz) aus. Eigenentwicklungen bieten hier weiterhin den Vorteil, dass während des Lebenszyklus des Planungswerkzeuges neuere Erfordernisse leicht abgebildet und eingebaut werden können.

Das Optimierungsmodell selbst wird zum einen aus dem vorformulierten Modell und zum anderen aus genau definierten Stammdatenelementen, die direkt aus `SAP APO` und/oder `SAP R/3` stammen, aufgebaut – es sollte stets von einem erfahrenen Modellierer geprüft werden, ob damit die Erfordernisse des Auftraggebers erfüllt werden können, oder ob eine Eigenentwicklung nicht geeigneter wäre. Die Datenübernahme geschieht in einem Vorverarbeitungsschritt im sogenannten *Modellgenerator* (siehe Abb. 3.5). Dieses endgültige Modell wird gemeinsam mit den transaktionalen Daten für die Variablen in dem Optimierungsverfahren übertragen und dort verarbeitet. Das Ergebnis des Optimierungslaufs wird in transaktionale Daten übersetzt und ist direkt in `SAP APO` und im Falle der Integration mit `SAP R/3` auch dort sofort zugänglich. Für technische Auswertungen stehen Protokolle des Modellgenerators und der Optimierungsverfahren zur Verfügung.

3.6 Optimierung und Integration in der Industrie

Im Abschnitt 3.5 haben wir einige Grundlagen zusammengestellt, die sich mit dem Aspekt SCM und Optimierung beschäftigen. Hier wollen wir nun kurz erörtern, zu welchen Konsequenzen die zunehmende Integration in der Industrie geführt hat; es wird sich da-

bei zeigen, dass das Potential der mathematischen Optimierung noch längst nicht ausgeschöpft ist. In den vergangenen Jahren haben sich in allen Industriesegmenten die Anwender damit beschäftigt, flächendeckend ERP-Systeme (Enterprise Resource Planning) einzuführen. In vielen Fällen wurden Eigenentwicklungen dabei durch den Jahrtausendwechsel [engl.: *Y2K hype*] nur zu gerne von den IT Organisationen durch Standardsysteme großer Hersteller ersetzt. Allerdings ist hier Vorsicht geboten. Standards an sich müssen nicht unbedingt ein Vorteil sein.

Zwar wird häufig behauptet, damit gingen deutliche Effizienzsteigerungen einher, aber Standards beenden vielleicht auch den ohne sie stattfindenden Prozess, mit viel Kreativität eigenständige Lösungen zu entwickeln. Der Wert, der mit dem Verlust dieser Fähigkeit einher geht, wird leider nicht betrachtet. Das Hauptaugenmerk liegt nun eben auf Analyse, Harmonisierung und Neudesign bestehender, meist globaler Prozesse, wie sie durch immer globaler auftretende Supply Chains gefordert sind, mit dem Ziel kurzfristig Effizienzsteigerungen zu erzielen. Die alten klassischen Unternehmen, die sich im Weltmarkt behaupten, weichen immer mehr Unternehmensverbünden, die sich immer neu zu einer Supply Chain zusammenfinden. Wechselt die Supply Chain, so wechselt möglicherweise auch der eine oder andere Teilnehmer oder zumindest ändern sich die Rollen und die Frage, wer der Hauptakteur ist.

Solche Supply Chains bergen Themen mit hoher Priorität, wobei es generell immer gilt, alle Elemente der Wertschöpfungskette zu vernetzen, einen durchgängigen Informationsfluss zu schaffen und die Zusammenarbeit einzelner Einheiten zu synchronisieren. Dies reduziert Lagermengen, Produktions-, Bearbeitungs- sowie Lieferzeiten und erhöht deutlich die Lieferbereitschaft bzw. Servicegrad und die Flexibilität.

In einigen Industriesegmenten wie z. B. der chemischen Industrie liegt ein besonderer Fokus auf SRM (Supplier Relationship Management). Da der Anteil von Rohstoff- und Beschaffungskosten an den Herstellkosten der Endprodukte vergleichsweise hoch ist, kommt der Verzahnung von Beschaffungsfunktionen in der Supply Chain eine besondere Bedeutung zu. Durch den Einsatz von SRM ist es möglich, dieses Spannungsfeld gezielt zu harmonisieren. SRM beschreibt sowohl fokussierte Beschaffungsstrategien als auch verbesserte Lieferantenqualifizierungs- und Auswahlprozesse und unterstützt unternehmensweite Standards sowie einen effizienten Beschaffungsprozess.

Durch eine zunehmende Spezialisierung von Transaktionssystemen mit hohem Datenaufkommen einerseits (z. B. `SAP R/3`) und reinen Planungssystemen andererseits (z. B. `SAP APO`) geht eine Konzentration auf sauberen Datenhaushalt und Zusammenführung der für die Planung notwendigen Daten in eigens dafür etablierten Systemen einher. Momentan führen zwar noch technische Probleme wie der Datenaustausch zwischen diesen Systemen und die Identifikation der unterschiedlichen Planungsanforderungen (z. B. im MRP Lauf im `SAP R/3` verbrauchsgesteuerte, im `SAP APO` plangesteuert Disponierung) zu hohem Projektaufwand. Es wird dabei aber als Begleiterscheinung die notwendige Voraussetzung geschaffen, um danach in vielen Bereichen auf mathematische Optimierungsmethoden und Werkzeuge zugreifen zu können und diese sinnvoll in einen großen Kontext einzubinden. Die dabei zu überwindenden Schwierigkeiten sind einerseits technischer Natur (Schnittstellenthematik mit allen möglichen Systembrüchen, oftmals auch über Firmengrenzen hinweg), aber in noch größerem Maße menschlicher Natur, da es im deutschsprachigen Raum, im Gegensatz zum angelsächsischen beispielsweise, bislang keinen richtigen Durchbruch für OR Methoden und mathematische Optimierung gab.

Das Aufkommen von spezialisierten Planungswerkzeugen wie `SAP APO` und deren technischer Integration mit Transaktionssystemen wie `SAP R/3` und der dort schon vor-

gesehenen relativ einfachen Möglichkeit, mit bestimmten Daten in Subsysteme zu verzweigen und dort z. B. in individuell entwickelten Optimierungswerkzeugen spezielle Probleme zu lösen, gibt Grund zu der Hoffnung, dass nun echte Optimierungsansätze auch in betriebswirtschaftlichen Fragestellungen und den entsprechenden Gruppen diskutiert und zunehmend als Lösungsmöglichkeit in Betracht kommen.

3.7 Optimierung in kleinen und mittelständischen Firmen

Sieht man von den einführenden Beispielen in Kapitel 2 ab, so könnten die in Kapitel 6-10 diskutierten Probleme oder Fallstudien in diesem Buch beim Leser den Eindruck erwecken, dass die mathematische Optimierung eher nur für große Firmen und Unternehmungen geeignet ist. Tatsächlich wird die Optimierung häufiger in großen als in kleinen und mittelständigen Firmen angewendet. Dies hat zum Teil damit zu tun, dass letzteren die Möglichkeiten der Optimierung gar nicht klar sind oder die Kenntnisse, das Personal und eventuell der passende akademische Hintergrund nicht zur Verfügung stehen.

Mathematisch gesehen ist es nicht bloß die Tatsache, dass die Modelle eher klein und damit nach unten skalierte Versionen der Modelle sind, die in großen Firmen auftreten. Hinzu kommt noch, dass große Unternehmungen häufig nach Lösungen für integrierte Produktions- oder Distributionsnetzwerke suchen, in kleinen Firmen dagegen oft die optimale Lösung sehr spezifischer und limitierter Probleme bestimmt werden soll.

Um das Bewusstsein und die Akzeptanz der mathematischen Modellierung zu erhöhen, ist es erforderlich, dass der Modellbilder auf spezifische Eigenschaften in kleinen und mittelständigen Firmen eingeht. Dies betrifft insbesondere die Kommunikation mit dem Personal und die Tatsache, dass es eher weniger Entscheidungsebenen gibt, sicherlich aber auch ein Budget, das von einer anderen Größenordnung ist als das, welches großen Unternehmen zur Verfügung steht.

4 Grundlagen der Mathematischen Lösungstechniken

Dieses Kapitel gibt einen einführenden Überblick über den mathematischen und algorithmischen Hintergrund der linearen Programmierung (LP), linearen gemischt-ganzzahligen Programmierung (MILP), nichtlinearen kontinuierlichen Programmierung (NLP), der nichtlinearen gemischt-ganzzahligen Programmierung (MINLP) und schließlich der Globalen Optimierung (GO). Mit diesen Ansätzen lassen sich das Problem (1.8.1) auf Seite 14 oder Spezialfälle davon lösen. Eine strengere und ausführlichere mathematische Behandlung einiger Aspekte ist dem Anhang A vorbehalten. Insbesondere wird der Leser in diesem Kapitel mit den folgenden Themen vertraut gemacht:

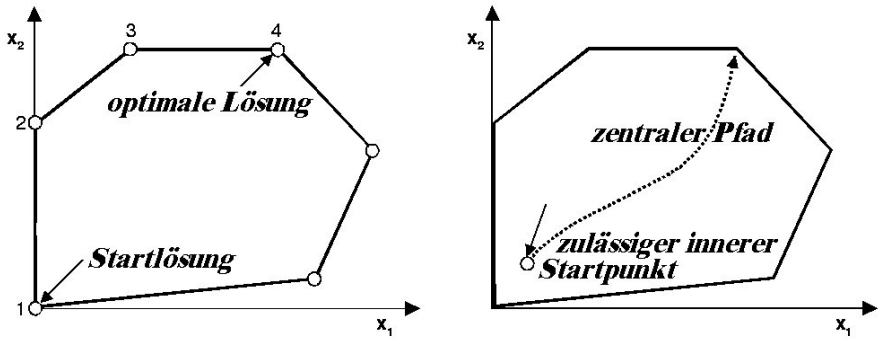
- Standardformulierung von LP-Problemen;
- Gebrauch von Schlupf- und Überschussvariablen;
- Simplexverfahren, Abbruchkriterien und Optimalitätsbeweis;
- Innere-Punkte-Methoden [engl.: *interior point methods, IPM*];
- Branch&Bound-Verfahren zur Lösung von MILP- und MINLP-Problemen sowie einigen grundlegenden Prinzipien bei der Verwendung von Branch&Cut-Verfahren;
- Grundlagen der Dualitätstheorie;
- Interpretation dualer Variablen (Schattenpreise) und reduzierter Kosten;
- Bestimmung zulässiger Lösungen von Optimierungsproblemen;
- notwendige und hinreichende Bedingungen bei NLP-Problemen;
- lokale und globale Optima bei nichtkonvexen Optimierungsproblemen.

4.1 Lineare Optimierung - Lineare Programmierung

Viele praktische Probleme können mit Hilfe der linearen Programmierung gelöst werden. Die Lösung von LP-Problemen ist aber auch relevant für die Lösung anderer Optimierungsprobleme, in denen lineare Probleme als Unterprobleme auftreten. Daher werden in diesem Abschnitt zwei Verfahren zur Lösung linearer Optimierungsprobleme vorgestellt: Das Simplexverfahren und die Inneren-Punkte-Methoden.

J. Kallrath, *Gemischt-ganzzahlige Optimierung: Modellierung in der Praxis*,
DOI 10.1007/978-3-658-00690-7_4, © Springer Fachmedien Wiesbaden 2013

Abbildung 4.1 Illustration des Simplexverfahrens und Innerer-Punkte-Methoden. Beim Simplexverfahren (links) findet in jeder Iteration ein Übergang von einer Ecke zu einer anderen Ecke statt; dabei wird zu Beginn des Verfahren eine Startlösung, die einer Ecke entspricht, benötigt. Bei den Inneren-Punkte-Methoden (rechts) wird – ausgehend von einem zulässigem Startpunkt – eine Folge von Punkten im Innern des zulässigen Bereichs erzeugt.



4.1.1 Das Simplexverfahren — Ein kurzer Überblick

Die bekannteste Methode zur Lösung von LP-Problemen ist wohl das Simplexverfahren. Eine Variante dieses Verfahrens wurde eigentlich schon im Jahre 1939 von A.N. Tolstoi [272] in der damaligen Sowjetunion zur Lösung eines Transportproblems verwendet. Weitere Arbeiten ([169]; [170]), in denen das Verfahren weiter entwickelt und beschrieben wurde, fanden aber im Reich derer, denen die Maximierung des Profits als zu kapitalistisch erschien, kaum Beachtung. Unabhängig von diesen Arbeiten wurde in der westlichen Welt das Simplexverfahren im Jahre 1947 von George B. Dantzig [61] entwickelt; gute Beschreibungen des Verfahrens finden wir in Dantzig (1963,[63]) oder z. B. Padberg (1996,[228]). Wir illustrieren die grundlegenden Ideen des Verfahrens auf einem elementaren Niveau im Abschnitt 4.2.4. Der erste Schritt besteht darin, wie im Abschnitt A.1.2 beschrieben, eine zulässige Startlösung zu berechnen. Hierzu wird meist ein Hilfsproblem – ebenfalls ein LP-Problem – gelöst, dessen optimale Lösung eine zulässige Lösung des ursprünglichen Problems ist oder die anzeigt, dass das ursprüngliche Problem keine Lösung besitzt. Das Simplexverfahren bzw. die in der Praxis häufiger verwendete und in den meisten kommerziellen Softwarepakten implementierte effizientere Variante des *revidierten Simplexverfahrens* bestimmt eine optimale Lösung eines LP-Problems nach einer endlichen, möglicherweise aber exponentiell in der Anzahl der Variablen und Nebenbedingungen anwachsenden Anzahl von Iterationsschritten. In der Praxis sind LP-Probleme jedoch meist gutartiger und mit dem Simplexverfahren recht schnell zu lösen.

4.1.2 IPM — Ein kurzer Überblick

Der zulässige Bereich entspricht in einem linearen Problem einem Polyeder. Bei nicht entarteten Problemen iteriert das Simplexverfahren im geometrischen Sinne wie in Abb.

4.1 (links) illustriert entlang einer Kante bzw. Hyperfläche von Ecke zu Ecke.

Im Gegensatz dazu iterieren IPM im Innern des Polyeders [siehe Abb. 4.1 (rechts)]. Im Zusammenhang mit der linearen Optimierung erfreuen sich diese Verfahren seit der Arbeit von Karmarkar (1984,[171]) großer Aufmerksamkeit (Freund & Mizuno, 1996, [95]). Die IPM haben ihre Wurzeln in der nichtlinearen Optimierung und sind intuitiv einfach zu verstehen, aber es sind einige Besonderheiten im Zusammenhang mit der linearen Optimierung zu beachten. Die Lösung liegt (in nicht entarteten Fällen) in einer Ecke, also auf dem Rand des zulässigen Bereichs und nicht im Innern. Desweiteren hat das Polyeder des zulässigen Bereichs bei praktischen Problemen weniger eine kugelnahe, sondern eher eine längliche Gestalt, so dass es einiger Sorgfalt bedarf, den zulässigen Bereich nicht zu verlassen. Dies wird erreicht, indem jede Annäherung an den Rand des zulässigen Bereichs dynamisch bestraft wird. Da das Verfahren nur innere Punkte erzeugt, ist es irgendwann erforderlich, ein z. B. von Andersen & Ye (1994,[15]) entwickeltes Umschaltverfahren [engl.: *cross-over schemes*] bzw. Pivotisierungsverfahren [engl.: *„purification" pivoting procedures*] zu starten, das aus einem inneren Punkt, der den Abbruchkriterien der IPM genügt, eine naheliegende Ecke und eine Basislösung generiert, deren Zielfunktionswert nicht schlechter ist als der des besten inneren Punktes.

Die kommerziell verfügbaren Algorithmen basieren auf den im Anhang A.3 beschriebenen *logarithmischen Barriere-Verfahren* [engl.: *logarithmic barrier methods*], allerdings ist neuerlich ein Trend zu homogenen und selbstdualen[1] Verfahren [engl.: *self-dual solvers*] erkennbar. Diese Barriere-Verfahren werden gewöhnlich Frisch (1955,[96]) zugeschrieben und wurden formal von Fiacco & McCormick (1968,[82]) im Zusammenhang mit nichtlinearen Optimierungsproblemen untersucht, also lange Zeit vor Karmarkar. Gill *et al.* (1986,[103]) konnten zeigen, dass es einen formalen Zusammenhang zwischen Karmarkars neuen IPM und den klassischen logarithmischen Barriere-Verfahren gibt.

Da die IPM einige tiefergehende Mathematik erfordern, werden sie ausführlicher im Anhang A.3 beschrieben.

4.1.3 Lineare Programmierung als Unterproblem

Die Lösung von LP-Problemen ist relevant für die Lösung anderer Optimierungsprobleme, da diese als Unterprobleme auftreten in der

NLP	nichtlinearen Programmierung	SLP-Verfahren
ILP & MILP	reinen und gemischt-ganzzahligen LP	LP-Relaxierungen
MINLP	gemischt-ganzzahligen NLP	Outer-Approximation

Da in diesen Problemen mehrere Tausende oder auch Millionen von LP-Problemen gelöst werden, lohnt es sich darauf zu achten, dass das Modell so formuliert ist, dass die Rechenblöcke des Simplexverfahren[2]

- *Pricing-out* oder kurz *Pricing*,

[1] Derartige Verfahren werden z. B. in dem Übersichtsartikel von Freund & Mizuno (1996,[95]) diskutiert. Sie beruhen auf einigen Eigenschaften selbstdualer Optimierungsprobleme; hierbei ist das duale Probleme gerade wieder das primale Ausgangsproblem. Weitere Details finden sich in Andersen & Ye (1995,[16]) und Vanderbei (1996,[282]).

[2] Bei Verwendung von IPM gelten entsprechende, auf die mathematischen Eigenschaften dieser Methode zugeschnittene Empfehlungen.

- Eliminierung einer existierenden Basisvariablen und
- die Operationen der linearen Algebra (pivoting)

so effizient wie möglich durchgeführt werden können. Hierbei helfen die folgenden Punkte:

- effizientes *Pricing* profitiert von einer Zielfunktion mit klaren Sensitivitäten;
- Gleichungen bzw. größere Blöcke mit verschwindender rechter Seite sollten möglichst vermieden werden, da dies zu Schwierigkeiten bei der Eliminierung der Basisvariablen führen kann;
- je dünner besetzt die Systemmatrix und je geringer ihre Dichte, desto besser und um so effizienter die lineare Algebra.

4.2 Elementare Erläuterung des Simplexverfahrens

Im Abschnitt 2.1 wurde als Beispiel das „Bootsverleihproblem" in der Form

$$\max_{p,s} z \quad , \quad z = z(p,s) = 800p + 600s \tag{4.2.1}$$

unter den Nebenbedingungen

$$\begin{array}{rcrcr} p & + & s & \leq & 350 \\ p & & & \leq & 200 \\ p & - & s & \geq & 0 \\ 4p & + & 3s & \leq & 1400 \end{array} \tag{4.2.2}$$

vorgestellt mit der zusätzlichen impliziten Bedingung, dass die Variablen p und s keine negativen Werte annehmen dürfen.

4.2.1 Standardformulierung linearer Optimierungsprobleme

Um ein lineares Optimierungsproblem einem allgemeinen mathematischen Lösungsverfahren zugänglich zu machen, ist es sinnvoll, eine Standardformulierung[3] und -notation mit Variablen x_i und Daten A_{ij}, b_i und c_i einzuführen:

$$\max_{x_1,\ldots,x_n} z \quad , \quad z = z(x_1,\ldots,x_n) = c_1x_1 + c_2x_2 + \ldots + c_nx_n$$

unter den Nebenbedingungen

$$\begin{array}{rcl} A_{11}x_1 + A_{12}x_2 + \ldots + A_{1n}x_n & = & b_1 \\ A_{21}x_1 + A_{22}x_2 + \ldots + A_{2n}x_n & = & b_2 \\ & & \vdots \\ A_{m1}x_1 + A_{m2}x_2 + \ldots + A_{mn}x_n & = & b_m \\ x_1, x_2, \ldots, x_n & \geq & 0 \quad . \end{array}$$

[3] Natürlich sind auch andere Standardformulierungen möglich. Im Anhang A.1.3 werden Standardprobleme der Form $\max \mathbf{c}^{\mathrm{T}}\mathbf{x}$ u.d.N. $\mathbf{Ax} = \mathbf{b}$ und $\mathbf{l} \leq \mathbf{x} \leq \mathbf{u}$ betrachtet. Anstelle der Nichtnegativitätsbedingungen treten hier untere und obere Schranken.

In der kompakten Matrix-Vektornotierung erscheint dieses Problem in der Form

$$\begin{array}{ll} \max & \mathbf{c}^{\mathrm{T}}\mathbf{x} \\ \textit{u.d.N.} & \mathsf{A}\mathbf{x} = \mathbf{b} \qquad , \quad \mathbf{x} \geq 0 \end{array} \quad .$$

Wenn die Bedeutung klar ist und keine Missverständnisse zu befürchten sind, schreiben wir statt der ausführlicheren Variante $\max_{\mathbf{x}} z$, $z = z(\mathbf{x})$ häufig auch kurz $\max \mathbf{c}^{\mathrm{T}}\mathbf{x}$, d. h. wir kombinieren Optimierungssinn und Zielfunktion. Hierbei ist zu beachten, dass $\mathbf{x}$ und $\mathbf{b}$ Spaltenvektoren, $\mathbf{c}^{\mathrm{T}}$ dagegen ein Zeilenvektor ist; das hochgestellte T bedeutet *Transponieren* und wandelt einen Spaltenvektor in einen Zeilenvektor um – siehe dazu bei Bedarf auch Anhang A.1. Das Produkt aus Zeilen- und Spaltenvektor ist das Skalarprodukt

$$\mathbf{c}^{\mathrm{T}}\mathbf{x} := \sum_{j=1}^{n} c_j x_j \quad .$$

Das Matrix-Vektorprodukt $\mathsf{A}\mathbf{x}$ genügt den üblichen Regeln der linearen Algebra; für die i-te Nebenbedingung erhalten wir z. B.

$$\mathbf{A}_i\mathbf{x} := \sum_{j=1}^{n} A_{ij} x_j \quad ,$$

wobei $\mathbf{A}_i$ den aus der i-ten Zeile der Matrix A gebildeten Zeilenvektor darstellt.

Das „Bootsverleihproblem" [(4.2.1) und (4.2.2)] führt mit der Identifikation

$$\mathbf{x} = \begin{pmatrix} p \\ s \end{pmatrix}$$

auf die Zielfunktion

$$\max_{(p,s)} z \quad , \quad z = z(p,s) = (800, 600) \begin{pmatrix} p \\ s \end{pmatrix} \quad .$$

Die linke Seite von (4.2.2) – unter Berücksichtigung von $p - s \geq 0 \Leftrightarrow -p + s \leq 0$ – schreibt sich nun

$$\begin{pmatrix} 1 & 1 \\ 1 & 0 \\ 1 & -1 \\ 4 & 3 \end{pmatrix} \begin{pmatrix} p \\ s \end{pmatrix} \quad ,$$

während die rechte Seite lediglich durch den Vektor

$$\begin{pmatrix} 350 \\ 200 \\ 0 \\ 1400 \end{pmatrix}$$

dargestellt wird. Die Verknüpfung zwischen diesen Termen liegt noch in der Form von Gleichungen und Ungleichungen vor und muss noch in die Standardnotation überführt werden. Minimierungsprobleme können mit Hilfe der Beziehung

$$\min f(\mathbf{x}) = -\max(-f(\mathbf{x}))$$

in Maximierungsprobleme transformiert werden. Im Folgenden wird gezeigt, wie jedes Problem, das Ungleichungen enthält, in die Standardnotation überführt werden kann.

4.2.2 Schlupf- und Überschussvariablen

Ungleichungen lassen sich mit Hilfe zusätzlicher Variablen in Gleichungen überführen. So kann z. B. die Ungleichung (2.1.1)

$$p + s \leq 350 \tag{4.2.3}$$

als Gleichung

$$p + s + t = 350$$

reformuliert werden, wobei t als *Schlupfvariable* [engl.: *slack variable*] bezeichnet wird. Zu beachten ist, dass aus (4.2.3) die Ungleichung

$$t = 350 - p - s \geq 0$$

folgt, d. h. die Schlupfvariable ist stets als nichtnegative Variable gewählt. In manchen Fällen kommt einer Schlupfvariable keine reale Bedeutung zu; sie stellt lediglich die Differenz zwischen dem aktuellen Wert einer Nebenbedingung und ihrer rechten Seite dar. Häufiger dagegen, wie auch in unserem Beispiel, repräsentiert sie unverbrauchte Ressourcen, ungenutze Kapazitäten oder unbefriedigte Nachfragen.

Im Falle einer "$\geq$"-Ungleichung spricht man von *Überschussvariablen*; so wird z. B. aus der Ungleichung

$$3x + 2y \geq 500$$

die Gleichung

$$3x + 2y - u = 500$$

mit der nichtnegativen Überschussvariable u. Führt man schließlich für jede Ungleichung eine Schlupf- oder Überschussvariable ein, so erhält man die Standardform.

4.2.3 Unterbestimmte lineare Gleichungssysteme und Optimierung

Nachfolgend seien die linearen Gleichungssysteme betrachtet, die sich bei linearen Optimierungsproblemen in Standardform ergeben; sie sind meist *unterbestimmt*, d. h. haben mehr Variablen (Freiheitsgrade) als Gleichungen oder enthalten redundante[4] Gleichungen. Diese Unterbestimmtheit ist wesentlich für die Optimierung, denn ohne diese Freiheitsgrade könnte man gar nicht optimieren. Das lineare Gleichungssystem (4.2.4), welches das „Bootsverleihproblem" in Standardnotation darstellt, verwendet sowohl Schlupf- als auch Überschussvariablen und enthält z. B. mehr Variablen als Gleichungen. Das lineare System

$$\begin{aligned} 2x + 3y &= 1 \\ 4x + 6y &= 2 \\ 6x + 9y &= 3 \end{aligned}$$

[4] Wir nennen eine Gleichung $f_i(\mathbf{x}) = 0$ eines Gleichungssystem $\mathbf{f}(\mathbf{x}) = \mathbf{0}$ *redundant*, wenn sie keine weitere Information hinsichtlich der Lösung des System beiträgt, d. h. die Lösung ändert sich nicht, wenn man $f_i(\mathbf{x}) = 0$ einfach weglässt. Die Lösung von $\mathbf{f}(\mathbf{x}) = \mathbf{0}$ ist bereits vollständig durch alle übrigen Gleichungen bestimmt. Zu beachten ist hierbei jedoch, dass Redundanz eine Eigenschaft ist, die dem Gleichungssystem $\mathbf{f}(\mathbf{x}) = \mathbf{0}$ simulan zukommt. Statt $f_i(\mathbf{x}) = 0$ als redundante Gleichung anzusehen, könnte auch eine andere Gleichung $f_j(\mathbf{x}) = 0$ als redundanter Kandidat zählen.

ist ein Beispiel mit 2 Variablen und 3 Gleichungen, von denen 2 Gleichungen jedoch redundant sind. Obwohl die Gleichungen unterschiedlich aussehen, tragen sie doch nur dieselbe Information

$$y = \frac{1}{3} - \frac{2}{3}x \quad .$$

Der Grund für diese Redundanz ist die lineare Abhängigkeit; wird die erste Gleichung mit 2 multipliziert, so erhält man die zweite, multipliziert man sie mit 3, so folgt die dritte. Das folgende Gleichungssystem

$$\begin{aligned} x + y &= 4 \\ 8x - y &= 5 \end{aligned}$$

mit 2 Unbekannten und 2 Variablen soll als einfaches Beispiel für ein nicht unterbestimmtes System dienen. Addieren wie beide Gleichungen, so erhalten wir

$$9x = 9 \quad ,$$

woraus eindeutig $x = 1$ und schließlich $y = 3$ folgt. In diesem Falle erlauben die beiden Gleichungen also die eindeutige Bestimmung der beiden Variablen.

In einem unterbestimmten Gleichungssystem ist die eindeutige Bestimmung einer Lösung nicht möglich. Daraus ergibt sich aber die Freiheit, aus all den Lösungen, die das Gleichungssystem erfüllen, eine Lösung auszuwählen, die z. B. eine bestimmte Funktion, die Zielfunktion, maximiert.

Daher kann Lineare Optimierung auch als das Problem angesehen werden, ein unterbestimmtes Gleichungssystem zu lösen und dabei eine Lösung dieses linearen Gleichungssystem zu wählen, die zu einem optimalen Wert der Zielfunktion führt. Die dabei abzuarbeitende Reihenfolge von Arbeitsschritten wird in einem Algorithmus zusammengefasst. Dieser Algorithmus wird zwar noch mit dem Konzept der Lösung linearer Gleichungen verbunden sein, wird sich aber dahingehend unterscheiden, dass in der Regel mehr Variablen als Nebenbedingungen vorliegen. Bezeichnen n und m die Anzahl von Variablen und Nebenbedingungen. Da mehr Variablen als Nebenbedingungen vorliegen, kann man $n - m$ von ihnen als *unabhängig* oder *ungebunden* ansehen; man könnte sie z. B. auf den Wert 0 fixieren, oder wie im Abschnitt A.1.3 beschrieben auf die untere oder obere Schranke [engl.: *bounds*] der betreffenden Variable. Die übrigen m Variablen werden als *abhängige* oder *gebundene* Variablen bezeichnet und ergeben sich als Lösung eines linearen Gleichungssystems mit m Variablen und m Nebenbedingungen.

4.2.4 Simplexverfahren und Bootsverleihproblem

Um zu demonstrieren, wie das Simplexverfahren funktioniert, wollen wir in diesem Abschnitt das in Abschnitt 2.1 vorgestellte Bootsverleihproblem mit Hilfe des Simplexverfahrens lösen. Da wir das Symbol **s** für den Vektor der Schlupfvariablen verwenden wollen, benennen wir die vormals mit p und s bezeichneten Variablen in x_1 und x_2 um und nähern uns damit auch der mathematischen Standardnotation. In der Standardformulierung führt das Problem auf die Darstellung

$$\max_{(x_1,x_2)} z \quad , \quad z = z(x_1, x_2) = 800x_1 + 600x_2$$

unter den Nebenbedingungen

$$\begin{array}{rcrcrcrcrcrcr} x_1 & + & x_2 & + & s_1 & & & & & & = & 350 \\ x_1 & & & & & + & s_2 & & & & = & 200 \\ -x_1 & + & x_2 & & & & & + & s_3 & & = & 0 \\ 4x_1 & + & 3x_2 & & & & & & & + & s_4 & = & 1400 \end{array} \quad . \tag{4.2.4}$$

Zusätzlich zu den nichtnegativen Variablen x_1 und x_2, die die Anzahl der Premier- und Standardboote bezeichnen, führen wir die Hilfsvariablen (Schlupf- und Überschussvariablen) $s_1, \ldots s_4$ ein. Die dritte Gleichung ist aus $x_1 - x_2 \geq 0$ abgeleitet und führt auf $x_1 - x_2 - s_3 = 0$. Um die Hilfsvariablen mit Koeffizienten +1 erscheinen zu lassen –wir sehen weiter unten, warum dies vorteilhaft ist– multiplizieren wir diese Gleichung mit -1.

Für einen Moment soll die Zielfunktion zunächst vernachlässigt werden. Damit reduziert sich das Problem auf die Lösung eines linearen Gleichungssystems. Wie kann nun das (unterbestimmte) Gleichungssystem (4.2.4) gelöst werden? Offenbar ist

$$x_1 = x_2 = 0, \quad s_1 = 350, \quad s_2 = 200, \quad s_3 = 0, \quad s_4 = 1400$$

eine Lösung. Aber wie erhalten wir diese Lösung? Im vorliegenden Fall ist dies einfach, da alle Schlupfvariablen mit Koeffizienten +1 in verschiedenen Zeilen auftreten und jede Zeile (Gleichung) nur genau eine Schlupfvariable hat. Variablen mit dieser Eigenschaft (sie treten nur in einer Gleichung mit Koeffizient +1 auf) wollen wir *kanonische* Variablen nennen. Kanonische Variablen stellen einen Spezialfall der Basisvariablen[5] dar und erlauben uns, die folgende Heuristik anzuwenden: Allen übrigen Variablen wird der Wert Null zugeordnet; sie werden Nichtbasisvariablen genannt. Die rechte Seite der Gleichungen ergibt dann die Werte der Basisvariablen. Die Lösung wird *Basislösung* genannt und eine *zulässige Basislösung* ist eine solche, in der alle Basisvariablen nichtnegative Werte haben. Allerdings wird der Wert $z = 0$ der Zielfunktion, der sich bei der Startlösung $\mathbf{x}^0 = (0, 0, 350, 200, 0, 1400; 0)$ ergibt, unseren Bootsverleiher nicht sehr erfreuen.

Die Lösung $\mathbf{x}^0$ besagt, dass überhaupt keine Boote gepachtet werden sollen. Nun könnte man sich fragen, was geschieht, wenn man eine Nichtbasisvariable von 0 auf 1 erhöht. Dabei sollte jedoch nicht die Zulässigkeit der Lösung verloren gehen. Hierzu sei für jede Nichtbasisvariable x_j die Größe

$$d_j := Z^{neu} - Z^{alt}$$

eingeführt, die misst, welchen Effekt eine normierte Erhöhung einer Nichtbasisvariable auf die Zielfunktion hat, wobei das lineare Gleichungssystem berücksichtig werden soll. Hierbei bezeichnen Z^{neu} und Z^{alt} die Werte nach und vor der Erhöhung der Nichtbasisvariablen. Als Beispiel sei die mit x_1 assoziierte Größe d_1 betrachtet:

$$Z^{neu} = 800 + 0 \cdot (350 - 1) + 0 \cdot (200 - 1) + 0 \cdot (0 + 1) + 0 \cdot (1400 - 4) = 800$$

[5] Für Leser mit Hintergrund in Linearer Algebra: Betrachtet seien die Matrix A der Nebenbedingungen und eine Menge von m Spaltenvektoren $\mathbf{A}_j$ aus A. Sind die Vektoren $\mathbf{A}_j$ linear unabhängig, so werden die zu den Spalten gehörenden Variablen x_j Basisvariablen [engl.: *basic variables*] genannt. Da es meist verschiedene Mengen von m linear unabhängiger Spaltenvektoren gibt, gibt es entsprechend auch verschiedene Mengen von Basisvariablen. Es ist also nur sinnvoll von einer Variablen als Basisvariablen zu sprechen, wenn dies in einem entsprechenden Kontext geschieht. Hier unterscheiden sich die Basisvariablen von den kanonischen Variablen; bei letzteren kann sofort entschieden werden, ob es eine kanonische Variable ist oder nicht. Bei Betrachtung der Spalten einer Matrix mit m Zeilen ergibt sich nach eventueller Umsortierung als Basismatrix die Einheitsmatrix, ein sehr spezielles Beispiel einer Menge linear unabhängiger Spaltenvektoren.

mit $Z^{alt} = 0$ und daher $d_1 = 800$.

Zur Ableitung einer allgemeinen Formel mögen die folgenden Überlegungen dienen: Der erste Term ist offensichtlich der Zielfunktionskoeffizient c_j der interessierenden Nichtbasisvariable mit Index j. Die Koeffizienten vor den Klammern sind die Zielfunktionskoeffizienten der aktuellen Basisvariablen. Die Terme in den Klammern sind jeweils der Wert der aktuellen Basisvariablen abzüglich des Koeffizienten der Nichtbasisvariablen in der korrespondierenden Zeile (Gleichung). Mit etwas linearer Algebra und dem Skalarprodukt können wir die Größe d_j mit der folgenden Formel

$$d_j := \left[c_j + \mathbf{c}_B^{\mathrm{T}} \left(\mathbf{x}_B - \mathbf{A}_j\right)\right] - \mathbf{c}_B^{\mathrm{T}} \mathbf{x}_B \tag{4.2.5}$$

berechnen, wobei der Zeilenvektor $\mathbf{c}_B^{\mathrm{T}}$ die mit den Basisvariablen assoziierten Zielfunktionskoeffizienten enthält, $\mathbf{x}_B$ ist der Spaltenvektor mit den Werten der Basisvariablen, $\mathbf{A}_j$ ist der Spaltenvektor mit den Koeffizienten der j-ten Spalte der aktuellen Matrix[6] A. Die Formel (4.2.5) lässt sich vereinfachen und ergibt schließlich die *relativen Profite* [engl.: *relative profits*] d_j

$$d_j = c_j - \mathbf{c}_B^{\mathrm{T}} \mathbf{A}_j \quad . \tag{4.2.6}$$

Im obigem Beispiel haben wir

$$\mathbf{c}_B^{\mathrm{T}} = (0,0,0,0) \quad , \quad c_1 = 800 \quad , \quad \mathbf{A}_1 = \begin{pmatrix} 1 \\ 1 \\ -1 \\ 4 \end{pmatrix}$$

und daher wieder

$$d_1 = 800 - (0,0,0,0) \begin{pmatrix} 1 \\ 1 \\ -1 \\ 4 \end{pmatrix} = 800 \quad .$$

Der Begriff *relativer Profit* bringt zum Ausdruck, dass es sich um den spezifischen Profit handelt, der bei Einheitserhöhung einer Nichtbasisvariable und gleichzeitiger Verringerung einer Basisvariable auftritt [der Term $\mathbf{x}_B - \mathbf{A}_j$ berücksichtigt dies in (4.2.5)]. Im Falle eines Minimierungsproblems verwenden wir eher die Begriffe relative Kosten [engl.: *relative costs*] oder reduzierte Kosten [engl.: *reduced costs*]. Hinsichtlich einer ökonomischen Interpretation der reduzierten Kosten sei auf Abschnitt 3.3.5.2 verwiesen.

Wenden wir (4.2.6) auf die Nichtbasisvariable x_2 an, so finden wir $d_2 = 600$. Was können wir dagegen erwarten, wenn wir (4.2.6) auf eine Basisvariable x_j anwenden? In diesem Falle folgt $d_j = 0$, da die zu x_j korrespondierende Spalte $\mathbf{A}_j$ gerade ein Spaltenvektor ist, der nur in einem Eintrag den Wert 1 hat, während alle anderen Komponenten den Wert 0 haben. Daher ergibt das Skalarprodukt $\mathbf{c}_B^{\mathrm{T}} \mathbf{A}_j$ den zu x_j korrespondierenden Koeffizienten der Zielfunktion, also $\mathbf{c}_B^{\mathrm{T}} \mathbf{A}_j = c_j$ und daher $d_j = 0$.

Wozu sind nun die d_j nützlich? Wir können sie dazu verwenden, um zu entscheiden, welche Nichtbasisvariable wir in die Basis aufnehmen. In unserem Beispiel ist $d_1 > d_2$ und daher betrachten wir x_1 als mögliche Variable, die in die Basis aufgenommen werden kann. Dieses Verfahren der Basisauswahl wird in der angelsächsischen Literatur *Pricing* bzw. *Pricing-out* genannt, wobei die d_j als Preise angesehen werden. Die Wahl, die Nichtbasisvariable mit größtem d_j in die Basis zu nehmen, ist eine mögliche Heuristik; es existieren

[6] Im ursprünglichen Simplexverfahren wurde die Matrix A in jeder Iteration verändert.

andere Verfahren. Im Prinzip ist jede Variable mit $d_j > 0$ ein geeigneter Kandidat. Unter Gesichtspunkten der numerischen Effizienz wäre es natürlich wünschenswert, diejenige Variable zu wählen, die zu der kleinsten Anzahl von Iterationen führen würde; nur ist leider kein Verfahren bekannt, das dies leisten würde.

Aus der Definition der Basisvariablen bzw. speziell der kanonischen Variablen folgt, dass höchstens m Basisvariablen existieren können, wobei m die Anzahl linearer Gleichungen bezeichnet. Deshalb muss bei Aufnahme einer neuen Basisvariable eine bereits existierende Basisvariable eliminiert werden; fragt sich nun, welche. Um dies zu entscheiden, sei wieder (4.2.4) betrachtet bzw. speziell die Basisvariablen $s_1, \ldots, s_4$ und die neue Basisvariable x_1; weiterhin gilt $x_2 = 0$. Damit reduziert sich (4.2.4) zu

$$\begin{array}{rcrcrcrcrcr} x_1 & + & s_1 & & & & & & & = & 350 \\ x_1 & & & + & s_2 & & & & & = & 200 \\ -x_1 & & & & & + & s_3 & & & = & 0 \\ 4x_1 & & & & & & & + & s_4 & = & 1400 \end{array} \quad .$$

Die Frage, welche existierende Basisvariable eliminiert werden kann, folgt aus der Überlegung, wie weit x_1 erhöht werden kann, bevor die erste der existierenden Basisvariablen ihre untere Schranke, in diesem Fall Null, erreicht. Die Bedingungen $s_1, \ldots, s_4 \geq 0$ führen zu den Ungleichungen

$$\left\{ \begin{array}{rcrcl} 350 & - & x_1 & \geq & 0 \\ 200 & - & x_1 & \geq & 0 \\ 0 & + & x_1 & \geq & 0 \\ 1400 & - & 4x_1 & \geq & 0 \end{array} \right\} \Leftrightarrow \left\{ \begin{array}{rcl} x_1 & \leq & 350 \\ x_1 & \leq & 200 \\ x_1 & \geq & 0 \\ x_1 & \leq & 325 \end{array} \right\} \quad .$$

Die zweite Ungleichung wird zuerst *aktiv*[7], nämlich dann, wenn x_1 den Wert 200 annimmt. Die dritte Ungleichung ist stets erfüllt und wird nie aktiv. Somit ist s_2 als zu eliminierende Variable identifiziert. Die sogenannte *Regel des minimalen Verhältnisses* [engl.: *minimum ratio rule*] verallgemeinert die Beobachtung des Beispiels: Zu eliminieren ist die Basisvariable x_e, für die der minimale Wert

$$\frac{b_e}{A_{ej}} = \min_i \left\{ \left. \frac{b_i}{A_{ij}} \right| A_{ij} > 0 \right\} \tag{4.2.7}$$

angenommen wird; hierbei ist j der Index, der die neue Basisvariable indiziert und das Symbol | führt eine logische Bedingung ein, hier die Positivitätsbedingung an A_{ij}. Gilt für alle zu x_j gehörenden Spaltenwerte $A_{ij} \leq 0$, so ist das LP-Problem unbeschränkt [engl.: *unbounded*].

Während nun der Wert der neuen Basisvariablen x_1 bekannt ist, bleiben noch die Werte der übrigen Basisvariablen zu bestimmen. Insbesondere sollte die Tatsache, dass x_1 eine Basisvariable ist (im vorliegenden Fall sogar eine kanonische Variable) im linearen Gleichungssystem offenbar werden, d. h. x_1 sollte mit Koeffizient +1 in nur einer Gleichung auftreten.

Wir können diesen Zustand elementar oder mit etwas linearer Algebra erreichen. Zwei lineare Gleichungssysteme $\mathsf{A}_1\mathbf{x} = \mathbf{b}_1$ und $\mathsf{A}_2\mathbf{x} = \mathbf{b}_2$ heißen *äquivalent*, wenn sie die gleiche Lösungsmenge haben. Dies sei anhand der Systeme

[7] Eine Ungleichung heisst *aktiv* für bestimmte Werte der Variablen, wenn die Ungleichung als Gleichung erfüllt wird. So wird z. B. $x + y \leq 5$ für $x = 3$ und $y = 2$ aktiv, da sich dann $5 \leq 5$ ergibt.

$$\begin{array}{rcrcrcrcrcl} x_1 & - & 2x_2 & + & x_3 & - & 4x_4 & + & 2x_5 & = & 2 \\ x_1 & - & x_2 & + & x_3 & - & 3x_4 & - & x_5 & = & 4 \end{array} \tag{4.2.8}$$

und

$$\begin{array}{rcrcrcrcrcl} x_1 & & & + & x_3 & - & 2x_4 & - & 4x_5 & = & 6 \\ & & x_2 & & & + & x_4 & - & 3x_5 & = & 2 \end{array} \tag{4.2.9}$$

erläutert. Anstatt zu zeigen, dass beide die gleiche Lösungsmenge haben, genügt es zu zeigen, dass beide Systeme mit Hilfe *elementarer Zeilenoperationen* ineinander überführt werden können. Darunter versteht man die Multiplikation von Zeilen mit einem von Null verschiedenen Faktor, sowie die Addition bzw. Subtraktion von Zeilen; diese Operation verändert die Lösungsmenge des Gleichungssystems nicht. Im Beispiel folgt (4.2.9) aus (4.2.8) mit Hilfe der folgenden Operationen: Subtrahiere die erste Zeile von der zweiten, dann addiere das doppelte der zweiten Zeile zu der ersten hinzu.

Nun sei dieses Verfahren auf (4.2.4) angewendet, wobei berücksichtigt werden soll, dass x_1 die existierende Basisvariable s_2 ersetzt; bezeichne der Index e die Gleichung (Zeile), aus der eine existierende Basisvariable eliminiert wurde. Diese Zeile transformiert sich dann gemäß der folgenden Formel:

$$A_{ec} \to A'_{ec} = \frac{A_{ec}}{A_{ej}} \quad ; \quad c = 1, \ldots, n+1 \quad , \tag{4.2.10}$$

wobei A_{ej} der Koeffizient der neuen Basisvariablen in dieser Gleichung (Zeile) ist und die $(n+1)$-te Spalte der Matrix A der rechten Seite der Gleichung entspricht, d. h.

$$A_{r,n+1} = b \quad , \quad A'_{r,n+1} = b' \quad .$$

Die Zeilen (Gleichungen) mit $r \neq e$ transformieren sich etwas komplizierter

$$A_{rc} \to A'_{rc} = A_{rc} - \frac{A_{rj}}{A_{ej}} A_{ec} \quad ; \quad \begin{array}{l} r = 1, \ldots, m \\ r \neq e \end{array} \quad ; \quad c = 1, \ldots, n+1 \quad . \tag{4.2.11}$$

Bei Anwendung der Formeln (4.2.10) und (4.2.11) auf (4.2.4) erhalten wir wieder ein Gleichungssystem in kanonischer Form, d. h. jede Gleichung enthält genau eine kanonische Variable:

$$\begin{array}{rcrcrcrcrcl} & x_2 & + & s_1 & - & s_2 & & & & = & 150 \\ x_1 & & & & + & s_2 & & & & = & 200 \\ & x_2 & & & + & s_2 & + & s_3 & & = & 200 \\ & 3x_2 & & & - & 4s_2 & & & + \quad s_4 & = & 600 \end{array} \tag{4.2.12}$$

und

$$\mathbf{x}_B = \begin{pmatrix} s_1 \\ x_1 \\ s_3 \\ s_4 \end{pmatrix} = \begin{pmatrix} 150 \\ 200 \\ 200 \\ 600 \end{pmatrix} \quad , \quad \mathbf{c}_B^{\mathrm{T}} = (0, 800, 0, 0) \quad .$$

Der Firmeninhaber der Bootsverleihfirma ist nun schon etwas glücklicher, da die Zielfunktion immerhin schon einen Wert von £160,000 hat. Bleibt die Frage: Ist das schon alles, also das Optimum? Wie kann diese Frage sicher beantwortet werden? Hier hilft ein erneutes *Pricing* mit dem Ergebnis

$$\mathbf{d} = (0, 600, 0, 0, 0, 0) \quad .$$

Offensichtlich ist x_2 ein weiterer Kandidat, der zu einer besseren Lösung führen kann. Diesmal zeigt (4.2.7) an, dass in der ersten Gleichung in (4.2.12) s_1 eliminiert werden muss. Erneute Transformation der linearen Gleichungen liefert

$$\begin{array}{rcrcrcrcrcr} & x_2 & + & s_1 & - & s_2 & & & & = & 150 \\ x_1 & & & & + & s_2 & & & & = & 200 \\ & & - & s_1 & + & 2s_2 & + & s_3 & & = & 50 \\ & & - & 3s_1 & - & s_2 & & & + \; s_4 & = & 150 \end{array}$$

und

$$\mathbf{x}_B = \begin{pmatrix} x_2 \\ x_1 \\ s_3 \\ s_4 \end{pmatrix} = \begin{pmatrix} 150 \\ 200 \\ 50 \\ 250 \end{pmatrix} \quad , \quad \mathbf{c}_B^{\mathrm{T}} = (600, 800, 0, 0) \tag{4.2.13}$$

mit einem Nettoerlös von £250,000. Das erfreut den Bootsverleiher sehr und seine Begeisterung für die mathematische Optimierung wächst enorm. Dennoch stellt sich die Frage nach der Optimalität erneut. Diesmal liefert das *Pricing*

$$\mathbf{d} = (0, 0, -600, -800, 0, 0) \quad . \tag{4.2.14}$$

Da $\mathbf{d}$ keine positiven Komponenten mehr enthält, können wir sicher schließen, dass die Lösung (4.2.13) nicht weiter verbessert werden kann und somit optimal ist. Dies ist ein bedeutsames Ergebnis und es sollte daher klar sein, wie diese Information aus den Werten von $\mathbf{d}$ abgeleitet werden kann. Der Vektor $\mathbf{d}$ ist mit möglichen Änderungen der Werte der Nichtbasisvariablen verknüpft; die Basisvariablen können wiederum nicht geändert werden, ohne dass sich die Nichtbasisvariablen ändern. Wegen der unteren Schranken, hier die Bedingung der Nichtnegativität, können die Nichtbasisvariablen nur wachsen. Die relativen Profite $\mathbf{d}$ in (4.2.14) können dagegen den Wert der Zielfunktion nur verringern. Damit ist bewiesen, dass $x_1 = 200$ und $x_2 = 150$ die optimale Lösung ist und wir haben gelernt, dass das *Pricing* für zwei Zwecke nützlich ist: *Nachweis der Optimalität* und *Auswahl einer neuen Basisvariablen*.

Damit liefert das Simplexverfahren die folgenden Informationen: Die Werte der Variablen, eine Auswertung der Nebenbedingungen und den mit ihnen verbundenen Schlupf- oder Überschussvariablen, den Wert der Zielfunktion, die reduzierten Kosten (Profite) und den Optimalitätsnachweis. Weiterhin werden noch die in den Abschnitten 3.3.5.1 und A.2 sowie im Anhang A.1.1 diskutierten *Schattenpreise* oder *Werte der dualen Variablen* ausgewiesen; in Anhang A.1.3 wird gezeigt, wie Schattenpreise und reduzierte Kosten miteinander in Beziehung stehen. Der schon auf Seite 3.3.5.1 eingeführte Schattenpreis, oder der duale Wert einer Nebenbedingung ist in einem Maximierungsproblem der Wertzuwachs der Zielfunktion, der mit einer Einheitserhöhung der rechten Seite der Nebenbedingung einhergeht. Für die Nebenbedingung (2.1.1) ergibt eine Einheitserhöhung der Kapazität einen Zuwachs der Zielfunktion um 600 Einheiten; hierbei ist zu beachten, dass die Einheiten dieses Zuwachses sich als Quotient der Einheiten Zielfunktion und Nebenbedingung ergibt. Die dualen Werte lassen sich mit Hilfe der Formel (A.1.5) ausrechnen und ergeben für das Bootsverleihproblem £600/Boot, £200/Boot, £0/Boot und £0/Stunde für die vier Nebenbedingungen. Der duale Wert der Wartungsbedingung ist Null und zeigt damit an, dass die Bedingung nicht *bindend* oder *aktiv* ist; es sind noch 100 Einheiten verfügbar, bis die Kapazitätsgrenze erreicht ist.

Weitere geometrische und algebraische Eigenschaften des Simplex- und des revidierten Simplexverfahrens sowie Fragen der Entartung sind im Anhang A.1.1 vertieft.

4.3 Lineare gemischt-ganzzahlige Optimierung

Eine spezielle Klasse diskreter Optimierungsaufgaben mit großer Praxisrelevanz (Kallrath & Wilson 1997,[167]) sind lineare gemischt-ganzzahlige Probleme der Form

$$\min_{\mathbf{x}\in S} \mathbf{c}^{\mathrm{T}}\mathbf{x} \quad , \quad S := \{\mathbf{x}|\mathsf{A}\mathbf{x} = \mathbf{b}, \mathbf{x} \geq \mathbf{0}\} \tag{4.3.15}$$

mit ganzzahligen Variablen $x_1, ..., x_r \in \mathbb{N}_0$ und kontinuierlichen Variablen $x_{r+1}, ..., x_n \in \mathbb{R}_0^+$; in manchen Fällen ist es zudem möglich, nichtlineare Probleme mit Hilfe von Binärvariablen umzuformulieren oder doch wenigstens approximativ in der Form (4.3.15) zu lösen. Bei der Darstellung des zulässigen Bereichs S in der Form $\mathsf{A}\mathbf{x} = \mathbf{b}$ wurde bereits ausgenutzt, dass sich jede Ungleichung mit Hilfe von nichtnegativen Schlupfvariablen als eine Gleichungsbedingung schreiben lässt. Ganzzahlige Variablen $x_1, ..., x_r$, die nur die beiden Werte 0 und 1 annehmen können, nennt man *binäre Variablen*. Enthält ein Problem n Binärvariablen, so sind 2^n Wertekombinationen dieser Variablen möglich, wenn man mögliche weitere Relationen zwischen diesen Variablen zunächst einmal unbetrachtet lässt. Im Prinzip könnte man sich auf *gemischt-binäre Probleme* beschränken, da ganzzahlige Variablen $x \in \{0, 1, ..., N\}$ durch binäre Variablen $\delta_i \in \{0, 1\}$ in Form von Summen von 2er-Potenzen dargestellt werden können; numerisch ist dies jedoch ineffizient, was aufgrund folgender Überlegung ersichtlich wird. Angenommen, während der LP-Relaxierung ergibt sich 5.8 als Wert für x; behandelt man x als ganzzahlige Variablen, so wird man weitere Unterprobleme mit den Schranken $x \leq 5$ und $x \geq 6$ erzeugen. Bei der entsprechenden binären Darstellung ergibt sich möglicherweise $5.8 = 1 \cdot 2^0 + 1 \cdot 2^1 + 0.7 \cdot 2^2$; das ist aber nicht zwingend so, denn es könnten auch noch nichtganzzahlige Anteile höherer Zweierpotenzen auftreten. Betrachten wir aber den einfachsten Fall, so folgt $\delta_2 \leq 0$ und $\delta_2 \geq 1$; dies entspricht aber für x einer Erzeugung der Schranken $x \leq 3$ und $x \geq 7$, ist also weniger scharf, d. h. hier wesentlich weiter entfernt von eigentlich sinnvollen, aber nicht zulässigen Wert 5.8.

Lösungsansätze für lineare gemischt-ganzzahlige Optimierungsprobleme der Form (4.3.15) sind zum einen Heuristiken bzw. simulationsbasierte, lokale und globale Suchverfahren [211], zum anderen exakte Methoden (Nemhauser & Wolsey 1988,[222]). Zur ersten Gruppe gehören z. B. *Simulated Annealing* (siehe z. B. [1], [77] oder [2]) und *Tabusuche* [engl.: *Tabu Search*] [109]; sie generieren bestenfalls zulässige Punkte ohne Optimalitäts- oder Qualitätsnachweis, taugen damit nur bedingt zur Lösung sicherer, quantitativ fundierter Entscheidungsprozesse und sollen hier nicht weiter betrachtet werden - der interessierte Leser sei aber auf das instruktive Buch von Michaelewicz & Fogel (2000,[211]) verwiesen.

Exakte Verfahren erlauben die Bestimmung einer optimalen Lösung; hierzu zählen Entscheidungsbaum- und Schnittebenenverfahren, die sich in vollständige und begrenzte oder implizite Enumerationsverfahren unterteilen. Zur Gruppe der letzteren gehören z. B. Branch&Bound (B&B) und Branch&Cut (B&C) Algorithmen,), *L-Klassen-Enumeration* in Abschnitt 4.3.5 und dynamische Optimierung unterteilen. *Vollständige Enumeration* ist ein intuitiv einleuchtendes Verfahren, das alle zulässigen Punkte durch Kombination aller zulässigen Werte der ganzzahligen Variablen generiert und selektiert daraus, wie der Bauer es in Abschnitt 2.2.1 tat, die optimale Lösung. Bei gemischt-ganzzahligen linearen Problemen muss hierbei noch nach Fixierung der ganzzahligen Variablen ein LP-Problem hinsichtlich der verbleibenden kontinuierlichen Variablen gelöst werden. Da

der Aufwand zur Berechnung aller Lösungen exponentiell[8] steigt, sind diese expliziten Verfahren nur für kleinste Probleme sinnvoll anwendbar und werden in der Praxis durch *implizite Enumerationsverfahren* ersetzt; hierin werden zwar auch alle Kombinationen berücksichtigt, aber nicht explizit. *Branch&Bound*-Verfahren gehören zu dieser Klasse von Problemen und schränken den Suchbaum durch geeignete Verzweigung [engl.: *to branch*] eines Problems in mehrere Unterprobleme und Berechnung von Schranken [engl.: *to bound*] durch Lösung der Unterprobleme ein. Bei den Unterproblemen handelt es sich um LP-Probleme.

Bei den *Schnittebenenverfahren* [engl.: *cutting plane methods*] werden bezüglich der konvexen Hülle zulässige Ungleichungen generiert, die Teile der LP-Relaxierung eliminieren und statisch oder dynamisch zu einem Modell hinzugefügt; ähnliches gilt beim *Branch&Cut*-Verfahren.

Auf dynamische Optimierung oder auch Dynamische Programmierung [engl.: *dynamic programming*], wie sie z. B. in Nemhauser & Wolsey (1988,[222]) oder Ravindran *et al.* (1987,[240]) beschrieben wird, soll hier nicht eingegangen werden, da es sich nicht um einen allgemeinen Lösungsansatz handelt. Sie wurde ursprünglich zur Optimierung sequentieller Entscheidungsprozesse entwickelt. Dieses Verfahren zur Lösung mehrstufiger Probleme lässt sich sowohl auf lineare als auch nichtlineare Optimierungsprobleme anwenden, die sich mit Hilfe von ineinander geschachtelter Familien von Unterproblemen lösen lassen; Beispiele hierzu finden sich in Nemhauser und Wolsey (1988,[222]).

Darüber hinaus existieren noch eine Reihe anderer Methoden zur Lösung von MILP-Problemen, die nicht primär auf der Lösung von LP-Problemen beruhen. Hierzu zählen Probleme ohne kontinuierliche Variablen oder solche, die nur Binärvariablen enthalten. Eines der ältesten Verfahren ist der auf Balas (1965,[22]) zurückgehende *Additive Algorithmus*. Andere Methoden wurden von Granot & Hammer (1972,[116]) und Wolsey (1971,[296]) entwickelt; diese Methoden sind zwar selten in kommerziellen Softwarepakten integriert, aber ihre grundlegenden Ideen haben doch gelegentlich Einzug gefunden in bestimmte Verzweigungsstrategien des B&B-Verfahrens.

4.3.1 Elementare Einführung des Branch&Bound-Verfahrens

Zur Illustrierung des Verfahrens soll das im Abschnitt 2.2.1 beschriebene, von Kühen und Schweinen handelnde Beispiel mit den Beziehungen (2.2.1) bis (2.2.2)

$$\max_{\kappa,\sigma} z \quad , \quad z = z(\kappa,\sigma) = 3\kappa + 2\sigma \tag{4.3.16}$$

unter den Nebenbedingungen

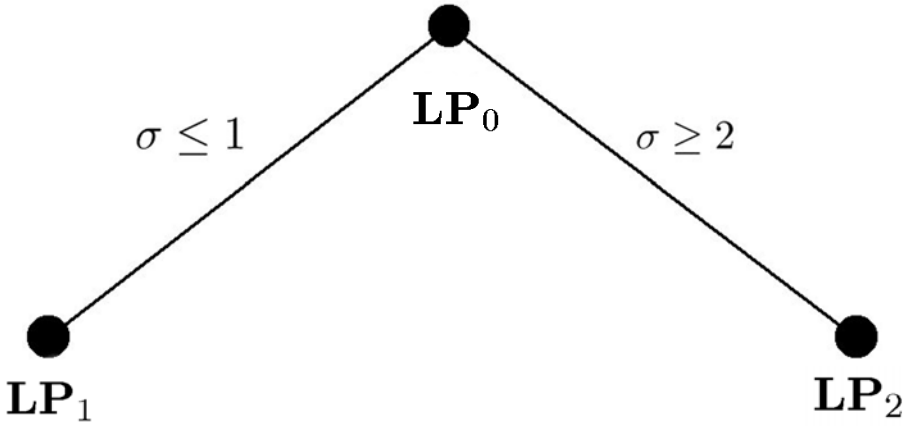
$$0 \le \kappa \le 2 \tag{4.3.17}$$

$$0 \le \sigma \le 2 \tag{4.3.18}$$

$$\kappa + \sigma \le 3.5 \tag{4.3.19}$$

[8] Enthält ein Problem n Binärvariablen, so sind 2^n Wertekombinationen dieser Variablen möglich, wenn man mögliche weitere Relationen zwischen diesen Variablen zunächst einmal unbetrachtet lässt. Während $n = 3$ auf $2^3 = 8$, $n = 10$ auf $2^{10} = 1024$ Kombinationen führt, ergeben sich bei $n = 50$ schon 2^{50} Kombinationen: eine Zahl mit insgesamt schon 16 Ziffern. Dabei gilt approximativ $2^n \approx 10^{0.3n}$, bzw. $2^n \approx 10^{M \cdot n}$ mit $M = \ln(2)/\ln(10) \approx 0.30103\ldots$, wobei M und Mn und den 10-Logarithmus der Zahlen 2 und Mn bezeichnen.

Abbildung 4.2 LP-Relaxierung LP_0 und die beiden ersten Unterprobleme LP_1 und LP_2 und deren Knoten im B&B-Baum.



$$\kappa, \sigma \in \{0, 1, 2\} \tag{4.3.20}$$

dienen; der zulässige Bereich ist in Abb. 4.3 dargestellt. Das allgemeine Verfahren, das derartige Probleme lösen kann, heißt *Branch und Bound* (abgekürzt B&B) mit LP-Relaxierung, geht auf Land & Doig (1960,[190]) zurück und ist eines der am häufigsten verwendeten Verfahren zur Lösung von ganzzahligen oder gemischt-ganzzahligen Problemen. Dabei gehen wir wie folgt vor: Zunächst lösen wir das Problem (4.3.16) bis (4.3.20) unter Vernachlässigung der Ganzzahligkeitsbedingung; das so erhaltene LP-Problem $\mathbf{LP}_0$, also (4.3.16) bis (4.3.19), wird als *LP-Relaxierung* bezeichnet. In den Abschnitten 2.1 und 4.2.4 wurde gezeigt, wie ein derartiges kontinuierliches lineares Optimierungsproblem gelöst werden kann. Als Lösung dieses relaxierten Problems ergibt sich

$$\mathbf{LP}_0: \qquad \kappa = 2 \quad , \quad \sigma = 1.5 \quad , \quad Z^0 = z = 9 \quad .$$

Offensichtlich, und vielleicht nicht ganz unerwartet, erfüllt σ noch nicht die Ganzzahligkeitsbedingung. Dennoch können wir bereits eine wichtige Information ableiten. Da das relaxierte Problem mehr Freiheiten als das ursprüngliche Maximierungsproblem besitzt, liefert Z^0 eine obere Schranke für den Zielfunktionswert, den wir erwarten können. Wir haben formal bewiesen, dass Wert der Zielfunktion nicht größer als 9 kEuro sein kann. In der formalen Beschreibung des B&B-Verfahrens wird die obere Schranke weiter verfolgt, mit z^{LP} bezeichnet und durch $z^{LP} = Z^0 = 9$ initialisiert. Zu betrachten ist ferner die untere Schranke z^{IP}, die mit $z^{IP} = -\infty$ initialisiert wird. Diese Schranken begrenzen die möglichen Werte des optimalen Zielfunktionswerts z^*

$$z^{LP} \geq z^* \geq z^{IP} \quad . \tag{4.3.21}$$

Wie können wir nun zu einer ganzzahlig zulässigen Lösung kommen? Hierzu werden, wie in Abbildung 4.2 illustriert, zwei neue Unterprobleme

$$\begin{array}{lllll} \mathbf{LP}_1: & \mathbf{LP}_0 & \oplus & \sigma \leq 1 \\ \mathbf{LP}_2: & \mathbf{LP}_0 & \oplus & \sigma \geq 2 \end{array}$$

erzeugt, in dem die LP-Relaxierung $\mathbf{LP}_0$, also das aus den Nebenbedingungen (4.3.16) bis (4.3.19) bestehende LP, durch eine Schrankenbedingung, hier $\sigma \leq 1$ bzw. $\sigma \geq 2$, ergänzt wird; dies wird durch das Symbol $\oplus$ angedeutet. Die Vereinigung der zulässigen Bereiche der Unterprobleme $\mathbf{LP}_1$ und $\mathbf{LP}_2$ enthält vollständig den zulässigen Bereich des ursprünglichen Problems; lediglich die Menge $1 < \sigma < 2$, also das offene Intervall $(1, 2)$ wurde ausgespart; $\kappa = 1$ und $\sigma = 2$ sind noch enthalten. Die Unterprobleme werden auch Zweige [engl.: *branches*] genannt; es bleibt zu entscheiden, auf welches Unterproblem zuerst verzweigt wird. Entschließt man sich zunächst, $\mathbf{LP}_1$ zu lösen, so findet man

$$\mathbf{LP}_1: \qquad \kappa = 2 \quad , \quad \sigma = 1 \quad , \quad Z^1 = z = 8 \quad .$$

Damit ist bereits eine zulässige Lösung des Problems gefunden; aus der Tabelle in Abschnitt 2.2.1 ist bekannt, dass dies auch schon die optimale Lösung ist. Wie kann man aber ohne Tabelle sicher sein, dass dies auch die optimale Lösung ist? Nun, da wir eine ganzzahlige Lösung mit $z = 8$ ermittelt haben, aktualisieren wir $z^{IP} = 8$; zudem wissen wir, dass keine Lösung mit einem Zielfunktionswert größer als $z^{LP} = 9$ existieren kann. Um die Optimalität von $z = 8$ zu beweisen, wird das andere Unterproblem (später auch *Knoten* genannt) untersucht; es besitzt die Lösung

$$\mathbf{LP}_2: \qquad \kappa = 1.5 \quad , \quad \sigma = 2 \quad , \quad Z^2 = z = 8.5 \quad ,$$

die leider aber noch nicht ganzzahlig zulässig ist. Immerhin wissen wir nun, dass wir keine Lösung erwarten können, die besser als $Z = 8.5$ ist. Setzt man die Prozedur fort, so werden die Unterprobleme

$$\mathbf{LP}_3: \quad \mathbf{LP}_2 \quad \oplus \quad \kappa \leq 1$$

und

$$\mathbf{LP}_4: \quad \mathbf{LP}_2 \quad \oplus \quad \kappa \geq 2$$

erzeugt, von denen das erste die ganzzahlig-zulässige Lösung

$$\mathbf{LP}_3: \qquad \kappa = 1 \quad , \quad \sigma = 2 \quad , \quad Z^3 = z = 7 \quad ,$$

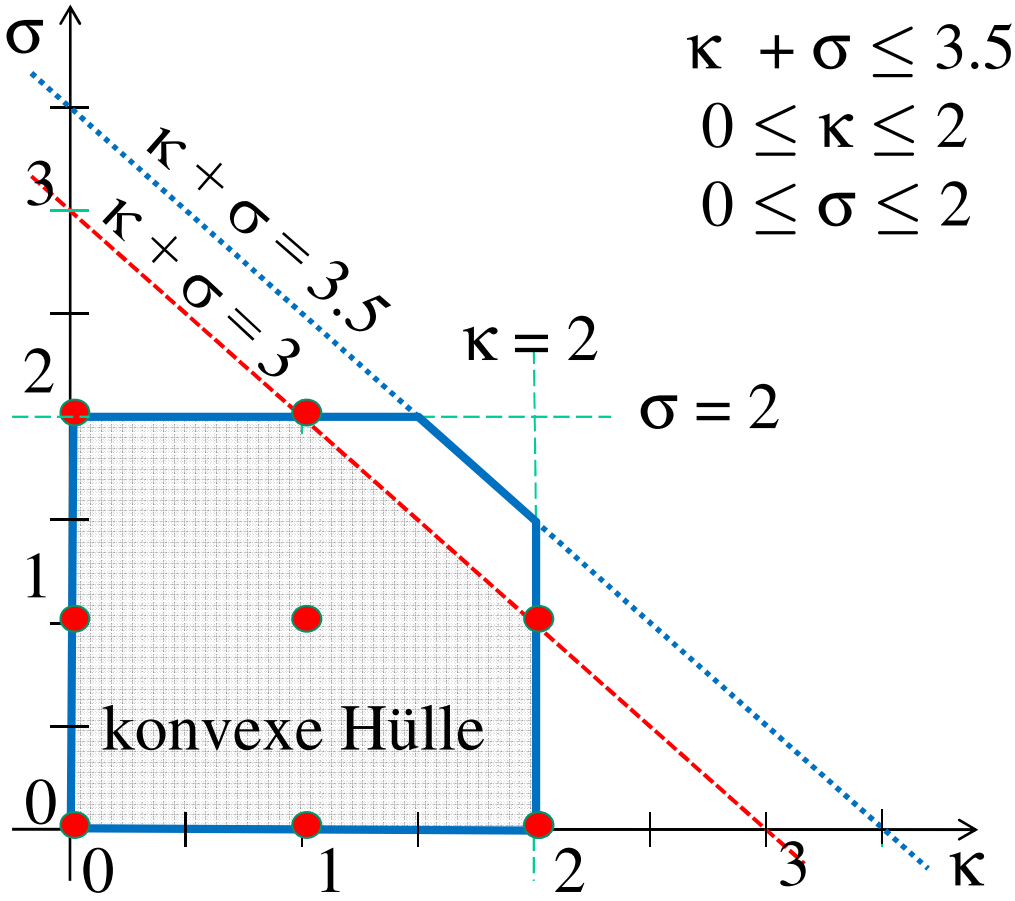
aber einen Zielfunktionswert $z = 7$ hat, der kleiner als unsere aktuell beste Lösung $z = 8$ ist, und deshalb nicht weiter berücksichtigt werden braucht. Das andere Unterproblem, $\mathbf{LP}_4$, besitzt keine Lösung, da es sich von $\mathbf{LP}_2$ ableitet; hinzugefügt wurde die neue Schranke $\kappa \geq 2$, die aber nicht simultan mit $\sigma \geq 2$ erfüllt werden kann. Damit ist die Betrachtung abgeschlossen und die Optimalität des Punktes $(\kappa, \sigma) = (2, 1)$ nach 5 Knoten, d. h. durch Lösung von 5 LP-Problemen, nachgewiesen.

In diesem Beispiel hätte die optimale Lösung bereits durch Lösung eines einzigen linearen Programms bestimmt werden können. Die Bedingung (4.3.19) kann nämlich infolge der Ganzzahligkeit von κ und σ, ohne den zulässigen Lösungsbereich des ganzzahligen Problems einzuschränken, zu

$$\kappa + \sigma \leq 3 \tag{4.3.22}$$

verschärft werden; da die Summe ursprünglich nicht größer als 3.5 sein darf, kann sie auch nicht größer als 3 sein. Die Lösung des so erhaltenen LPs liefert sofort $(\kappa = 2, \sigma = 1)$ und $z = 8$; da in diesem Fall die LP-Relaxierung des ganzzahligen Problems bereits ganzzahlig zulässig ist, wird die optimale Lösung durch Lösung eines einzigen linearen Programms

Abbildung 4.3 LP-Relaxierung (äußeres Polyeder mit blauer Umrandung) und konvexe Hülle (schraffiert). Verwendet man statt der ursprünglichen Ungleichung $\kappa + \sigma \leq 3.5$ die schärfere Ungleichung $\kappa + \sigma \leq 3$, so erhält man die konvexe Hülle. Wie in der Abbildung schön zu sehen ist, ist sie das kleinste Polyeder, das alle ganzzahlig zulässigen Punkte enthält.



bestimmt – der Grund hierfür ist, dass die Bedingungen $0 \leq \kappa \leq 2$, $0 \leq \sigma \leq 2$ und (4.3.22), wie in Abb. 4.3 gezeigt, genau die *konvexe Hülle*[9] des Optimierungsproblems (4.3.16) bis (4.3.19) beschreiben. Dieser günstige Fall, in dem bereits die LP-Relaxierung die konvexe Hülle und damit die optimale, ganzzahlig zulässige Lösung ergibt, hat eine verschwindene Ganzzahligkeitslücke [engl.: *integrality gap*]; dieses Konzept wird weiter im Abschnitt 5.8 vertieft. Der vorliegende Fall gibt aber auch bereits einen Hinweis auf den wesentlichen Punkt, der bei der Modellierung und Lösung ganzzahliger oder gemischt-

[9] Die *konvexe Hülle* ist der kleinste zulässige Bereich bzw. das kleinste Polyeder, der bzw. das mit Hilfe linearer Bedingungen beschrieben werden kann und gerade noch alle gemischt-ganzzahligen zulässigen Punkte enthält.

ganzzahliger Probleme zu beachten ist: Bei einem guten Modell liegt die LP-Relaxierung nahe an der konvexen Hülle des ganzzahligen oder gemischt-ganzzahligen Lösungsbereiches; in vielen Fällen kann dies während der Modellformulierung unter Hinzufügung weiterer *zulässiger* Ungleichungen erreicht werden.

Schließlich ist noch eine Bemerkung nützlich. Nach Inspektion der Lösungen der Probleme $\mathbf{LP}_1$ und $\mathbf{LP}_2$ kann schon aus

$$z^{LP} = \max\{z(\mathbf{LP}_1), z(\mathbf{LP}_2)\} = 8.5$$

und der bereits verfügbaren Lösung $z^{IP} = 8$ und damit der Ungleichungskette (4.3.21)

$$8 \leq z^* \leq 8.5 \tag{4.3.23}$$

die Optimalität $z^{IP} = 8$ bewiesen werden. Da nämlich die Variablen κ und σ nur ganzzahlige Werte annehmen können und die Koeffizienten der Zielfunktion allesamt ganzzahlig sind, kann auch die Zielfunktion nur ganzzahlig sein. Zwischen 8 und 8.5 gibt es aber keine ganzen Zahlen; also ist die zu $z^{IP} = 8$ gehörende, im Knoten $\mathbf{LP}_1$ gefundene Lösung $(\kappa = 2, \sigma = 1)$ optimal. Die Auswertung der Unterprobleme $\mathbf{LP}_2$ und $\mathbf{LP}_3$ ist also überflüssig; das Beispiel zeigt uns also, wie mit Hilfe von Schranken die B&B-Suche erheblich reduziert werden kann.

4.3.2 Branch&Bound (B&B) mit LP-Relaxierung

Die im Abschnitt 4.3.1 skizzierten Ideen sollen nun aufgegriffen werden und zu einer formaleren Beschreibung des Branch&Bound (B&B)-Verfahrens führen. Zunächst wird eine hinsichtlich der Ganzzahligkeitsbedingung relaxierte Variante von (4.3.15) gelöst, die dadurch entsteht, dass die Bedingung $x_1, ..., x_r \in \mathbb{N}_0$ durch $x_1, ..., x_r \in \mathbb{R}_0^+$ ersetzt wird; das so entstehende kontinuierliche Problem resultiert aus der *LP-Relaxierung* des ursprünglichen Problems. Intuitiv könnte man nun vorschlagen, nichtganzzahlige Werte zu runden. In der Regel, insbesondere im Extremfall binärer Variablen $x \in \{0, 1\}$, ist dieser Weg nicht gangbar. Das nachfolgende Beispiel zeigt, dass sich gemischt-ganzzahlige lineare Probleme und deren LP-Relaxierung drastisch unterscheiden können und daher auch mit einfachen Rundungsmechanismen aus relaxierten Programmen keine optimalen Lösungen für das ganzzahlige Problem

$$\max_{x_1, x_2} \quad x_1 + x_2 \quad ,$$

$$\begin{array}{r} -2x_1 + 2x_2 \geq 1 \\ -8x_1 + 10x_2 \leq 13 \end{array} \quad , \; x_1 \in \mathbb{N}_0 \; , \; x_2 \in \mathbb{N}_0$$

gefunden werden können. Die Geraden

$$g_1 : x_2 = x_1 + \frac{1}{2} \; ; \; g_2 : x_2 = 0.8x_1 + 1.3$$

begrenzen zusammen mit der x_1- und x_2-Achse den zulässigen Bereich, in diesem Fall ein sehr langgezogenes schmales Dreieck, in dem sich die Lösungen $Z^{LP} = 8.5(4, 4.5)$ und $Z^{IP} = 3(1, 2)$ graphisch bestimmen lassen. Z^{LP} und Z^{IP} unterscheiden sich sehr deutlich sowohl hinsichtlich des Lösungsvektors als auch des Zielfunktionswertes. Einfache Rundungsverfahren scheiden damit offensichtlich zur Lösung diskreter Probleme aus.

Beim B&B-Verfahren wird die LP-Relaxierung jedoch in folgender Weise ausgenutzt: Ist bei der Lösung $\mathbf{x}^k$, $k = 0, 1, 2, ...$, eines LP-Unterproblems P^k der Wert $\bar{x}_j^k$ einer ganzzahligen Variablen x_j mit $1 \leq j \leq r$ noch nicht ganzzahlig, so werden zwei disjunkte Unterprobleme P^{k+1} bzw. P^{k+2} erzeugt, indem zum Problem P^k die Ungleichungen $x_j \leq \lfloor \bar{x}_j^k \rfloor$ bzw. $x_j \geq \lfloor \bar{x}_j^k \rfloor + 1$ hinzugefügt werden, wobei $\lfloor \bar{x}_j^k \rfloor$ die größte ganze Zahl[10] ist, die nicht größer als $\bar{x}_j^k$ ist. Bei der Lösung der Unterprobleme P^{k+1} und P^{k+2} kann auf die bekannte Lösung P^k zurückgegriffen werden, indem die zusätzlichen Ungleichungen hinzugefügt werden und dann das *duale Simplexverfahren* eingesetzt wird.

Die Untersuchung eines so erzeugten Unterproblems bzw. Knotens P^k führt auf drei attraktive Fälle, die den Suchbaum nicht erweitern:

1. Für die Lösung $\mathbf{x}^k$ von P^k gilt: $\mathbf{x}^k \in S$; Vergleich mit einem anderen eventuell schon existierenden zulässigen Punkt von (4.3.15) und evtl. Ersetzung der bisherigen besten Lösung;
2. Das Problem P^k ist unzulässig;
3. Für die Lösung $\mathbf{x}^k$ von P^k gilt: $\mathbf{x}^k \notin S$ und $\mathbf{c}^{\mathrm{T}}\mathbf{x}^k$ ist größer als der Wert der Zielfunktion eines eventuell schon existierenden zulässigen Punktes von (4.3.15).

Andernfalls werden durch Hinzufügung weiterer Ungleichungen wieder zwei Unterprobleme generiert. Sowohl mit der *Variablenwahl* als auch mit der *Knotenwahl* lässt sich der Algorithmus steuern und die Effizienz der Methode steigern. Die LP-Relaxierung und die im Suchbaum gefundenen Lösungen von (4.3.15) erzeugen untere und obere Schranken Z^u und Z^o. Alle aktiven Knoten mit einer Bewertung $z \geq Z^o$ brauchen nicht weiter untersucht zu werden. Sind alle aktiven Knoten abgearbeitet, so ist entweder eine optimale Lösung von (4.3.15) bestimmt oder nachgewiesen, dass (4.3.15) keine Lösung besitzt.

Das auf Land & Doig (1960,[190]) zurückgehende B&B-Verfahren ist das in kommerziellen Paketen am häufigsten vorzufindende, hinsichtlich der kommerziellen Implementierungen von Dakin (1965,[56]) beschriebene exakte Verfahren zur Lösung gemischtganzzahliger Programme. Im Wesentlichen handelt es sich dabei um auf Verwerfungskriterien aufbauendes, effizientes implizites Enumerationsverfahren, das vermeidet, „ungünstige" Bereiche des Verzweigungsbaumes zu untersuchen. Die wesentlichen Konzepte dieses Verfahrens sind *Relaxierung*, *Sondierung* und *Separierung*; bei den nachfolgenden Betrachtungen ist vorausgesetzt, dass es sich um ein Maximierungsproblem handelt.

Relaxierung bedeutet im allgemeinen, einige Bedingungen des Problems zu lockern oder zu vernachlässigen. Angenommen, das ursprüngliche Problem besitzt die Menge der zulässigen Punkte S. Das relaxierte Problem besitzt dann ebenfalls zulässige Punkte, die meist eine umfassendere Menge S_{R} bilden, d. h. $S \subseteq S_{\mathrm{R}}$. Eine Konsequenz für die Zielfunktionswerte eines Optimierungsproblems ist

$$\max\{f(\mathbf{x}) | \mathbf{x} \in S_{\mathrm{R}}\} \geq \max\{f(\mathbf{x}) | \mathbf{x} \in S\} \quad ,$$

d. h. bei einem Maximierungsproblem liefert die Lösung des relaxierten Problems eine obere Schranke für das ursprüngliche Problem.

[10] Die *Floor-Funktion* $\lfloor y \rfloor$ bildet die Variable y auf die größte ganze Zahl ab, die y nicht überschreitet, z. B. $\lfloor 3.7 \rfloor = 3$, $\lfloor 2 \rfloor = 2$ oder $\lfloor -3.7 \rfloor = -4$.

Relaxierung kann zum Beispiel bedeuten, bestimmte Nebenbedingungen abzuschwächen oder diese, wie in Abschnitt 8.1.3.3 ausführlich behandelt, als Nebenbedingungen komplett zu vernachlässigen und stattdessen in die Zielfunktion bestrafend aufzunehmen (*Lagrange-Relaxierung*). Die meisten Verfahren im MILP-Umfeld verwenden jedoch eine Bereichsrelaxierung, d. h. bei der Lösung des MILP-Problems werden die Ganzzahligkeitsbedingungen $x \in \mathbb{N}_0$ vernachlässigt und durch $x \in \mathbb{R}_0^+$ ersetzt. Das so entstehende LP wird gelöst und liefert die Lösung x^0 und den Zielfunktionswert Z^o, der als obere Schranke Z^o für die gesuchte optimale Lösung weiter verwendet wird. In der Regel wird x^0 nicht alle Ganzzahligkeitsbedingungen erfüllen. In jedem Fall ist eine genauere Analyse, d. h. Sondierung der Verhältnisse der Relaxierung, nötig.

Die *Sondierung* ermöglicht, eine Verzweigung in einem Knoten zu beenden, wenn eines der nachstehenden *Verwerfungskriterien* erfüllt ist:

- das zugehörige LP-Problem ist unzulässig,
- x^0 erfüllt sämtliche Ganzzahligkeitsbedingungen und wird als ganzzahlig zulässiger Punkt Lösung weitergeführt,
- $Z^0 \leq Z^u$, Z^u ist der Zielfunktionswert eines bereits vorliegenden ganzzahlig zulässigen Punktes, oder
- $Z^0 \leq Z^u + \Delta Z$; sinnvolle Trennung der Knoten mit Hilfe der Minimalverbesserung [engl.: *addcut*] ΔZ.

Eine wesentliche Komponente des B&B-Verfahrens ist es, den kombinatorischen Baum so zu durchlaufen, dass möglichst viele Teile davon aufgrund obiger Verwerfungskriterien [engl.: *pruning criteria*] verworfen werden können. Liegt eine der beiden zuletzt angeführten Fälle vor, so endet die Verzweigung im Baum, da die Fortführung weitere Nebenbedingungen einbeziehen würde und somit zu Zielfunktionswerten $Z' \leq Z^0$ führen würde. Insbesondere die Einführung der Minimalverbesserung ΔZ kann sehr nützlich sein. Ist z. B. die Zielfunktion in einem Minimierungsproblem ganzzahlig, so werden mit $\Delta Z = 1$ lediglich noch die Kandidaten betrachtet, die mindestens um eins kleiner als der Zielfunktionswert eines bereits vorliegenden ganzzahlig zulässigen Punktes sind.

Separierung und weitere Verzweigungen sind erforderlich, wenn die obigen Verwerfungskriterien nicht auf die optimale Lösung x^0 des vorliegenden LP-Problems zutreffen. Beschränkt man sich auf *binäre* Separierung, so wird die für x zulässige Menge $\mathcal{T}$ in zwei disjunkte Mengen $\mathcal{T} = \mathcal{T}_1 \cup \mathcal{T}_2$ geteilt

$$\begin{aligned} \mathcal{T}_1 &= \mathcal{T} \cap \{\mathbf{x} \in \mathbb{R}_+^n \mid \mathbf{d}^{\mathrm{T}}\mathbf{x} \leq d_0\} \\ \mathcal{T}_2 &= \mathcal{T} \cap \{\mathbf{x} \in \mathbb{R}_+^n \mid \mathbf{d}^{\mathrm{T}}\mathbf{x} \geq d_0 + 1\} \quad , \end{aligned}$$

wobei es sich bei $\mathbf{d}$ um einen Vektor und bei $\mathbf{d}^{\mathrm{T}}\mathbf{x}$ um ein Skalarprodukt handelt. Hierbei wird $(\mathbf{d}, d_0) \in \mathbb{N}_+^{n+1}$ so gewählt, dass $d_0 \leq \mathbf{d}^{\mathrm{T}}\mathbf{x}^0 \leq d_0 + 1$ und $\mathbf{x}^0$ somit in beiden Teilproblemen $\mathcal{T}_1(LP)$ und $\mathcal{T}_2(LP)$ unzulässig ist. *Beispiel:* Sei $n = 1$ und $x^0 = 1.7$. Mit $(\mathbf{d}, d_0) = (3, 5)$ werden die beiden Teilprobleme durch $3x \leq 5$ und $3x \geq 6$ erzeugt; mit $(\mathbf{d}, d_0) = (1, 1)$ erhält man die beiden Teilprobleme durch $x \leq 1$ und $x \geq 2$.

In der Praxis sind eigentlich nur zwei spezielle Separierungen von $(\mathbf{d}, d_0)$ in Gebrauch: Die *Variablen-(Zwei)Teilung* und die *GUB-(Zwei)Teilung*, deren Name sich aus dem englischen Begriff „*general upper bound divison*" ableitet; die *direkte Teilung* (für ganzzahliges x_j in den Schranken $0 \leq x_j \leq X_j^+$ die separate Betrachtung jeden ganzzahligen Wertes $x_j = 0$, $x_j = 1, ...,$ $x_j = X_j^+$) erweist sich als nicht sehr effizient und findet deshalb in kommerziellen Paketen keine Verwendung.

Die *Variablen-(Zwei)Teilung* kann mit Hilfe des folgenden theoretischen Konzeptes (Nemhauser & Wolsey 1998,[222]) beschrieben werden. Man wählt für $\mathbf{d}$ den j^{ten} Einheitsvektor in $\mathbb{R}^n$, also $\mathbf{d} = \mathbf{e}_j$, und $d_0 = \lfloor x_j^0 \rfloor$. Die Teilung wird also durch die beiden Ungleichungen

$$x_j \leq d_0 \quad , \quad x_j \geq d_0 + 1$$

vorgenommen. Handelt es sich bei x_j um eine binäre Variable, für die noch $0 < x_j < 1$ gilt, so liefern diese Verzweigungen sogleich die Bedingungen $x_j = 0$ und $x_j = 1$. Ein wesentlicher Vorteil der Variablen-Teilung besteht darin, dass diese nur einfache untere und obere Schranken als weitere Restriktionen zu der bestehenden LP-Relaxierung hinzu addiert. Damit kann direkt das duale Simplexverfahren angewendet werden; die Basis vergrößert sich nicht.

Die GUB-(Zwei)Teilung wird vorgenommen, wenn das Problem eine Nebenbedingung der Form $\sum_{j \in Q} x_j = 1$ enthält; dies ist z. B. bei der Verwendung der in Abschnitt 5.7 beschriebenen SOS-1- oder SOS-2-Mengen der Fall. Ist Q_1 eine nichtleere Teilmenge von Q, so erfolgt die Teilung gemäß $\sum_{j \in Q_1} x_j = 1$ und $\sum_{j \in Q \setminus Q_1} x_j = 0$.

Das B&B-Verfahren lässt sich damit in die folgende *Schleifenstruktur* mit Schritten 1-5 fassen:

1. *Initialisierung* der unteren Schranke $Z^u = -\infty$ und Liste der aktiven Knoten
2. *Knotenwahl*: Wahl eines Knotens aus der Liste
3. Lösung des zugehörigen LPs $\rightarrow x^0$ und Z^0
4. Test der Verwerfungskriterien an : x^0 zulässig $Z^u \rightarrow Z^u = Z^0 \rightarrow$ Schritt 2
5. *Variablenwahl* und Wahl der Verzweigungsrichtung, Schätzung der Degradierung der ganzzahligen Lösung und obere Schranke, Hinzufügung der beiden neuen Unterprobleme zur Liste der aktiven Knoten $\rightarrow$ Schritt 2

Damit werden einige Freiheitsgrade und Steuermöglichkeiten im B&B-Verfahren offenbar:

- Wahl des Knoten,
- Wahl der Richtung,
- Wahl der Variablen,
- Berechnung der Schätzungen.

Das B&B-Verfahren hat zwar in schlechten Fällen exponentielles Laufzeitverhalten, erweist sich aber in der Praxis häufig als gutartig, insbesondere auch durch geeignete Nutzung dieser Freiheitsgrade, die die Effizienz des Verfahrens erheblich verbessern.

4.3.2.1 Knotenwahl

Die Knotenwahl spielt eine wichtige Rolle in einem B&B-Verfahren. Zur Wahl der Knoten gibt es zunächst zwei komplementäre Strategien:

- *A-priori*-Regeln: Der Durchlauf durch den Entscheidungsbaum wird im vorhinein fixiert.
- Adaptive Regeln: Sie beziehen Informationen über die aktuellen Knoten mit ein.

Oft kann es eine gute Strategie sein, mit einer *Tiefensuche* [engl.: *depth-first*] mit *backtracking* bei der B&B-Suche zu beginnen und einen guten ganzzahlig zulässigen Punkt so schnell wie möglich zu finden. Damit können bereits große Teile des Baumes eliminiert und die B&B- Suche erheblich beschleunigt werden. Wird ein Knoten wegen der durch einen *Akzeptanzwert* [engl.: *cut-off*] implizierten Wertdominanz des aktuellen LP-Problems, das noch keine ganzzahlige Lösung produziert hat, oder weil das mit ihm assoziierte LP-Problem keine Lösung besitzt, verworfen, oder weil ein ganzzahlig zulässiger Punkt bestimmt wurde, der besser ist als der beste existierende ganzzahlig zulässige Punkt, so wird in den betrachten binären Baumsuchverfahren sein Geschwisterknoten, d. h. das parallele Unterproblem gleichen Niveaus, untersucht. Kann der vorliegende Knoten nicht verworfen werden [engl.: *pruned*], so wird als nächstes einer seiner beiden Söhne untersucht. *Backtracking* bedeutet, dass nach Verwerfung eines Sohnes der Baum zurückverfolgt wird, bis ein noch nicht bearbeiteter Sohn gefunden wird. Beschließt man noch, dass stets der linke Sohn vor dem rechten untersucht wird, so liegt eine vollständige Fixierung der Verzweigungswege im Sinne der *a priori*-Regel vor. Mit dieser Strategie sind zwei wichtige Vorteile verbunden:

- Die LP-Relaxierung eines Sohnes erhält man aus der LP-Relaxierung des Vaters durch Hinzufügen einer einfachen *oberen* oder *unteren Schranke*. Dieses Problem kann in vielen Fällen direkt mit dem dualen Simplexverfahren ohne aufwendige Matrixinversion gelöst werden.
- Bei vielen praktischen Problemen findet man zulässige Knoten eher in der Tiefe als in der Breite [48]; dies müssen allerdings nicht notwendigerweise bereits gute Lösungen sein.

Im Gegensatz zur Tiefensuche werden bei der *Breitensuche* [engl.: *breadth-first*] zunächst sämtliche Knoten einer bestimmten Ebene untersucht, bevor man zur nächst tieferen Ebene fortschreitet. Ein Nachteil dieser Methode ist, dass man sehr viele Knoten generiert, aber die Methode kann erfolgreich sein. Innerhalb dieser Strategie oder auch in anderen adaptiven Knotenwahlverfahren können die folgenden heuristischen Auswahlregeln eingesetzt werden:

- *Beste Obere Schranke*: Hier wird der Knoten mit der größten oberen Schranke gewählt in der Hoffnung, dass auch die nachfolgenden verschärften Relaxierungen noch möglichst große Werte liefern und damit, wenn man einen zulässigen Knoten findet, eine möglichst gute untere Schranke gefunden werden kann.
- *Beste Schätzung*: Dieses Verfahren basiert auf einer Schätzung, nach der ein Zweig gewählt wird, der höchst wahrscheinlich die optimale Lösung enthält.
- *Forrest-Hirst-Tomlin-Kriterium*: Nach Forrest-Hirst-Tomlin (1974,[93]) wird hierbei in einem Maximierungsproblem der Knoten mit dem höchsten Wert von

$$\frac{z^E - z^{IP}}{z - z^E} \tag{4.3.24}$$

gewählt, wobei z der Zielfunktionswert des aktuellen Knotens ist, z^E ist eine Schätzung des Zielfunktionswertes der ganzzahligen Lösung, die aus dem aktuellen Knoten folgen könnte, und z^{IP} der Zielfunktionswert des augenblicklich besten ganzzahlig zulässigen Punktes ist. Liegt bereits aus einem ähnlichen Problem eine Lösung mit Zielfunktionswert z_*^{IP} vor, so kann dieser Wert anstelle von z^{IP} in (4.3.24)

verwendet werden, wenn noch keine ganzzahlige Lösung des aktuellen Problems vorliegt. Der Schätzwert z^E folgt aus der Betrachtung der *Pseudo-Kosten* [26], die im Zusammenhang mit den im Simplexverfahren verwendeten Einheitsraten der Degradierung der reduzierten Kosten d_j bei Wertänderung einer Variablen stehen. In ähnlicher Weise können separate *plus* und *minus Pseudo-Kosten* berechnet werden, die mit dem An- oder Abstieg einer Binärvariablen von z. B. 0.4 auf 1, bzw. 0.4 auf 0 verbunden sind; je höher die Pseudo-Kosten, desto unwahrscheinlicher die Verzweigung in diese Richtung. Dies ist z. B. dann sehr nützlich, wenn mit Hilfe einer Binärvariablen die *Rüst*-Kosten beschrieben werden und die obere Schranke X einer kontiniuerlichen Variablen x in der Ungleichung $x \leq X\delta$. In der LP-Relaxierung nimmt δ dann relativ kleine Werte nahe bei Null an und der Lösungsalgorithmus verzweigt dann vorzugsweise nach $\delta = 0$. Erhöht man die *minus Pseudo-Kosten*, so kann dies vermieden werden. Bei diesem Kriterium werden also jene Knoten bevorzugt, bei denen die Differenz $z - z^E \geq 0$ aus oberer Schranke und Schätzung möglichst klein ist, bzw. auch jene Knoten, bei denen die Schätzung z^E größer als die untere Schranke z^{IP} ist.

4.3.2.2 Variablenwahl

Hat man einen aktiven Knoten gewählt, so bleibt die Frage offen, nach welcher Variablen die Teilung erfolgen soll. Hier bieten sich

- vom Benutzer vorgegebene Prioritäten oder
- geschätzte Degradierung

an. Der Hintergrund des ersten Verfahrens ist, dass keine allgemeine robuste Strategie verfügbar ist und der Benutzer noch am besten die ökonomische Relevanz und den Einfluss bestimmter Variablen abschätzen kann. Oft zeigt es sich, dass nach der Fixierung einiger weniger Variablen, die noch keine ganzzahligen Werte angenommen haben, sämtliche anderen automatisch ganzzahlig zulässig werden.

Die Abschätzung der Degradierung [222] versucht, ausgehend von einer vorliegenden Variablen $x_j = \lfloor x_j \rfloor + f_j$, die Verringerung $D_j{}^- = p_j{}^- f_j$ und $D_j{}^+ = p_j{}^+(1 - f_j)$ der Zielfunktion für den linken und rechten Sohn abzuschätzen, wobei die Koeffizienten $(p_j{}^-, p_j{}^+)$ z. B. aus Informationen des dualen Simplexverfahrens ableitbar sind. Ein mögliches Kriterium ist das der *maximum integer feasibility* ($p_j{}^- = p_j{}^+ = 1$), d. h. max $\min(D_j{}^-, D_j{}^+)$. Dieses Verfahren basiert auf der Annahme, dass jene Variable x_j, deren kleinste Verringerung hinsichtlich ihrer Söhne, also $\min(D_j{}^-, D_j{}^+)$, von allen Variablen die maximale ist, wohl von besonderer Bedeutung sein muss, wenn man Ganzzahligkeit erreichen möchte. In diesem Fall wird man als nächstes auch den Sohn mit der kleineren Degradierung in der Zielfunktion wählen. Das alternative Kriterium max $\max(D_j{}^-, D_j{}^+)$ geht insbesondere bei Kenntnis einer unteren Schranke davon aus, dass jene Variablen, die einen maximalen Abstieg versprechen, leicht zu Zielfunktionswerten führen können, die unterhalb der unteren Schranke liegen und damit zur Verwerfung des Knotens führen; entsprechend wird man hier den Sohn mit $\max(D_j{}^-, D_j{}^+)$ wählen.

4.3.2.3 Das B&B-Verfahren im Überblick, Abbruchkriterien

Hier seien nochmals kurz die wesentlichen Merkmale und Rechenschritte des B&B-Verfahrens unter einem etwas anderen Blickwinkel zusammengefasst. Der Begriff *Verzweigung* [engl.: *branch*] in B&B deutet auf den Zerlegungs- und Verzweigungsprozess hin, der neue Lösungen produzieren oder den Nachweis der Optimalität erbringen kann. In diesem Prozess werden untere und obere Schranken generiert, um eine vollständige Enumeration zu vermeiden; dies macht das B&B-Verfahren zu einem *impliziten Enumerationsverfahren*. Nach der Initialisierung wird mit Hilfe der LP-Relaxierung der erste Knoten erzeugt. Zu diesem Zeitpunkt ist die Knotenwahl trivial (es wird gerade die LP-Relaxierung, d. h. der erste Knoten gewählt), später wird sie mit Hilfe der in Abschnitt 5.13.2 beschriebenen Heuristiken durchgeführt. Das B&B-Verfahren nach Art von Dakin (1965,[56]) mit LP-Relaxierung kennt drei Verwerfungskriterien für die Knoten: Unzulässigkeit, Optimalität und Dominanz. In einem Maximierungsproblem führen die ermittelten ganzzahligen Lösungen zu einer monoton wachsenden Folge z^{IP} unterer Schranken, während die LP-Probleme im Baum eine monoton fallende Folge oberer Schranken z^{LP} erzeugen. Hierbei ist noch die Minimalverbesserung α zu beachten, die dazu führt, dass eine weitere ganzzahlige Lösung nur als Verbesserung akzeptiert wird, wenn ihr Wert um α besser (Dominanzkriterium) ist. Kommen die Verwerfungskriterien nicht zur Anwendung, so wird eine weitere Verzweigung durch die Variablenzweiteilung initialisiert: Für einen nichtganzzahligen Wert y_j^* werden zwei Unterknoten bzw. Unterprobleme durch Hinzufügung der zusätzlichen Schranken $y_j \leq \lfloor y_j^* \rfloor$ bzw. $y_j \geq \lfloor y_j^* \rfloor + 1$ erzeugt; in Abschnitt A.1.3 ist erkennbar, warum Schranken wesentlich effizienter als allgemeine Nebenbedingungen behandelt werden können.

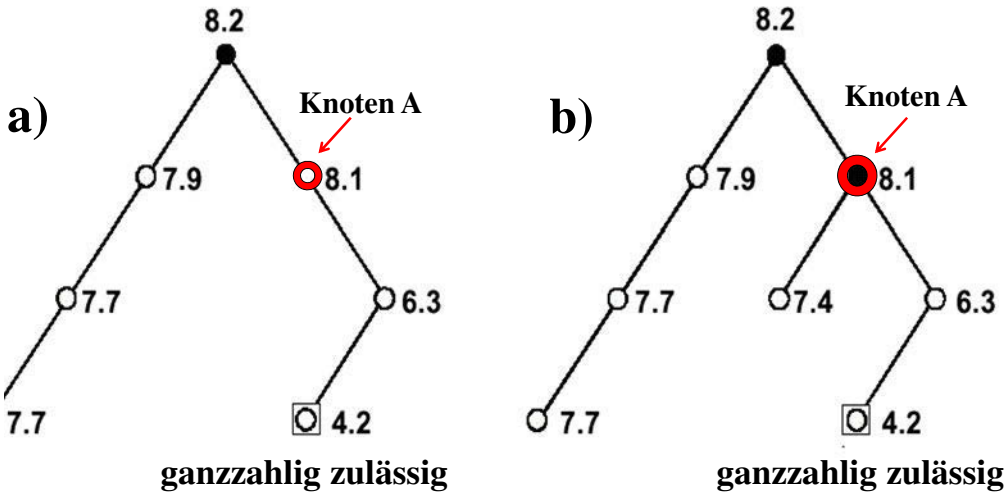
Das B&B-Verfahren terminiert nach einer endlichen Anzahl von Schritten. Nachdem die Knotenliste vollständig abgearbeitet wurde, ist entweder die optimale ganzzahlige Lösung gefunden oder für das Problem existiert kein ganzzahlig-zulässiger Punkt. In der Praxis wird man häufig nicht warten wollen, bis die Knotenliste komplett abgearbeitet ist, sondern beendet z. B. die Suche nach der ersten gefundenen ganzzahligen Lösung und verwendet die ermittelten oberen und unteren Schranken zur Abschätzung der Qualität dieser Lösung in folgender Weise. Das B&B-Verfahren produziert für ein Maximierungsproblem während seiner Laufzeit die Schrankeninformation

$$z^{LP} \geq z^* \geq z^{IP} \quad ,$$

wobei z^* den Zielfunktionswert der noch (unbekannten) optimalen Lösung, z^{IP} (möglicherweise $-\infty$) der Zielfunktionswert der bisher zu diesem Zeitpunkt besten gefundenen ganzzahligen Lösung bezeichnet und $z^{LP} = \max_i\{z_i^{LP}\}$ mit den Zielfunktionswerten z_i^{LP} der LP-Probleme im *aktiven* Knoten i (Knoten, die noch nicht verworfen wurden). In Abbildung 4.4(a), sind dies $z^{IP} = 4.2$ und $z^{LP} = 8.1$ (nicht jedoch 8.2, da der Ausgangsknoten nicht länger aktiv ist). In Abbildung 4.4(b) wurden beide Unterknoten des Knotens A untersucht. Deshalb ist Knoten A nicht länger aktiv und das Maximum der betrachteten LP-Lösungen aller aktiven Knoten ist 7.9, was zu einer verbesserten Ganzzahligkeitslücke von $\Delta = 3.7$ führt.

Sobald $\Delta := z^{LP} - z^{IP} \leq \varepsilon$ gilt, ist die optimale Lösung mit einer Toleranz von ε bestimmt. Damit ist es im B&B-Verfahren sinnvoll, den nächsten Knoten so zu wählen, dass die Ganzzahligkeitslücke Δ verkleinert wird. Eine Möglichkeit, dies zu tun, ist einen Knoten zu wählen, für den eine gute Chance besteht, das die nächste ganzzahlige Lösung, die gefunden wird, einen Zielfunktionswert hat, der besser ist als der gegenwärtig beste

Abbildung 4.4 Zwei Branch & Bound - Bäume. Der Wert neben jedem Knoten ist der Zielfunktionswert, der aus der LP-Relaxierung und den mit dem Knoten assoziierten Schrankenbedingungen resultiert. Offene Kreise stellen aktive Knoten dar, d. h. mindestens einer der beiden Kindknoten wurde noch nicht ausgewertet. Nachdem nach Auswertung des Knotens mit Wert 4.2 eine ganzzahhlig zulässige Löesung gefunden wurde, hat die Ganzzahligkeitslücke in a) den Wert $\Delta = 8.1 - 4.2 = 3.9$. In b) sieht man, dass Δ nach Auswertung *beider* vom Knoten A abstammenden Kindknoten den reduzierten Wert $\Delta = 7.9 - 4.2 = 3.7$ annimmt.



Wert z^{IP}. Die andere Möglichkeit besteht darin, auf den Knoten zu verzweigen, der den höchsten z_i^{LP}-Wert hat; hierbei ist die Überlegung, dass die Unterknoten dieses Knotens keinesfalls größere Werte z_i^{LP} haben und im besten Falle allesamt kleinere Werte, wodurch z^{LP} abnimmt; in der Praxis beobachtet man, dass sich die Schranke z^{LP} meist nur sehr langsam verbessert.

Bei praktischen Problemen wird die B&B-Suche im Falle zu großer Rechenzeiten gelegentlich vorzeitig abgebrochen. In diesem Fall schließen die oberen und unteren Schranken $z^U := z^{LP}$ und $z^L := z^{IP}$ den optimalen Zielfunktionswert ein, liefern in einem Maximierungsproblem die Ganzzahligkeitslücke Δ, sowie die prozentuale Ganzzahligkeitslücke, die sich unter der Voraussetzung $z^L > 0$ in einem Maximierungsproblem zu

$$\Delta^P_{\max} := 100\frac{\Delta}{z^L} = 100\frac{z^U - z^L}{z^L} = 100\frac{z^{LP} - z^{IP}}{z^{IP}} \tag{4.3.25}$$

berechnet und in einem Minimierungsproblem ($z^L := z^{LP}$, $z^U := z^{IP}$) unter der Voraussetzung $z^U > 0$ die Gestalt

$$\Delta^P_{\min} := 100\frac{\Delta}{z^L} = \frac{z^U - z^L}{z^U} = 100\frac{z^{IP} - z^{LP}}{z^{IP}} \tag{4.3.26}$$

annimmt und besagt, dass die gefundene Lösung schlimmstenfalls $\Delta^P_{\max}\%$ bzw. $\Delta^P_{\min}\%$

vom exakten Optimum entfernt ist. Man kann natürlich auch die prozentuale Ganzzahligkeitslücke als Abbruchkriterium vorgeben und fordern, dass das B&B-Verfahren terminiert, wenn z. B. $\Delta^P_{\max} \leq 1$ erreicht wurde.

4.3.2.4 Konvexe Hülle und Lösungsverhalten des B&B-Verfahrens

Zwar kann das B&B-Verfahren in vielen Fällen leicht Probleme mit einigen hunderten oder gar tausenden von ganzzahligen Variablen lösen, aber einige Probleme mit sogar weniger als 100 ganzzahligen Variablen lassen sich überhaupt nicht damit in einer sinnvollen Rechenzeit lösen. Damit ist schwer abzuschätzen, wann ein MILP-Modell nun gutartig oder als problematisch und überhaupt lösbar anzusehen ist. Diese Schwierigkeit hat ihre Ursache in einigen mathematischen Eigenschaften von MILP-Problemen und dem B&B-Verfahren selbst. Typischerweise zeigen zwei Instanzen eines Problems mit gleicher Anzahl von m Nebenbedingungen und n Variablen, von denen das eine als LP-Problem, das andere als MILP-Problem gelöst wird, einen erforderlichen Rechenaufwand, der beim MILP-Problem 2^n mal so groß ist wie bei der Lösung des LP-Problems. Weiterhin beobachtet man, dass bei einer Änderung der Daten die benötigte Rechenzeit zur Lösung des LP-Problem annäherend gleich bleibt, bei der Lösung des MILP-Problems aber erheblich variiert. Zum Glück zeigen praktische Probleme aber nicht immer das schlimmst mögliche Verhalten und diesen drastischen Anstieg in der Rechenzeit. Daher sollte auch nicht aus dem oben Gesagten gefolgert werden, dass MILP-Modelle nicht erstellt und angewendet werden sollten, sondern lediglich, dass eine gewisse Vorsicht und Sorgfalt erforderlich ist.

Was genau zu tun ist, um ein gutartiges Modell zu entwickeln, lässt sich nicht ganz ohne einen tieferen mathematischen Hintergrund erklären: Die Formulierung sollte hinsichtlich der Variablenwahl und der resultierenden Nebenbedingungen so gewählt sein, dass die LP-Relaxierung so nahe wie möglich an die *konvexe Hülle* [engl.: *convex hull*] heranreicht. Der Begriff konvexe Hülle wird an mehreren Stellen in diesem Buch verwendet; in Abbildung 4.5 ist ein Beispiel dafür gezeigt. Es ist das Polyeder kleinsten Volumens, das gerade noch alle zulässigen Punkte des MILP-Problems enthält. Es lässt sich zeigen, dass die konvexe Hülle stets durch eine Menge linearer Nebenbedingungen beschreibbar ist. Leider ist es in den meisten Fällen nicht möglich, dieses System von Nebenbedingungen explizit aufzuschreiben. In den Fällen, in denen es möglich ist, ist die Lösung des relaxierten LP-Problems eingeschränkt auf den zulässigen Bereich der konvexen Hülle identisch mit der Lösung des ursprünglichen Problems.

4.3.3 Schnittebenenverfahren und Branch&Cut

Schnittebenen-Verfahren [engl.: *cutting-plane methods*] beginnen wie das B&B-Verfahren mit einer LP-Relaxierung, reduzieren aber dann den zulässigen Bereich des relaxierten Problems durch zusätzliche lineare Ungleichungen. Auch hier wird die optimale Lösung meist vor Abbruch des Verfahrens gefunden, aber es braucht eine zusätzliche Zeit, um die Optimalität der Lösung zu beweisen. Das erste Schnittebenen-Verfahren wurde von Gomory (1958,[111]) eingeführt; diese Technik wurde durch Glover (1968,[108]) und Young (1968,[299]) weiter verfeinert. Während das B&B-Verfahren einige weniger ergiebige Bereiche des Suchbaums abarbeitet, nähern sich Schnittebenenverfahren der optimalen Lösung etwas gezielter, konvergieren in der späten Phase eher langsam; daher findet man diese Technik nicht häufig in kommerzieller Software implementiert.

Abbildung 4.5 Diese Abbildung zeigt die LP-Relaxierung, die konvexe Hülle und die Konturlinien der Zielfunktion.

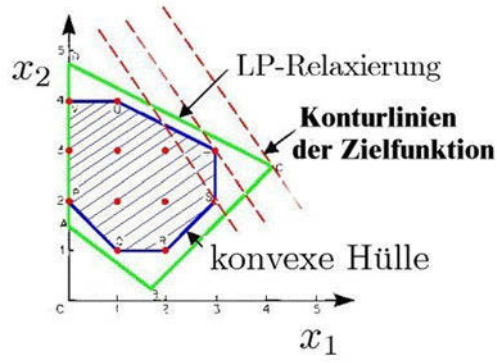
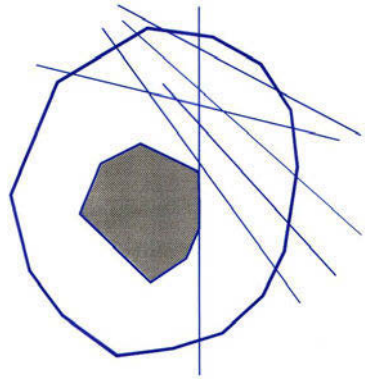


Abbildung 4.6 LP-Relaxierung (äußere Begrenzung), konvexe Hülle (schraffierter Bereich) und Schnittebenen. Mit Hilfe der Schnittebenen lassen sich unerwünschte Gebiete zwischen der LP-Relaxierung und der konvexen Hülle abtrennen.



B&C-Verfahren [*cf.* Padberg & Rinaldi (1987,[229]) oder Savelsbergh *et al.* (1994,[258])] kombinieren die Vorteile von B&B- und Schnittebenenverfahren und haben Einzug in kommerzielle Software gehalten. Während B&B jedoch für jeden nichtganzzahligen Wert einer ganzzahligen Variablen zwei neue Unterprobleme erzeugt, wird bei B&C eine lineare Ungleichung hinzugefügt, die den fraktionalen Wert ausschließt. Daher funktioniert das Verfahren wie ein Baum-Such-Verfahren, aber in jedem Knoten wird statt der Verzweigung eine *zulässige Ungleichung* oder ein *Schnitt* [engl.: *valid inequality*, *cut*] hinzugefügt, wodurch ein Teil des zulässigen Bereichs der LP-Relaxierung wie in Abbildung 4.6 abgetrennt wird, ohne dabei ganzzahlig zulässige Lösungen zu verlieren. B&C kommt in zwei Varianten a) und b) vor: Es werden nur Ungleichungen erzeugt, die a) für das Problem als ganzes zulässig oder b) nur für bestimmte Knoten zulässig sind. Es wird also sparsam mit zusätzlichen Ungleichungen umgegangen, denn ihre Anzahl nimmt exponentiell zu, wodurch das resultierende LP-Problem schnell zu groß wird. Wichtig für ein gutes

B&C-Verfahren ist daher, wie z. B. in Parija *et al.* (1999,[231]) oder Günlük & Pochet (2001,[123]) beschrieben, effiziente Schnitte zu erzeugen, die die gegenwärtige LP-Lösung eliminieren. In Variante a) werden wirksame Schnitte progressiv hinzugefügt, während in b) nur für einen bestimmten Knoten wirksame Schnitte addiert werden. Beide Varianten erlauben, derartige Ungleichungen wieder zu entfernen.

4.3.4 Branch&Price: Optimierung mit Spaltenerzeugung

Branch&Price (B&P) stellt eine Erweiterung des B&B-Verfahrens mit LP-Relaxierung für MILP-Probleme mit sehr vielen, d. h. einigen Millionen Variablen dar [281]. Wegen dieser großen Anzahl von Variablen lässt man zunächst die meisten Variablen bei der LP-Relaxierung unberücksichtigt; dies ist gar nicht mal eine so schlechte Startheuristik, denn die meisten Variablen in großen Problemen nehmen ohnehin den Wert Null an. Zum Nachweis der Optimalität wird ein Pricing-Optimierungsproblem gelöst, wobei neu in die Basis aufzunehmende Variablen identifiziert werden und die LP-Relaxierung gegebenfalls erneut durchgerechnet wird. Verzweigt wird, falls die LP-Relaxierung nicht alle Ganzzahligkeitsbedingungen erfüllt. Die Spaltenerzeugung findet schließlich in jedem Knoten des B&B-Baumes statt. Hat man erst einmal ein Verfahren zur Verfügung, das eine große Anzahl von Variablen handhaben kann, so kann das sehr nutzbringend eingesetzt werden, denn bei praktischen Problemen wird häufig folgendes beobachtet:

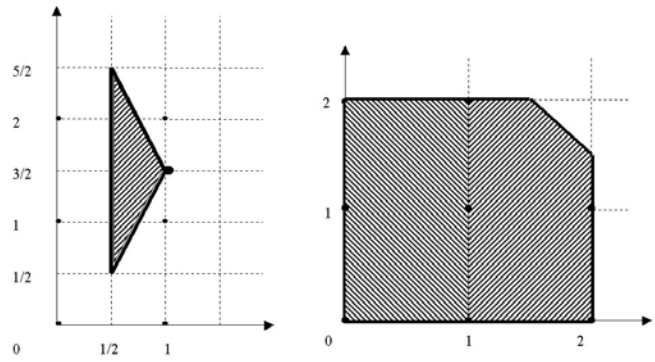
1. Kompakte Formulierungen von MILP-Problemen, gerade im Scheduling oder in der Routenplanung, haben oft sehr schwache LP-Relaxierungen. Häufig kann die Modellgüte erheblich verbessert werden, indem man eine Reformulierung mit einer riesigen Anzahl von Variablen entwickelt.
2. Kompakte Formulierungen von MILP-Problemen enthalten oft symmetrische Strukturen, die bei Standard-B&B-Verfahren zu ineffizientem Verhalten führen; eine erheblich größere Anzahl von Variablen kann diese Symmetrie zerstören.
3. Die Spaltenerzeugung liefert eine Dekomposition des Problems in ein Haupt- und ein Unterproblem, die in manchen Fällen zu algorithmischen Verbesserungen führt.

Die Grundidee von B&P [23] – siehe auch Abschnitt 8.1.3.2.3 – scheint einfach, insbesondere wenn das Problem klassisch nach Dantzig & Wolfe (1960,[59]) zerlegt wird. Technische Schwierigkeiten, z. B. kompatible Verzweigungsregeln, treten häufig erst dann auf, wenn das Problem nicht zerlegt wird und die Modellformulierung, wie z. B. bei Set-Partioning Problemen, sehr viele ganzzahlige Variablen hat. Desrosiers *et al.* (1995,[71]) erläutern dies für Routenplanungs- und Scheduling-Probleme, Savelsbergh (1997,[256]) wendet B&P auf das Verallgemeinerte Zuordnungsproblem [engl.: *generalised assignment problem*, *GAP*] an, das wir in Abschnitt 8.1.3.3.2 mittels Lagrange-Relaxierung lösen.

4.3.5 L-Klassen – Enumeration

Die von Kolokolov vorgeschlagene Idee der L-Klassen-Enumeration, die von ihm und anderen ausgearbeitet und auf ganzzahlige Optimierungsprobleme angewendet wurde – siehe z. B. [183] und weitere darin enthaltene Referenzen auf meist russische Veröffentlichungen – basiert auf regulären Zerlegungen des zulässigen Bereichs der LP-Relaxierung und liefert eine Reihe interessanter Einblicke in die gemischt-ganzzahlige Optimierung,

Abbildung 4.7 Beispiele zur L-Klassen-Enumeration. Gezeigt sind für jedes Beispielproblem der zulässigen Bereich sowie die Menge der zulässigen Punkte der LP-Relaxierung.



z. B. im Zusammenhang mit der Stabilität der optimalen Lösung und der relaxierten Mengen. Wenngleich diese Methode wenig verbreitet und in der Praxis kaum an größeren Problemen getestet ist, soll sie doch in diesem Buch behandelt werden, da sie einige Ideen enthält, die auch einen Praktiker motivieren können, ein vorliegendes Problem vielleicht einmal aus einem anderen Blickwinkel zu betrachten.

Der Begriff *L-Klassen* leitet sich von *lexikographisch* geordneten Klassen ab. Die Methode der L-Klassen-Enumeration wurde zwar zur Lösung ganzzahliger Optimierungsprobleme vorgeschlagen; sie lässt sich jedoch auch auf gemischt-ganzzahlige Optimierungsprobleme übertragen. Insbesondere zur Lösung des Erfüllbarkeitsproblems und des Set-Covering-Problems zeigt sich dieses Verfahren als effizient.

Mit Hilfe regulärer Zerlegungen des Euklidischen Raumes $\mathbb{R}^n$ ist es möglich, die Struktur und Algorithmen ganzzahliger Optimierungsprobleme zu untersuchen. Sei F eine *Zerlegung*[11] des $\mathbb{R}^n$, X eine beliebige Menge in $\mathbb{R}^n$ und die *Faktormenge*[12] X/F eine Zerlegung der Menge X durch F. Die Zerlegung F heißt *regulär*, wenn sie die folgenden Bedingungen erfüllt.

1. Jeder ganzzahlige Punkt $\mathbf{z} \in \mathbb{Z}^n$ bildet eine (Äquivalenz-)Klasse der Zerlegung; die anderen Klassen heißen *fraktionale Klassen* und enthalten nur nichtganzzahlige Punkte.
2. Ist X beschränkt, so ist die Faktormenge X/F endlich.
3. Für jede Menge $X \subset \mathbb{R}^n$ und $\mathbf{z} \in \mathbb{Z}^n$ gilt die Gleichung $(X + \mathbf{z})/F = (X/F) + \mathbf{z}$.

[11] Eine Zerlegung F einer Menge $\mathcal{M}$ ist in der Mengenlehre ein System disjunkter Mengen $\mathcal{M}_i \neq \emptyset$, deren Vereinigung identisch mit $\mathcal{M}$ ist.

[12] Die *Faktormenge* $\mathcal{M}/F$ ist die Menge aller *Restklassen* von Elementen von $\mathcal{M}$ bezüglich F, d. h. $\mathcal{M}/F := \{[x]_F | x \in \mathcal{M}\}$ mit den Restklassen $[x]_F := \{y | y \in \mathcal{M} \wedge x \sim y\}$, wobei $\sim$ die zugrunde liegende Äquivalenzrelation, in unserem Falle später die lexikographische Äquivalenzrelation $\succ$, ist. Im Folgenden unterscheiden wir nicht strikt zwischen der Restklasse $[x]_F$ als Element des Faktorraumes $\mathcal{M}/F$ und der Restklasse als Untermenge $\{y | y \in \mathcal{M} \wedge x \sim y\}$ von $\mathcal{M}$, die wir dann auch nur kurz mit x bezeichnen.

Die erste Bedingung ist mit der Tatsache verbunden, dass viele Algorithmen Verfahren enthalten, die nichtganzzahlige Punkte eliminieren, während sie die ganzzahlig zulässigen Punkte weiterführen. Die zweite Bedingung wird benötigt, um zu garantieren, dass ein Verfahren für beschränkte Mengen, die z. B. aus einer LP-Relaxierung resultieren, nach endlichen vielen Schritten terminiert. Schließlich beschreibt die dritte Bedingung die Strukturerhaltung einer Zerlegung bei Translation um einen ganzzahligen Vektor.

Die L-Klassen-Enumerationsverfahren basieren auf L-Partitionen bzw. L-Zerlegungen der Menge X, die wie folgt definiert sind. Zwei Punkte $\mathbf{x}$ und $\mathbf{y} \in \mathbb{R}^n$, für die $\mathbf{x} \succ \mathbf{y}$ gilt, heißen L-äquivalent, wenn es kein $\mathbf{z} \in \mathbb{Z}^n$ gibt, sodass gilt $\mathbf{x} \succeq \mathbf{z} \succeq \mathbf{y}$; sie gehören damit zur gleichen Klasse. Die Relationen $\succ, \succeq$ bezeichnen die *lexikographische Ordnungsrelation*[13]; so gilt z. B. $(0,1,0) \succ (0,0,1)$ oder auch $(1,0,0) \succ (0,1,1)$. Äquivalente Punkte bilden die Klassen der L-Partition. Die Zerlegung nach L-Klassen erfüllt die Bedingungen der Definition regulärer Zerlegungen. Weiterhin gelten z. B. die folgenden wichtigen Eigenschaften:

1. Jede fraktionale L-Klasse $V \in X/L$ kann als

$$V = X \cap \{\mathbf{x} | x_1 = a_1, ..., \ x_{r-1} = a_{r-1}, \ a_r < x_r < a_r + 1\} \quad ,$$

   dargestellt werden, wobei $a_j \in \mathbb{Z}$ für $j = 1, ..., r$ mit geeignetem $r \leq n$.

2. Seien X und X' nichtleere Mengen im $\mathbb{R}^n$. Die Menge X wird als lexikographisch größer als $X^{'}(X \succ X')$ bezeichnet, wenn für alle $x \in X$ und $x' \in X'$ gilt $x \succ x'$. Diese Relation etabliert eine lineare Ordnung auf dem Faktorraum $\mathbb{R}^n/L$. Ist X eine beschränkte Menge, so kann X/L wie folgt dargestellt werden:

$$X/L = \{V_1, ..., V_p\} \quad , \quad V_i \succ V_{i+1} \quad , \quad i = 1, ..., p-1 \quad .$$

Mit Hilfe der L-Struktur einer Menge lassen sich obere Schranken für die Anzahl der Iteration von Schnittebenenverfahren ableiten.

Im Folgenden wollen wir nun die Idee der L-Klassen-Enumeration im Hinblick auf eine algorithmische Implementierung näher erläutern. Der Hauptschritt besteht darin, sich von der lexikographisch größten L-Klasse zur nächsten entsprechend absteigender lexikographischer Ordnung zu bewegen; im schlimmsten Falle entspricht dies einer vollständigen Ausschöpfung des zulässigen Bereichs. Das L-Klassen-Enumerierungsverfahren wird hier für ein Maximierungsproblem

$$\max f(\mathbf{x}) \quad , \quad f(\mathbf{x}) = \mathbf{c}^{\mathrm{T}}\mathbf{x}$$

unter der Nebenbedingung $\mathbf{x} \in \mathcal{S}_0$ mit

$$\mathcal{S}_0 := \{\mathbf{x} \,|\mathsf{A}\mathbf{x} \leq \mathbf{b} \quad \wedge \quad \mathbf{x} \geq \mathbf{0} \quad \wedge \quad \mathbf{x} \in \mathbb{N}_0^n \} \quad ,$$

und $\mathbf{b} \in \mathbb{R}^m$ erläutert; die optimale Lösung des Problems sei mit $\mathbf{x}_*$ bezeichnet. Weiterhin sei $\mathcal{S} := \{\mathbf{x} \in \mathbb{R}^n | \mathsf{A}\mathbf{x} \leq \mathbf{b}, \mathbf{x} \geq \mathbf{0}\}$ die Menge aller zulässigen Punkte der LP-Relaxierung. Als abgeschlossene und beschränkte Menge besitzt $\mathcal{S}_0 \subset \mathbb{R}^n$ einen lexikographisch maximalen[14] Punkt, der im Folgenden mit $\mathbf{z}_* = \mathrm{lexmax}\mathcal{S}_0$ bezeichnet sei. So ist z. B. $\mathbf{z}_* = (1,0)$ der lexikographisch größte Punkt der Menge $\mathcal{S}_\mathrm{K} = \{(z_1, z_2) | z_1^2 + z_2^2 = 1\}$.

[13] In der lexikographischen Ordnungsrelation heisst ein Vektor $\mathbf{x}$ lexikographisch größer als $\mathbf{y}$, wenn gilt: $x_k > y_k$, wobei k die erste Komponente ist, in der sich $\mathbf{x}$ und $\mathbf{y}$ unterscheiden. Identifiziert man beispielsweise ein solches k mit $k \geq 2$, so gilt also: $x_i = y_i$ für alle $1 \leq i < k$.

[14] Der lexikographisch maximale Punkt eines Polyeders bzw. Menge $\mathcal{M}$ kann z. B. mit einem

Die Grundidee der L-Klassen-Enumerierung besteht darin, $\mathcal{S}$ lexikographisch in geordnete Untermengen zu zerlegen bzw. die Faktormenge $\mathcal{S}/L = \{V_1, V_2, \ldots, V_p\}$ lexikographisch absteigender L-Klassen mit $V_i \succ V_{i+1}$ zu betrachten und die optimale Lösung durch Tests einiger dieser Untermengen zu bestimmen. Eine begrenzte Anzahl spezieller zulässiger Ungleichungen kann verwendet werden, wenn das Verfahren den nächsten ganzzahlig zulässigen Punkt sucht. Zu Beginn initialisieren wir $r = -\infty$. Im k-ten Iterationsschritt des L-Klassen-Enumerationsverfahrens wird ein neuer Punkt $\mathbf{x}^{(k)}$ aus einem vorherigen Punkt $\mathbf{x}^{(k-1)}$ konstruiert; der neue Punkt $\mathbf{x}^{(k)}$ liegt in einer L-Klasse $V \in \mathcal{S}/L$, die lexikographisch kleiner als der vorherige Punkt $\mathbf{x}^{(k-1)}$ ist, d. h. $\mathbf{x}^{(k-1)} \succ V$, und es gilt $f(\mathbf{x}^{(k)}) > r = f(\boldsymbol{\varsigma})$, wobei $\boldsymbol{\varsigma} \in \mathbb{Z}^n$ der zuletzt gefundene ganzzahlig zulässige Punkt vor Iteration k ist. Ist $\mathbf{x}^{(k)} \in \mathbb{Z}^n$, dann ist $\boldsymbol{\varsigma} := \mathbf{x}^{(k)}$.

Das LKE-Verfahren kann im Prinzip von jedem beliebigen Punkt $\mathbf{x}^{(0)} \in \mathbb{R}^n$ gestartet werden, wird sinnvollerweise aber, wie weiter unten ausgeführt, mit $\mathbf{x}^{(0)} = \text{lexmax}\mathcal{S}_0$ initialisiert. Die weiteren Punkte $\mathbf{x}^{(k)}$ werden als Lösungspunkte spezieller Probleme berechnet. Das Verfahren terminiert, wenn keine weitere L-Klasse V mehr zu bestimmen ist. Ist $\mathcal{S}_0 \neq \emptyset$ und $\mathbf{x}^{(0)} \succeq \mathcal{S}_0$, so ist der zuletzt gefundene Punkt $\boldsymbol{\varsigma} \in \mathbb{Z}^n$ die optimale Lösung des Optimierungsproblems.

Schritt S0: Zunächst wird das LP-Relaxierungsproblem gelöst, also

$$\max_{\mathbf{x}\in\mathcal{S}} f(\mathbf{x}) \quad .$$

Ist dieses Problem unzulässig, oder ist dessen optimale Lösung $\mathbf{x}^{\text{LP}}$ ganzzahlig zulässig, so → S4; andernfalls → S1.

Schritt S1: Bestimme $\mathbf{v} = \text{lexmax}\mathcal{S}$. Ist $\mathbf{v} \in \mathbb{Z}^n$, so wird $r = f(\mathbf{v})$ gesetzt und → S3; andernfalls → S2.

Schritt S2: Eingangsvektor $\mathbf{x}'$ – das ist der lexikographisch maximale Punkt der zuvor betrachteten Menge: In diesem Schritt wird zunächst der Index

$$p = \min\left\{j \mid x'_j \neq \lfloor x'_j \rfloor\,,\ j = 1, \ldots, n\right\}$$

bestimmt, der auf die erste Komponente des Vektors $\mathbf{x}'$ zeigt, die nicht ganzzahlig ist. Als nächstes wird ein Problem **P2** formuliert, dessen Lösung $\mathbf{x}^{(2)}_{\text{lex}} = \text{lexmax}\mathcal{S}_2$ das lexikographisch maximale Element der Menge

$$\mathcal{S}_2 = \{\mathbf{x} \in \mathcal{S} \mid\ (f(x) \geq r + \alpha) \wedge \left(x_p \leq \lfloor x'_p \rfloor\right) \wedge \left(x_j = x'_j\right), \quad j = 1, \ldots, p-1\}$$

wie in Korbut & Finkelstein (1971,[184]) beschriebenen lexikographischen dualen Simplexverfahren oder als Lösungspunkt der Sequenz von n linearen Problemen $\mathbf{P}_i$

$$\mathbf{P}_i: \quad \max_{\mathbf{x}\in\mathcal{M}_i} x_i$$

berechnet werden. Hierbei ist $\mathcal{M}_1 = \mathcal{M}$ und

$$\begin{aligned} \mathcal{M}_i &= \{\mathbf{x} \in \mathcal{M}_{i-1} | x_{i-1} = \bar{x}_{i-1}\} \quad , \quad i = 2, \ldots, n \\ &= \left\{\mathbf{x} \in \mathcal{M} | x_{i'} = \bar{x}_{i'}, i' = 1, \ldots, i-1\right\} \quad , \quad i = 2, \ldots, n \quad , \end{aligned}$$

wobei $\bar{\mathbf{x}}_i$ die optimale Lösung des Problems $\mathbf{P}_i$ bezeichnet; dieses Verfahren mag zwar aufwendig erscheinen, die Folge der LP-Probleme kann wegen der speziellen Gestalt der Zielfunktionen aber sehr schnell gelöst werden.

ist; der Parameter α entspricht einer geeigneten Minimalverbesserung; bei Problemen mit ganzzahliger Zielfunktion bietet sich $\alpha = 1$ an. Bei dem Versuch, $\mathbf{x}_{\text{lex}}^{(2)}$ zu bestimmen, können nun die folgenden Fälle auftreten:

2.1 Das Problem **P2** besitzt eine optimale Lösung $\mathbf{v}$. Hier sind wiederum zwei Fälle zu unterscheiden: Ist $\mathbf{v} \in \mathbb{Z}^n$, so setzen wir $r = f(\mathbf{v})$, $\mathbf{x}' = \mathbf{v}$ und $p = n + 1$ und gehen damit zur nächsten Iteration $\rightarrow$ S3; andernfalls $\rightarrow$ S2.

2.2 Das Problem **P2** besitzt keine Lösung, ist also unzulässig. Ist $p = 1 \rightarrow$ S4; ist $p > 1$, so wird die nächste Iteration eingeleitet $\rightarrow$ S3.

Schritt S3: Eingangsvektor $\mathbf{x}'$ (das ist die optimale Lösung des zuvor betrachteten LP-Problems): In diesem Schritt wird zuerst der maximale Index

$$q = \max\left\{j \leq p-1 \mid x'_j > 0\right\}$$

bestimmt, für den die Komponente des Eingangsvektors von Null verschieden ist. Gibt es keinen derartigen Index, gilt also $x_1 = x_2 = \ldots = x_{p-1} = 0$, so terminiert das Verfahren $\rightarrow$ S4. Existiert der Index, so wird ein Problem **P3** formuliert, dessen Lösung $\mathbf{x}_{\text{lex}}^{(3)} = \text{lexmax}\mathcal{S}_3$ das lexikographisch maximale Element der Menge

$$\mathcal{S}_3 = \{\mathbf{x} \in \mathcal{S} \mid \ (f(x) \geq r + \alpha) \wedge \left(x_q \leq x'_q - 1\right) \wedge \left(x_j = x'_j\right), \quad j = 1, \ldots, q-1\}$$

ist. Bei dem Versuch, $\mathbf{x}_{\text{lex}}^{(3)}$ zu berechnen, können die folgenden Fälle 3.1 und 3.2 auftreten:

3.1 Das Problem **P3** besitzt eine optimale Lösung $\mathbf{v}$. Hier sind nochmals zwei Fälle zu unterscheiden: Ist $\mathbf{v} \in \mathbb{Z}^n$, so setzen wir $r = f(\mathbf{v})$, $\mathbf{x}' = \mathbf{v}$ und $p = n + 1$ und gehen damit zur nächsten Iteration $\rightarrow$ S3; andernfalls $\rightarrow$ S2.

3.2 Das Problem **P3** besitzt keine Lösung, ist also unzulässig. Ist $q = 1 \rightarrow$ S4; ist $q > 1$, so wird die nächste Iteration mit $p = q$ eingeleitet $\rightarrow$ S3.

Schritt S4: Der Verfahren terminiert. Die beste gewonnene Lösung ist die optimale Lösung; hat man keine Lösung bestimmen können, so hat das Problem keine ganzzahligen Lösungen.

Die Verfahrensschritte 0 bis 3 können jederzeit durch zusätzliche Schnitte unterstützt werden. Das beschriebene Verfahren erzeugt eine Folge $\mathcal{F} = \{\mathbf{x}^{(k)}, k = 1, \ldots, K\}$ mit den folgenden Eigenschaften:

1. $\mathbf{x}^{(k)} \succ \mathbf{x}^{(k+1)}, k = 1, \ldots, K-1$.
2. Alle Punkte gehören zu verschiedenen L-Klassen.
3. Ist $\mathcal{S}_0 \neq \emptyset$, so enthält $\mathcal{F}$ eine Unterfolge $\mathcal{G} = \{\mathbf{x}^{(k_s)}\}$, $s = 1, \ldots, S$, die $\mathbf{z}_* \in \mathcal{G}$ enthält, und für die die folgende Monotonieeigenschaft gilt: Ist $k_{s+1} > k_s$, so gilt $f(\mathbf{x}^{(k_{s+1})}) > f(\mathbf{x}^{(k_s)})$ für alle $s = 1, \ldots, S-1$.

Bemerkenswert ist die Eigenschaft $K \leq |\mathcal{S}/L|$. Die Effizienz des Verfahrens hängt sehr davon ab, wie schnell eine gute Schranke r bestimmt wird, bzw. wie wirksam der Schnitt $f(\mathbf{x}) \geq r + \alpha$ ist.

Ein einfaches Beispiel: Der zulässige Bereich $\mathcal{S}$ sei durch die Eckpunkte

$$\left\{\left(\frac{1}{2},\frac{1}{2}\right),\left(\frac{1}{2},\frac{5}{2}\right),\left(1,\frac{3}{2}\right)\right\}$$

eines Dreieck (Abb. 4.7, links) definiert, das keine ganzzahligen Punkte enthält. Ein reines B&B-Verfahren benötigt 7 Knoten, um die Unzulässigkeit nachzuweisen. Das Problem hat zwei L-Klassen; in absteigender lexikographischer Ordnung sind dies

$$V_{01} = \left\{\left(1,\frac{3}{2}\right)\right\} \quad , \quad V_{02} = \left\{(x_1,x_2) \in \mathcal{S} \;\middle|\; \frac{1}{2} \le x_1 < 1\right\} \quad .$$

Die erste L-Klasse V_{01} ist der Punkt $(1,\frac{3}{2})$, der sich als Schnittpunkt des vertikalen Intervall $\{(x_1,x_2)|x_1 = 1, 1 < x_2 < 2\}$ mit dem Dreieck ergibt. Die zweite L-Klasse V_{02} ist die Schnittmenge des vertikaler Streifens $\{(x_1,x_2)|0 < x_1 < 1\}$ mit dem Dreieck. Nachfolgend sind die Schritte der L-Klassen-Enumeration für das Problem $\max_{(x_1,x_2)\in\mathcal{S}} x_1$ illustriert.

Schritt S0: LP-Relaxierung, $\mathbf{x}^{\mathrm{LP}} = (1,\frac{3}{2})$
Schritt S1: $v = \mathrm{lexmax}\mathcal{S} = (1,\frac{3}{2})$
Schritt S2: $\mathbf{x}' = (1,\frac{3}{2})$, $p = 2$,

$$\mathbf{x}^{(2)}_{\mathrm{lex}} = \mathrm{lexmax}\mathcal{S}_2 = \mathrm{lexmax}\{\mathbf{x} \in \mathcal{S} | x_1 = 1, x_2 \le 1\} \quad .$$

Offensichtlich ist $\mathcal{S}_2 = \emptyset$. Da $p > 1$, so wird die nächste Iteration eingeleitet $\rightarrow$ S3.
Schritt S3: $\mathbf{x}' = (1,\frac{3}{2})$, $q = 1$,

$$\mathbf{x}^{(3)}_{\mathrm{lex}} = \mathrm{lexmax}\mathcal{S}_3 = \mathrm{lexmax}\{\mathbf{x} \in \mathcal{S} | x_1 \le 0\} \quad .$$

Auch hier stellen wir fest, dass es wegen $\mathcal{S}_3 = \emptyset$ keine Lösung gibt. Wegen $q = 1$ wird mit Schritt S4 fortgefahren; da kein ganzzahlig zulässiger Punkt bestimmt wurde, ist die Unzulässigkeit des Problems festgestellt.

Betrachten wir zur Erläuterung noch einmal das in Abschnitt 4.3.1 beschriebene, von Kühen und Schweinen handelnde Beispiel mit den Beziehungen (4.3.16) bis (4.3.19)

$$\max_{\kappa,\sigma} z \quad , \quad z = 3\kappa + 2\sigma$$

unter den Nebenbedingungen

$$0 \le \kappa, \sigma \le 2 \quad ; \quad \kappa, \sigma \in \mathbb{N}_0$$

und

$$\kappa + \sigma \le 3.5 \quad ;$$

der zulässige Bereich der LP-Relaxierung und die ganzzahligen Lösungspunkte sind noch einmal in Abb. 4.7 (rechts) gezeigt. Dieses Problem hat 16 L-Klassen. Sie lauten in lexikographisch absteigender Ordnung:

$V_{01} = \{(\kappa,\sigma)|\kappa = 2, 1 < \sigma \le 1.5\}$
$V_{02} = \{(2,1)\}$ – dies ist die optimale Lösung –
$V_{03} = \{(\kappa,\sigma)|\kappa = 2, 0 < \sigma < 1\}$
$V_{04} = \{(2,0)\}$
$V_{05} = \{(\kappa,\sigma)|1 < \kappa < 2, 0 \le \sigma \le \min\{2, 3.5 - \kappa\}\}$
$V_{06} = \{(1,2)\}$
$V_{07} = \{(\kappa,\sigma)|\kappa = 1, 1 < \sigma < 2\}$
$V_{08} = \{(1,1)\}$
$V_{09} = \{(\kappa,\sigma)|\kappa = 1, 0 < \sigma < 1\}$
$V_{10} = \{(1,0)\}$
$V_{11} = \{(\kappa,\sigma)|0 < \kappa < 1, 0 \le \sigma \le 2\}$
$V_{12} = \{(0,2)\}$
$V_{13} = \{(\kappa,\sigma)|\kappa = 0, 1 < \sigma < 2\}$
$V_{14} = \{(0,1)\}$
$V_{15} = \{(\kappa,\sigma)|\kappa = 0, 0 < \sigma < 1\}$
$V_{16} = \{(0,0)\}$

In dieser Auflistung erkennt man die möglichen topologischen Formen der L-Klassen: Ganzzahlige Punkte (V_{02}, V_{04}, ..., V_{16}), vertikale Intervalle (V_{01}, V_{03}, V_{07}, V_{09}, V_{11}, V_{13}, V_{15}) und vertikale Streifen (V_{05} und V_{11}).

In der Initialisierung liefert die LP-Relaxierung $(\kappa, \sigma) = (x_1, x_2) = (2, 1.5)$. In Schritt 1 entspricht dies auch schon dem lexikographisch maximalen Punkt $v = (2, 1.5)$; dieser Punkt ist nicht ganzzahlig zulässig und deshalb erfolgt eine Verzweigung zum Schritt 2. Im Schritt 2 wird mit Hilfe des Eingangsvektors $(2, 1.5)$ der Index $p = 2$ identifiziert; somit ist der lexikographisch maximalen Punkt der durch $x_1 = 2$ und $x_2 \leq 1$ beschriebenen Menge $\mathcal{S}_2 = \mathcal{S} \cap \{(x_1, x_2) | x_1 = 2, x_2 \leq 1\}$ zu bestimmen. Dieser lautet $\mathbf{x}_{\text{lex}}^{(2)} = (2, 1)$; er ist ganzzahlig zulässig. Somit folgt $r = 8$, $\mathbf{x}' = (2, 1)$ und mit $p = 3$ die Initialisierung der nächsten Iteration in Schritt 3. Hier erhält man $q = 2$ und muss den lexikographisch maximalen Punkt der Menge $\mathcal{S}_3 = \mathcal{S} \cap \{(x_1, x_2) | f = 3x_1 + 2x_2 \geq 8 + \alpha, x_1 = 2, x_2 \leq 0\}$ bestimmen; die Menge $\mathcal{S}_3$ ist jedoch für jede beliebige Minimalverbesserung $\alpha > 0$ leer und das Problem somit unzulässig. Wegen $q > 1$ folgt mit $p = q = 2$ eine weitere Iteration in Schritt 3. Diesmal ist $q = 1$ und somit $\mathcal{S}_3 = \mathcal{S} \cap \{(x_1, x_2) | f = 3x_1 + 2x_2 \geq 8 + \alpha, x_1 \leq 1\}$; auch diese Menge $\mathcal{S}_3$ ist unzulässig und daher folgt Verzweigung zu Schritt 4. Die einzig bestimmte ganzzahlige Lösung (2,1) ist somit die optimale Lösung. Der Rechenaufwand der L-Klassen-Enumerierung bestand in diesem Beispiel also in der in der Lösung eines LP-Problems, nämlich der LP-Relaxierung, und der Bestimmung der lexikographisch maximalen Punkte von drei Mengen. Interessant ist bei diesem Verfahren, dass nur *ein* Problem mit der ursprünglichen Zielfunktion zu lösen ist; in anderen zu lösenden Problemen kommt es nur auf die Nebenbedingungen an.

Als weiteres Beispiel betrachten wir das schon bekannte Rucksackproblem

$$\max z \quad , \quad z := \sum_{i=1}^{n} V_i \sigma_i \quad , \quad u.d.N. \quad \sum_{i=1}^{n} W_i \sigma_i \leq 102 \quad , \quad \sigma_i \in \{0, 1\} \quad , \quad i = 1, \ldots, n$$

aus Abschnitt 2.2.7 mit den Daten

i	1	2	3	4	5	6	7	8
V_i	15	100	90	60	40	15	10	1
W_i	2	20	20	30	40	30	60	10

;

die LP-Relaxierung im Schritt S0 hat die Lösung

$$\mathbf{x}^{\text{LP}} = (1, 1, 1, 1, \frac{3}{4}, 0, 0, 0) \quad , \quad f(\mathbf{x}) = 295 \quad .$$

Schritt 1: $v = \text{lexmax}\mathcal{S} = (1, 1, 1, 1, \frac{3}{4}, 0, 0, 0)$
Schritt 2: $\mathbf{x}' = (1, 1, 1, 1, \frac{3}{4}, 0, 0, 0)$, $p = 5$,

$$\mathbf{x}_{\text{lex}}^{(2)} = \text{lexmax}\mathcal{S}_2 = \text{lexmax}\{\mathbf{x} \in \mathcal{S} | x_1 = x_2 = x_3 = x_4 = 1, x_5 \leq 0\} = (1, 1, 1, 1, 0, 1, 0, 0) \quad .$$

Dieser Punkt ist ganzzahlig zulässig, und daher $r = 280$, $\mathbf{x}' = \mathbf{x}_{\text{lex}}^{(2)}$, $p = 9$ und Fortsetzung mit Schritt 3.

Schritt 3, Iteration 1: Mit $\mathbf{x}' = (1, 1, 1, 1, 0, 1, 0, 0)$, $q = 6$ erweist sich das Problem der Bestimmung von

$$\mathbf{x}_{\text{lex}}^{(3)} = \text{lexmax}\mathcal{S}_3 = \text{lexmax}\{\mathbf{x} \in \mathcal{S} | f(x) \geq 280 + \alpha, x_1 = x_2 = x_3 = x_4 = 1, x_5 = 0, x_6 \leq 0\}$$

als unzulässig. Daher geht man jetzt noch einmal zurück zum Schritt 3 und versucht, da $q > 1$, nun für $p = q$ – daraus resultiert der neue Wert $q = 4$ – den lexikographisch maximalen Punkt im

Schritt 3, Iteration 2: $\mathbf{x}' = (1, 1, 1, 1, 0, 1, 0, 0)$, $q = 4$

$$\mathbf{x}_{\text{lex}}^{(3)} = \text{lexmax}\mathcal{S}_3 = \text{lexmax}\{\mathbf{x} \in \mathcal{S} | f(x) \geq 280 + \alpha, x_1 = x_2 = x_3 = 1, x_4 \leq 0\}$$

zu bestimmen. Da auch diese Menge $\mathcal{S}_3$ leer ist, wiederholt sich die Prozedur:

Schritt 3, Iteration 3: $\mathbf{x}' = (1, 1, 1, 1, 0, 1, 0, 0)$, $q = 3$

$$\mathbf{x}_{\text{lex}}^{(3)} = \text{lexmax}\mathcal{S}_3 = \text{lexmax}\{\mathbf{x} \in \mathcal{S} | f(x) \geq 280 + \alpha, x_1 = x_2 = 1, x_3 \leq 0\} \quad .$$

Dies geht mit gleichem Resultat so weiter

Schritt 3, Iteration 4: $\mathbf{x}' = (1, 1, 1, 1, 0, 1, 0, 0)$, $q = 2$

$$\mathbf{x}_{\text{lex}}^{(3)} = \text{lexmax}\mathcal{S}_3 = \text{lexmax}\{\mathbf{x} \in \mathcal{S} | f(x) \geq 280 + \alpha, x_1 = 1, x_2 \leq 0\}$$

bis schließlich auch im

Schritt 3, Iteration 5: $\mathbf{x}' = (1, 1, 1, 1, 0, 1, 0, 0)$, $q = 1$

$$\mathbf{x}_{\text{lex}}^{(3)} = \text{lexmax}\mathcal{S}_3 = \text{lexmax}\{\mathbf{x} \in \mathcal{S} | f(x) \geq 280 + \alpha, x_1 \leq 0\}$$

$\mathcal{S}_3 = \emptyset$ festgestellt wird. Da nun $q = 1$ ist, wird im Schritt 4 fortgefahren und der Punkt $(1, 1, 1, 1, 0, 1, 0, 0)$ als optimale Lösung ausgewiesen.

Die Indizierung bzw. der Aufbau des Vektors $\mathbf{x}$ bei der L-Klassen-Enumeration entspricht in einem gewissen Sinne der Priotisierung der ganzzahligen Variablen in B&B-Verfahren. Bemerkenswert bei der L-Klassen-Enumeration ist auch die Tatsache, dass das ursprüngliche Problem inklusive Zielfunktion nur einmal gelöst werden muss; danach werden nur lexikographisch maximale Punkte von Polyedern bestimmt.

4.4 Nichtlineare, kontinuierliche Optimierung

Die nichtlineare, kontinuierliche Optimierung auch nur ansatzweise vollständig darzustellen, würde über den Rahmen dieses Buches hinausgehen. Daher werden nur einige wesentliche Aspekte und Ideen skizziert sowie einige Grundlagen der nichtlinearen Optimierung zusammengestellt; Leser mit weitergehendem Interesse seien auf Spelluci (1993,[265]) und Gill *et al.* (1981,[104]) verwiesen. Die speziellen, meist in der Mineralölindustrie verwendeten Verfahren der Rekursion, der Sequentiellen Linearen Programmierung sowie der distributiven Rekursion werden im Abschnitt 9.1 gesondert behandelt.

4.4.1 Einige Grundlagen zur unbeschränkten Optimierung

Sei $f : X = \mathbb{R}^n \to \mathbb{R}, \mathbf{x} \to f(\mathbf{x})$ eine stetige, reellwertige Funktion. Das Problem

$$\min_{\mathbf{x} \in X}\{f(\mathbf{x})\} \quad \Longleftrightarrow \quad f_* := \min\{f(\mathbf{x}) \mid \mathbf{x} \in X\} \quad , \tag{4.4.27}$$

heißt *unbeschränktes Optimierungsproblem*. Ein Vektor $\mathbf{x}_*$, der bei Auswertung der Funktion $f(\mathbf{x})$ den Skalar f_* ergibt, d. h. $f_* = f(\mathbf{x}_*)$, heißt Minimum oder minimierender Punkt $\mathbf{x}_*$ und kann formal durch

$$\mathbf{x}_* = \arg\min\{\min_{\mathbf{x}\in X} f(\mathbf{x})\} := \{\mathbf{x} \mid f(\mathbf{x}) \leq f(\mathbf{x}'),\ \forall\ \mathbf{x}' \in X\} \tag{4.4.28}$$

beschrieben werden. Maximierungsprobleme lassen sich durch die Beziehung

$$\max_{\mathbf{x}\in X}\{f(\mathbf{x})\} = -\min_{\mathbf{x}\in X}\{-f(\mathbf{x})\}$$

auf Minimierungsprobleme zurückführen; daher können wir je nach Bedarf wählen, ob wir mit einem Maximierungs- oder Minimierungsproblem arbeiten wollen. Eine typische Schwierigkeit im Umfeld der nichtlinearen nichtkonvexen Optimierung besteht darin, dass es meist nur möglich ist, lokale Optima zu bestimmen, es aber nur selten möglich ist nachzuweisen, dass ein globales Optimum bestimmt wurde. In einfachen Worten ausgedrückt ist ein globales Optimum die beste Lösung für alle zulässigen Punkte, während ein lokales Optimum $\mathbf{x}_*$ nur die beste Lösung in einer Nachbarschaft dieses Punktes $\mathbf{x}_*$ ist. Die formale Definition dafür lautet: In einem gegebenem Minimierungsproblem heißt ein Punkt $\mathbf{x}_* \in X$ *lokales Minimum bezüglich einer Umgebung* $U_{\mathbf{x}_*}$ von $\mathbf{x}_*$ (oder einfach *lokales Minimum*), wenn gilt

$$f(\mathbf{x}_*) \leq f(\mathbf{x}), \quad \forall\ \mathbf{x} \in U_{\mathbf{x}_*} \quad .$$

Gilt $f(\mathbf{x}_*) \leq f(\mathbf{x})$ für alle $\mathbf{x}_* \in X$, so heißt $\mathbf{x}_*$ *globales Minimum*.

Erste Lösungsansätze waren Suchverfahren, die lediglich die Funktionswerte $f(\mathbf{x})$ für verschiedene Punkte $\mathbf{x}$ auswerteten. Eines der meist verwendeten Verfahren ist die Simplex-Methode, auch als Nelder-Mead-Methode bekannt, [Spendley *et al.* (1962,[266]), Nelder & Mead (1965,[220])], die nicht mit dem Simplexverfahren in Kapitel 4 verwechselt werden sollte. In dieser Gruppe findet sich auch die *alternating variables method*, in denen in jeder Iteration k ($k = 1, 2, ..., K$) lediglich die Variable x_k variiert wird mit dem Ziel, den Wert der Zielfunktion zu verringern; alle übrigen Variablen bleiben auf ihre Werte fixiert. Beide Verfahren sind leicht zu implementieren. Sie können auch auf Probleme mit nichtglatten Funktionen angewendet werden. Zwar zeigen sie gewöhnlich ein recht langsames Konvergenzverhalten, aber in einigen Situationen, in denen ableitungsbasierte Verfahren Probleme haben, können sie sehr nützlich sein. Dies trifft insbesondere dann zu, wenn ableitungsbasierte Verfahren empfindlich von den Startwerten abhängen; in diesem Fall kann die approximative Lösung des Optimierungsproblems, die mit einem ableitungsfreien Verfahren ermittelt wurde, als Startwert für das ableitungsbasierte Verfahren verwendet werden.

Ableitungsbasierte Verfahren benötigen Ableitungen erster, vielfach auch zweiter Ordnung; daher sei nun stets angenommen, dass $f(\mathbf{x})$ die erforderlichen Glattheitseigenschaften erfülle, so dass die Ableitungen stetig sind. Im einzelnen werden benötigt:

- der Gradient der skalaren, reellwertigen differenzierbaren Funktion $f(\mathbf{x})$ des Vektors $\mathbf{x}$; $f : X = \mathbb{R}^n \to \mathbb{R}$, $\mathbf{x} \to f(\mathbf{x})$

$$\bigtriangledown f(\mathbf{x}) := \left(\frac{\partial}{\partial x_1} f(\mathbf{x}), \ldots, \frac{\partial}{\partial x_n} f(\mathbf{x})\right) \in \mathbb{R}^n \quad , \tag{4.4.29}$$

- die Hesse-Matrix[15] H der Funktion $f(\mathbf{x})$

$$\mathsf{H}(\mathbf{x}) \equiv \bigtriangledown^2 f(\mathbf{x}) := \frac{\partial}{\partial x_i}\left(\frac{\partial}{\partial x_j} f(\mathbf{x})\right) = \left(\frac{\partial^2}{\partial x_i \partial x_j} f(\mathbf{x})\right) \in \mathcal{M}(n,n) \quad , \qquad (4.4.30)$$

- die Jacobi-Matrix[16] J der vektorwertigen Funktion $\mathbf{g}(\mathbf{x}) = (g_1(\mathbf{x}), \ldots, g_m(\mathbf{x}))^{\mathrm{T}}$

$$\mathsf{J}(\mathbf{x}) \equiv \bigtriangledown \mathbf{g}(\mathbf{x}) := (\bigtriangledown g_1(\mathbf{x}), \ldots, \bigtriangledown g_m(\mathbf{x})) = \left(\frac{\partial}{\partial x_j} g_i(\mathbf{x})\right) \in \mathcal{M}(m,n) \quad . \qquad (4.4.31)$$

Die Hesse-Matrix der skalaren Funktion $f(\mathbf{x})$ ist die Jacobi-Matrix des Gradienten $\bigtriangledown f(\mathbf{x})$. Wie im eindimensionalen Fall genügt ein minimierender Punkt $\mathbf{x}_*$ der Bedingung

$$\bigtriangledown f(\mathbf{x}_*) = \mathbf{0} \quad , \quad \bigtriangledown f(\mathbf{x}_*) \in \mathbb{R}^n \quad . \qquad (4.4.32)$$

Mit Hilfe einer Taylorreihenentwicklung der Funktion $f(\mathbf{x})$ um den Punkt $\mathbf{x}_*$ lässt sich das folgende Theorem über die hinreichenden Bedingung beweisen:

Theorem 1 *Hinreichend dafür, dass $\mathbf{x}_*$ ein lokal minimierender Punkt ist, ist dass (4.4.32) erfüllt ist und dass die Hesse-Matrix $\mathsf{H}_* := \mathsf{H}(\mathbf{x}_*)$ positiv definit ist, d. h.*

$$\mathbf{s}^{\mathrm{T}} \mathsf{H}_* \mathbf{s} > 0 \quad , \quad \forall\ \mathbf{s} \neq 0 \quad , \quad \mathbf{s} \in \mathbb{R}^n \quad .$$

Hierbei bezeichnet $\mathbf{s}$ einen beliebigen von Null verschiedenen Vektor. Ein grundlegender Ansatz zur numerischen Lösung des Minimierungsproblems (4.4.27) ist das *Liniensuchverfahren* [engl.: *line search algorithm*]. Aus einer bekannten Lösung $\mathbf{x}_k$ im k-ten Iterationsschritt lässt sich mit Hilfe der folgenden Schritte $\mathbf{x}_{k+1}$ berechnen:

- Bestimmung einer Suchrichtung $\mathbf{s}_k$;
- Bestimmung des Minimums der Hilfsfunktion[17] $f(\mathbf{x}_k + \alpha_k \mathbf{s}_k)$ mit Hilfe einer *Liniensuche*, wobei $\alpha_k > 0$ ein passender Dämpfungsfaktor ist; und
- Berechnung der Lösung $\mathbf{x}_{k+1} := \mathbf{x}_k + \alpha_k \mathbf{s}_k$ für den nächsten Iterationsschritt.

Algorithmen zur Lösung von (4.4.27) unterscheiden sich in der Art und Weise, wie die Suchrichtung $\mathbf{s}_k$ berechnet wird. *Gedämpfte Verfahren* verwenden den Dämpfungsfaktor α_k; die meisten berechnen ihn mit Hilfe eines Liniensuchverfahren. *Ungedämpfte Verfahren* setzen $\alpha_k = 1$. Wird $f(\mathbf{x}_k + \alpha_k \mathbf{s}_k)$ exakt minimiert, dann ist der Gradient $\bigtriangledown f(\mathbf{x}_{k+1})$ im neuen Punkt $\mathbf{x}_{k+1}$ senkrecht zur Suchrichtung $\mathbf{s}_k$, d. h. $\mathbf{s}_k^{\mathrm{T}} \bigtriangledown f(\mathbf{x}_{k+1}) = 0$. Dies folgt aus der notwendigen Bedingung

$$0 = \frac{\mathrm{d}}{\mathrm{d}\alpha_k} f(\mathbf{x}_k + \alpha_k \mathbf{s}_k) = \left[\bigtriangledown f(\mathbf{x}_k + \alpha_k \mathbf{s}_k)\right]^{\mathrm{T}} \mathbf{s}_k \quad .$$

[15] So benannt nach dem deutschen Mathematiker Ludwig Otto Hesse (1811–1874). $\mathcal{M}(m,n)$ bezeichnet die Menge aller Matrizen mit m Zeilen und n Spalten.

[16] So benannt nach dem schweizerischen Mathematiker Carl Gustav Jacob Jacobi (1804–1851).

[17] In der Regel wird nicht das exakte Minimum von $f(\mathbf{x}_k + \alpha \mathbf{s}_k)$ bestimmt. Eine mögliche Heuristik ist f für $\alpha_m = 2^{-m}$ für $m = 0, 1, 2, \ldots,$ auszuwerten und die Liniensuche zu beenden, wenn $f(\mathbf{x}_k + \alpha_m \mathbf{s}_k) \leq f(\mathbf{x}_k)$.

Auf Basis dieses grundlegenden Konzeptes lassen sich verschiedene Optimierungsverfahren zur Lösung von (4.4.27) klassifizieren. *Abstiegsverfahren* [engl.: *descent methods*] sind Liniensuchverfahren, in denen die Suchrichtung $\mathbf{s}_k$ der Eigenschaft

$$\bigtriangledown f(\mathbf{x}_k)^{\mathrm{T}}\mathbf{s}_k < 0 \tag{4.4.33}$$

genügt. Die *Methode des steilsten Abstiegs* [engl.: *steepest descent method*] verwendet $\mathbf{s}_k = -\bigtriangledown f(\mathbf{x}_k)$; die Bedingung (4.4.33) ist erfüllt. Der Gradient kann hierbei analytisch berechnet oder numerisch mit Hilfe finiter Differenzen approximiert werden. Hierbei werden manchmal die asymmetrischen Differenzen

$$\bigtriangledown_i f(\mathbf{x}) = \frac{\partial f}{\partial x_i}(\mathbf{x}) \approx \frac{f(\mathbf{x}+h\mathbf{e}_i) - f(\mathbf{x})}{h} + f''(\mathbf{x})h$$

oder die symmetrischen Differenzen

$$\bigtriangledown_i f(\mathbf{x}) = \frac{\partial f}{\partial x_i}(\mathbf{x}) \approx \frac{f(\mathbf{x}+\frac{1}{2}h\mathbf{e}_i) - f(\mathbf{x}-\frac{1}{2}h\mathbf{e}_i)}{h} + \tfrac{1}{24}f'''(\mathbf{x})h^2$$

verwendet, wobei $\mathbf{e}_i$ den Einheitsvektor entlang der i-ten Koordinatenachse bezeichnet. Vom numerischen Standpunkt sind symmetrische Differenzen vorzuziehen, da sie genauer sind und einen Approximationsfehler[18] haben, der von der Ordnung h^2 ist.

Eine andere Möglichkeit, die Suchrichtung $\mathbf{s}_k$ zu berechnen, besteht darin, die Funktion $f(\mathbf{x})$ um $\mathbf{x}_k$ in eine Taylorreihe bis zur zweiten Ordnung zu entwickeln. Dies entspricht einer quadratischen Approximation der Funktion $f(\mathbf{x})$, d. h. $\mathbf{x} = \mathbf{x}_k+\mathbf{s}_k$, $f(\mathbf{x}) = f(\mathbf{x}_k+\mathbf{s}_k)$ und

$$f(\mathbf{x}_k+\mathbf{s}_k) \approx f(\mathbf{x}_k) + \bigtriangledown f(\mathbf{x}_k)^{\mathrm{T}}\mathbf{s}_k+\tfrac{1}{2}\mathbf{s}_k^{\mathrm{T}}\mathsf{H}_k\mathbf{s}_k \quad . \tag{4.4.34}$$

Sind sowohl der Gradient $\bigtriangledown f(\mathbf{x})$ und die Hesse-Matrix H analytisch gegeben, so lässt sich aus (4.4.34) das klassische *Newton-Verfahren* ableiten. Wendet man die notwendigen Bedingungen (4.4.32) auf (4.4.34) an, so ergibt sich

$$\mathsf{H}_k\mathbf{s}_k = -\bigtriangledown f(\mathbf{x}_k) \quad , \tag{4.4.35}$$

woraus sich schließlich $\mathbf{s}_k$ gemäß

$$\mathbf{s}_k = -\mathsf{H}_k^{-1}\bigtriangledown f(\mathbf{x}_k) \tag{4.4.36}$$

berechnen lässt. Wie z. B. Bock (1987,[41]) gezeigt hat, konvergiert das gedämpfte Newton-Verfahren (Newton-Verfahren mit Liniensuche) global, d. h. von jedem Anfangspunkt aus, wenn H_k positiv definit. Der Fall, in dem H_k nicht positiv-definit ist, wird nur nach einer Modifikation der Hesse-Matrix behandelt.

[18] Hierbei stellt sich die Frage, wie die Größe der Inkremente h gewählt werden soll. Wie von Press *et al.* (1992,[237]) verständlich ausgeführt, hängt die optimale Wahl dieser Größe von der Krümmung, d. h. den zweiten Ableitungen von $f(x)$, ab. Da die zweite Ableitung in den meisten Fällen jedoch unbekannt ist, kann bestenfalls die in Press *et al.* (1992, p. 180)[237] gegebene heuristische Näherung $h \approx 2\epsilon_f^{1/3}x$ verwendet werden, wobei ϵ_f die relative Genauigkeit bezeichnet, mit der $f(x)$ berechnet wird. Für nicht zu komplizierte Funktionen entspricht dies etwa der Maschinengenauigkeit, d. h. $\epsilon_f \approx \epsilon_m$.

In den meisten praktischen Problemen ist die Hesse-Matrix H aber nicht verfügbar. Ist jedoch $\bigtriangledown f(\mathbf{x})$ analytisch bekannt, oder kann numerisch mit Hilfe asymmetrischer oder symmetrischer finiter Differenzen mit hinreichender Genauigkeit berechnet werden, so können *Quasi-Newton-Verfahren,* siehe z. B. Werner (1992,[288] , Abschnitt 7.3), angewendet werden. Hierbei existieren zwei Arten von Quasi-Newton-Verfahren. Entweder wird die auf finite Differenzen beruhende Approximation $\bar{\mathsf{H}}_k$ von H_k symmetrisiert und H_k in (4.4.36) wird durch

$$\tfrac{1}{2}\left(\bar{\mathsf{H}}_k + \bar{\mathsf{H}}_k^{\mathrm{T}}\right) \quad ,$$

ersetzt, oder die inverse Matrix H_k^{-1} wird durch eine symmetrische, positiv-definite Matrix $\tilde{\mathsf{H}}_k$ approximiert; die letztgenannte Alternative ist numerisch effizienter. Die Anfangsmatrix $\tilde{\mathsf{H}}_0$ kann dabei eine beliebige positiv-definite Matrix sein. In Ermangelung besserer Startwerte wird häufig die Einheitsmatrix $\mathbb{1}$, oder, wenn spezifische Informationen verfügbar sind, eine Diagonalmatrix zur Initialisierung von $\tilde{\mathsf{H}}_0$ verwendet. Die Methoden werden manchmal auch Variable-Metrik-Verfahren [engl.: *variable metric methods*] genannt; hierzu sei z. B. auf Press *et al.* (1992,[237], S.418-422) verwiesen. Die Effizienz dieses Quasi-Newton-Verfahren hängt von der Güte des Update-Verfahren ab, mit dessen Hilfe $\tilde{\mathsf{H}}_{k+1}$ aus berechnet wird $\tilde{\mathsf{H}}_k$; einige Update-Formeln operieren direkt auf der inverse Matrix H_k^{-1} und generieren H_{k+1}^{-1}.

Schließlich lassen quadratische Optimierungsprobleme wie (4.4.34) sich mit konjugierten Richtungs-Verfahren [engl.: *conjugate direction methods*; CDMs] lösen, da diese in solchen Fälle nach höchstens n Schritten oder Iterationen terminieren. CDMs basieren auf dem Konzept der Konjugation [engl.: *conjugacy*] einer Menge von von Null verschiedenen Vektoren $\{\mathbf{s}_1, \mathbf{s}_2, ...\mathbf{s}_n\}$ bezüglich einer gegebenen positiv-definiten Matrix H, d. h.

$$\mathbf{s}_i^{\mathrm{T}} \mathsf{H} \mathbf{s}_j = 0 \quad , \quad \forall\ i \neq j \tag{4.4.37}$$

und erzeugen diese Richtungen $\{\mathbf{s}_1, \mathbf{s}_2, ...\mathbf{s}_n\}$ bei Anwendung auf eine quadratische Funktion mit Hesse-Matrix H.

Ein besonderer Fall, dessen Struktur ausgenutzt werden kann, liegt vor, wenn $f(\mathbf{x})$ aus einer Summe von Quadraten aufgebaut ist; dies ist z. B. bei auf der *Methode der Kleinsten Quadrate* beruhenden Ausgleichsverfahren gegeben. In diesem Fall lässt sich die Hesse-Matrix H direkt aus $f(\mathbf{x})$ und $\bigtriangledown f(\mathbf{x}_k)$ berechnen.

4.4.2 Beschränkte Optimierung - Grundlagen und einige Theoreme

Ähnlich wie die lineare Programmierung wird aus historischen Gründen die beschränkte Optimierung [engl.: *constrained optimization*] auch nichtlineare Programmierung [engl.: *nonlinear programming*; NLP] genannt. Ein allgemeines NLP-Problem mit n Variablen, n_2 Gleichungen und n_3 Ungleichungen ist somit das **Problem NLP**

$$\begin{array}{lll} \text{Minimiere:} & f(\mathbf{x}), & \mathbf{x} \in \mathbb{R}^n, \\ \textit{u.d.N.:} & \mathbf{F}_2(\mathbf{x}) = 0, & \mathbf{F}_2 : \mathbb{R}^n \to \mathbb{R}^{n_2} \quad , \\ & \mathbf{F}_3(\mathbf{x}) \geq 0, & \mathbf{F}_3 : \mathbb{R}^n \to \mathbb{R}^{n_3} \quad . \end{array} \tag{4.4.38}$$

Die Funktionen $f(\mathbf{x})$, $\mathbf{F}_2(\mathbf{x})$ und $\mathbf{F}_3(\mathbf{x})$ seien hierbei als stetig differenzierbar über den Vektorraum $\mathbb{R}^n$ vorausgesetzt. Die Vektorungleichung $\mathbf{F}_3(\mathbf{x}) \geq 0$ stehe abkürzend für die n_3 Ungleichungen $F_{3k}(\mathbf{x}) \geq 0$, $1 \leq k \leq n_3$. Die Menge aller zulässigen Punkte

$$\mathcal{S} := \{\mathbf{x} \mid \mathbf{F}_2(\mathbf{x}) = 0 \wedge\ \mathbf{F}_3(\mathbf{x}) \geq 0\} \tag{4.4.39}$$

heißt zulässiger Bereich [engl.: *feasible region*] oder zulässige Menge $\mathcal{S}$. Die Menge der aktiven Nebenbedingungen im Punkt $\mathbf{x}$ bezüglich $\mathcal{S}$ wird durch die Indexmenge

$$\mathcal{I}(\mathbf{x}) := \{i \mid F_{3i}(\mathbf{x}) = 0, \;\; i = 1, \ldots, n_3\} \quad ,$$

charakterisiert, die manchmal auch Menge der aktiven Ungleichungen [engl.: *active set*] genannt wird. In den frühen 1950er Jahren erweiterten Kuhn & Tucker (1951,[188]) die Theorie der Lagrange-Multiplikatoren, die bis dahin zur Lösung gleichungsbeschränkter Optimierungsprobleme verwendete, dahingehend, dass nun sowohl Gleichungen als auch Ungleichungen einbezogen werden konnten. Die Kuhn-Tucker-Theorie beruht auf der folgenden Definition einer Lagrange-Funktion

$$L(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) := f(\mathbf{x}) - \boldsymbol{\lambda}^{\mathrm{T}}\mathbf{F}_2(\mathbf{x}) - \boldsymbol{\mu}^{\mathrm{T}}\mathbf{F}_3(\mathbf{x}) \quad , \tag{4.4.40}$$

die die Zielfunktion $f(\mathbf{x})$ mit den Nebenbedingungen $\mathbf{F}_2(\mathbf{x})$ und $\mathbf{F}_3(\mathbf{x})$ verbindet. Die Vektorvariablen $\boldsymbol{\lambda} \in \mathbb{R}^m$ und $\boldsymbol{\mu} \in \mathbb{R}^l$ heißen Lagrange-Multiplikatoren; sie werden dem Problem **NLP** als zusätzliche Unbekannte hinzugefügt.

Nachstehend stellen wir einige Theoreme und Resultate der NLP-Theorie zusammen. Die notwendigen Bedingungen für die Existenz eines lokalen Optimums lauten, wie von Kuhn & Tucker (1951,[188]) bewiesen und z. B. in Ravindran *et al.* (1987,[240]) in der heute üblichen Weise mit den Jacobi-Matrix J_2 und J_3 von $\mathbf{F}_2$ und $\mathbf{F}_3$ beschrieben:

Theorem 2 *Ist* $\mathbf{x}_*$ *eine Lösung des NLP-Problems und sind die Funktionen* $f(\mathbf{x})$, $\mathbf{F}_2(\mathbf{x})$ *und* $\mathbf{F}_3(\mathbf{x})$ *stetig-differenzierbar, dann existieren Vektoren* μ_* *und* λ_* *derartig, dass* $\mathbf{x}_*$, μ_* *und* λ_* *den folgenden Bedingungen genügen:*

$$\begin{aligned} \mathbf{F}_2(\mathbf{x}) &= 0 \quad , & (4.4.41)\\ \mathbf{F}_3(\mathbf{x}) &\geq 0 \quad ,\\ \boldsymbol{\mu}^{\mathrm{T}}\mathbf{F}_3(x) &= 0 \quad ,\\ \boldsymbol{\mu} &\geq 0 \quad ,\\ \bigtriangledown f(\mathbf{x}) - \boldsymbol{\lambda}^{\mathrm{T}}\mathsf{J}_2(\mathbf{x}) - \boldsymbol{\mu}^{\mathrm{T}}\mathsf{J}_3(\mathbf{x}) &= 0 \quad . & (4.4.42) \end{aligned}$$

Beweise finden sich in Collatz & Wetterling (1971,[52]), oder Fletcher (1987,[87]). Die Gleichungen (4.4.41)-(4.4.42) werden auch (*Karush*[19] –) *Kuhn-Tucker-Bedingungen*, *KKT-Bedingungen* genannt, oder schlicht Bedingungen erster Ordnung. Ein Punkt $(\mathbf{x}_*, \mu_*, \lambda_*)$, der diesen Bedingungen genügt, heißt *Karush-Kuhn-Tucker-Punkt* oder kurz *KKT-Punkt*.

Darüber hinaus müssen die Funktionen $f(\mathbf{x})$, $\mathbf{F}_2(\mathbf{x})$ und $\mathbf{F}_3(\mathbf{x})$ wie von Kuhn und Tucker spezifiziert noch bestimmten *Regularitätsbedingungen* [engl.: *constraint qualification*] genügen, um bestimmte irreguläre Fälle auszuschließen. Alternative Formen der Regularitätsbedingungen werden z. B. Bomze & Grossmann (1993,[43]) oder von Gill *et al.* (1981) diskutiert. Eine sehr allgemeine Formulierung findet sich bei Bock (1987).

[19] Es wurde erst später festgestellt, dass Karush (1939,[174]) bereits im Jahre 1939 das gleiche Ergebnis in seiner Masterarbeit an der Universität Chicago bewiesen hatte. In seinem Überblicksartikel gibt Kuhn (1976,[187]) einen historischen Überblick über die ungleichungsbeschränkte Optimierung.

Bezeichne $\mathcal{I}(\mathbf{x}')$ die Menge der aktiven Ungleichungen [engl.: *active inequality constraints*] im Punkt $\mathbf{x}'$. Sei $\tilde{\mathbf{F}}_3$ die Funktion, die aus allen Funktionen F_{3i} zusammengesetzt ist, für die gilt $i \in \mathcal{I}(\mathbf{x}')$; $\tilde{\mathsf{J}}_3$ bezeichnet die assoziierte Jacobi-Matrix. Ferner seien $\mathbf{u}^{\mathrm{T}} := (\mathbf{F}_2^{\mathrm{T}}, \tilde{\mathbf{F}}_3^{\mathrm{T}}) : \mathbb{R}^n \to \mathbb{R}^N$ und $N := m + \mid \mathcal{I} \mid$. Schließlich bezeichnet $L(\mathbf{x}, \boldsymbol{\mu}, \boldsymbol{\lambda})$ die in (4.4.40) definierte *Lagrange-Funktion* des Problems **NLP**. Sei $\mathsf{J}^N = \mathsf{J}^N(\mathbf{x}) = \partial \mathbf{u}/\partial \mathbf{x}$ die Jacobi-Matrix von $\mathbf{u}$. die mit den Gleichungen und aktiven Ungleichungen assoziiert ist. Ein zulässiger Punkt $\mathbf{x}'$ heißt *regulär*, wenn gilt $\operatorname{rank}(\mathsf{J}^N(\mathbf{x}')) = N$ [Bock (1987, p. 48)].

Theorem 3 *Sei $\mathbf{x}_* \in \mathbb{R}^n$ ein regulärer Punkt und ein lokaler Minimierungspunkt des Problems **NLP**. Dann existieren Vektoren $\boldsymbol{\mu}_*$ und $\boldsymbol{\lambda}_*$ derartig, dass $\mathbf{x}_*$, $\boldsymbol{\mu}_*$ und $\boldsymbol{\lambda}_*$ den KKT-Bedingungen [Gleichungen (4.4.41) bis (4.4.42)] genügen.*

Zu beachten ist, dass der Unterschied zwischen Theorem 2 und 3 in den Annahmen, d. h. in den Regularitätsbedingungen, besteht. Definiert man weiter die Richtungen

$$\mathcal{T}(\mathbf{x}_*) := \left\{ \mathbf{p} \neq 0 \;\middle|\; \begin{array}{ll} \mathsf{J}_2(\mathbf{x}_*)\mathbf{p} = 0, & \\ \tilde{\mathsf{J}}_3(\mathbf{x}_*)\mathbf{p} \geq 0, & \tilde{\boldsymbol{\mu}}_{i*}\tilde{\mathsf{J}}_3(\mathbf{x}_*)\mathbf{p} = 0, \quad \forall\, i \in \mathcal{I}(\mathbf{x}_*) \end{array} \right\} \quad ,$$

und die Hesse-Matrix

$$\mathsf{H}(\mathbf{x}_*, \boldsymbol{\mu}_*, \boldsymbol{\lambda}_*) := \frac{\partial^2}{\partial \mathbf{x}^2} L(\mathbf{x}_*, \boldsymbol{\mu}_*, \boldsymbol{\lambda}_*) \quad ,$$

so kann Theorem 2 erweitert und das folgende Resultat abgeleitet werden:

Theorem 4 ***(Notwendige Bedingungen Zweiter Ordnung).*** *Sind die Funktionen $f(\mathbf{x})$, $\mathbf{F}_2(\mathbf{x})$ und $\mathbf{F}_3(\mathbf{x})$ zweimal stetig-differenzierbar, so lauten die notwendigen Bedingungen zweiter Ordnung dafür, dass ein KKT-Punkt ($\mathbf{x}_*$, $\boldsymbol{\mu}_*$, $\boldsymbol{\lambda}_*$) ein lokales Optimum ist:*

$$\mathbf{p}^{\mathrm{T}} \mathsf{H}(\mathbf{x}_*, \boldsymbol{\mu}_*, \boldsymbol{\lambda}_*)\, \mathbf{p} \geq 0, \quad \forall\, \mathbf{p} \in \mathcal{T}(\mathbf{x}_*) \quad . \tag{4.4.43}$$

Die Interpretation dieses Theorem besteht darin, dass die Hesse-Matrix der Lagrange-Funktion für alle Richtung $\mathbf{p} \in \mathcal{T}(\mathbf{x}_*)$ positiv-definit ist.

Theorem 5 ***(Hinreichende Bedingungen Zweiter Ordnung).*** *Sei ($\mathbf{x}_*$, $\boldsymbol{\mu}_*$, $\boldsymbol{\lambda}_*$) ein KKT-Punkt des Problems **NLP** und für alle Richtungen $\mathbf{p} \in \mathcal{T}(\mathbf{x}_*)$ sei die Hesse-Matrix der Lagrange-Funktion positiv-definit, d. h.*

$$\mathbf{p}^{\mathrm{T}} \mathsf{H}(\mathbf{x}_*, \boldsymbol{\mu}_*, \boldsymbol{\lambda}_*)\, \mathbf{p} > 0, \quad \forall\, \mathbf{p} \in \mathcal{T}(\mathbf{x}_*) \quad . \tag{4.4.44}$$

*Dann ist $\mathbf{x}_*$ ein striktes lokales Minimum des Problems **NLP**.*

Ein Beweis dieses Theorems, weitere Diskussionen der Bedingungen zweiter Ordnung und eine weniger restriktive Formulierung der Regularitätsbedingungen, die auf einer lokalen Charakterisierung der linearisierten Nebenbedingungen beruhen, sind in Fletcher (1987,[87]) gegeben.

Für eine spezielle Klasse von Problemen, nämlich konvexen Problemen, lässt sich das folgende Theorem beweisen [siehe z. B. Kuhn & Tucker (1951,[188]), Collatz & Wetterling (1971,[52]), oder Bomze & Grossmann (1993,[43]):

Theorem 6 ***(Kuhn-Tucker-Theorem; Hinreichende Bedingung)****. Gegeben sei das nichtlineare Optimierungsproblem* ***NLP****. Die Zielfunktion $f(\mathbf{x})$ sei konvex, $\mathbf{F}_2(\mathbf{x})$ linear und $\mathbf{F}_3(\mathbf{x})$ konkav. Existiert eine Lösung $(\mathbf{x}_*, \boldsymbol{\mu}_*, \boldsymbol{\lambda}_*)$, die den KKT-Bedingungen (4.4.41) bis (4.4.42) genügt, dann ist $\mathbf{x}_*$ eine optimale Lösung des Problems.*

Genügen die Funktionen $f(\mathbf{x})$, $\mathbf{F}_2(\mathbf{x})$ und $\mathbf{F}_3(\mathbf{x})$ den Annahmen[20] des Theorems 6, so nennt man **NLP** konvexes Optimierungsproblem. Für konvexe Optimierungsprobleme ist ein lokales sogleich ein globales Optimum [230].

Algorithmen zur Lösung von (4.4.38) sind z. B. in Gill *et al.* (1981,[104]) oder Fletcher (1987,[87]) beschrieben. Die meisten von ihnen bauen in irgend einer Weise auf Linearisierung auf; Ungleichungen werden durch Active-Set-Strategien berücksichtigt. Zu den meist genutzten und besten Verfahren gehören das in Abschnitt 4.4.3 beschriebene *Reduzierte-Gradienten-Verfahren*, die in Abschnitt 4.4.4 erläuterte *Sequentielle Quadratische Programmierung* und *Innere-Punkte-Methoden*, siehe z. B. Bazaraa *et al.* (1993,[24]) oder Wright (1996,[298]), für Probleme mit vielen Ungleichungen.

4.4.3 Reduzierte-Gradienten-Verfahren

Reduzierte Gradientenverfahren sind die natürliche Erweiterung des Simplexverfahrens auf nichtlineare Optimierungsprobleme. In jedem Iterationsschritt erfolgt mit Hilfe der aktiven Nebenbedingungen eine Einteilung in unabhängige (ungebundene) und abhängige (gebundene) Variablen und ein Minimierungsschritt für die Zielfunktion bezüglich der freien Variablen. Dieses Verfahren wurde ursprünglich von Abadie & Carpenter (1969,[4]) entwickelt [neuere Entwicklungen sind in Abadie (1978,[3]), Lasdon *et al.* (1978,[192]) und Lasdon & Waren (1978,[191]) sowie Gill *et al.* (1981,[104], Abschnitt 6.3) oder Spelluci (1993,[265], S. 361ff) beschrieben]. Zur Erläuterung dieses Verfahrens, das in einigen kommerziellen Softwarepaketen wie das `MINOS` [siehe z. B. Murtagh & Saunders (1978,[218]; 1982,[219])] oder das von Drud (1994,[73]) entwickelte Programm `CONOPT`, und das, da diese Programme wiederum in der Modellierungssprache `GAMS` [45] implementiert sind, sehr häufig zur Lösung praktischer Probleme, die auf große, dünn besetzte NLP-Probleme führen, verwendet wird, sei das Optimierungsproblem

$$\min_{\mathbf{x},\mathbf{y}} \left\{ f(\mathbf{x}) + \mathbf{c}^{\mathrm{T}}\mathbf{x} + \mathbf{d}^{\mathrm{T}}\mathbf{y} \;\middle|\; \begin{array}{c} \mathbf{h}(\mathbf{x}) + \mathsf{A}_1\mathbf{y} \circ \mathbf{b}_1 \\ \mathsf{A_2}\mathbf{x} + \mathsf{A_3}\mathbf{y} \circ \mathbf{b}_2 \end{array}, \mathbf{L} \le \mathbf{x}, \mathbf{y} \le \mathbf{U} \right\} \tag{4.4.45}$$

mit $\mathbf{x} \in \mathbb{R}^{n_x}$, $\mathbf{y} \in \mathbb{R}^{n_y}$, $\mathbf{g} : \mathbb{R}^{n_x} \to \mathbb{R}^{m_1}$, $\mathsf{A}_1 \in \mathcal{M}_{m_1,n_y}$, $\mathsf{A}_2 \in \mathcal{M}_{m_2,n_x}$, $\mathsf{A}_3 \in \mathcal{M}_{m_2,n_y}$ und $\mathbf{b}_i \in \mathbb{R}^{m_i}$ betrachtet; $\circ$ steht für eine der Relationen $\le$, $=$ oder $\ge$ in den Einzelkomponenten der vektoriellen Nebenbedingungen. Zu beachten ist, dass die Variablen $\mathbf{y}$ nur in den linearen Termen auftreten; sie werden daher hier *lineare Variablen* genannt. Entsprechend werden die Begriffe *nichtlineare Variablen* sowie lineare und nichtlineare Nebenbedingungen in diesem Abschnitt verwendet.

Ist $\mathbf{h}(\mathbf{x}) = 0$, d. h. nur die Zielfunktion enthält nichtlineare Terme, so kann das in Murtagh & Saunders (1978,[218]) beschriebene *reduzierte-Gradienten-Verfahren* nach Wolfe (1962,[295]) mit einem Quasi-Newton-Algorithmus kombiniert werden. Ähnlich wie in der LP, bei der die Nebenbedingungen $\mathsf{A}\mathbf{x} \le \mathbf{b}$ nach Einführung passender Schlupfvariablen $\mathbf{u} \ge \mathbf{0}$ und $\mathsf{A}\mathbf{x} + \mathbf{u} = \mathbf{b}$ und anschließender Nullpunkttransformation $\mathbf{s} = \mathbf{u} - \mathbf{b}$

[20] Diese Annahmen garantieren, dass sowohl der zulässige Bereich als auch die Zielfunktion konvex ist.

aus der Form $\mathsf{A}\mathbf{x} + \mathbf{s} = \mathbf{0}$ mit Hilfe von Basis- und Nichtbasisvariablen $\mathbf{x}^{\mathrm{B}}$ und $\mathbf{x}^{\mathrm{N}}$ in die Darstellung $\mathsf{B}\mathbf{x}^{\mathrm{B}}+\mathsf{N}\mathbf{x}^{\mathrm{N}} = \mathbf{0}$ überführt werden, wird bei Problemen mit nichtlinearer Zielfunktion mit Hilfe von $n^s \leq n^x$ sogenannten Superbasisvariablen $\mathbf{x}^{\mathrm{S}}$ die Zerlegung

$$\mathsf{B}\mathbf{x}^{\mathrm{B}}+\mathsf{S}\mathbf{x}^{\mathrm{S}}+\mathsf{N}\mathbf{x}^{\mathrm{N}} = \mathbf{0}$$

berechnet. Bei nichtentarteten optimalen Lösungen nehmen sowohl die Variablen $\mathbf{x}^{\mathrm{S}}$ als auch die Variablen $\mathbf{x}^{\mathrm{B}}$ Werte zwischen ihren unteren und oberen Schranken an, während die Nichtbasisvariablen $\mathbf{x}^{\mathrm{N}}$ wie in der LP an ihren Schranken fixiert sind. Allerdings werden in reduzierten-Gradienten-Verfahren auch die $\mathbf{x}^{\mathrm{S}}$ wie die $\mathbf{x}^{\mathrm{N}}$ in der LP als unabhängige Variablen angesehen, die sowohl den Zielfunktionswert als auch die Summe der Unzulässigkeiten minimieren können; die Basisvariablen $\mathbf{x}^{\mathrm{B}}$ werden weiterhin dazu verwendet, die linearen Nebenbedingungen zu erfüllen. Stellt man fest, dass mit der gegenwärtigen Zahl n^s keine Verbesserung erzielt werden kann, so werden einige der Nichtbasisvariablen ausgewählt und als Superbasisvariablen weiter verwendet, d. h. der Wert von n^s wird erhöht. Nimmt dagegen eine Basis- oder Superbasisvariable den Wert einer ihrer Schranken an, so wird sie als Nichtbasisvariable weiter geführt und n^s wird um den Wert eins verkleinert.

Zur Bestimmung der Superbasisvariablen führen wir die aus den Blöcken $\mathsf{B}^{-1}\mathsf{S}$, $\mathbb{1}$ und 0-Matrix zusammengesetzen Matrix

$$\mathsf{Z} = \begin{pmatrix} -\mathsf{B}^{-1}\mathsf{S} & \mathbb{1} & 0 \end{pmatrix}^{\mathrm{T}}$$

ein, sie wird aber so nicht direkt berechnet. Bei bekannter LU-Zerlegung der Basis-Matrix B, also der Produktdarstellung $\mathsf{B} = \mathsf{L}\mathsf{U}$, wobei die Matrix L (U) eine untere (obere) Dreiecksmatrix ist, und Lösung linearer Gleichungssysteme, in denen B und B^{T} auftreten, lassen sich Produkte der Form $\mathsf{Z}\mathbf{q}$ und $\mathsf{Z}^{\mathrm{T}}\mathbf{g}$ berechnen. Die Superbasisvariablen werden mit Hilfe eines Quasi-Newton-Verfahrens berechnet; eine Suchrichtung ergibt sich aus

$$\mathsf{R}^{\mathrm{T}}\mathsf{R}\mathbf{q} = -\mathsf{Z}^{\mathrm{T}}\mathbf{g} \quad , \quad \mathbf{g} := \boldsymbol{\nabla}_{\mathbf{x}} f(\mathbf{x}) \quad ,$$

wobei $\mathsf{Z}^{\mathrm{T}}\mathbf{g}$ der reduzierte Gradient und R eine obere Dreiecksmatrix ist. Die Matrix kann durch verschiedene Update-Verfahren berechnet werden, um mit Hilfe von

$$\mathsf{R}^{\mathrm{T}}\mathsf{R} \approx \mathsf{Z}^{\mathrm{T}}\mathsf{H}\mathsf{Z}$$

die Hesse-Matrix H, d. h. die zweiten Ableitungen von $f(\mathbf{x})$, zu approximieren. Sobald $\mathbf{q}$ berechnet ist, folgt aus $\mathbf{p} = \mathsf{Z}\mathbf{q}$ die Suchrichtung für alle Variablen. Daran schließt sich eine Liniensuche zur Lösung des eindimensionalen Problems

$$\min_{\boldsymbol{\alpha}} \left\{ \; f(\mathbf{x}+\alpha\mathbf{p}) \;\middle|\; 0 \leq \alpha \leq \alpha^{+} \right\}$$

an, wobei sich α^{+} aus den Schranken der Variablen ableitet. Wie in der LP lassen sich die dualen Werte $\boldsymbol{\pi}$ bzw. Schattenpreise auch hier aus einer linearen Gleichungen ableiten:

$$\mathbf{g}\mathsf{B} - \mathsf{B}^{\mathrm{T}}\boldsymbol{\pi} = 0 \quad ,$$

wobei $\mathbf{g}_{\mathsf{B}}$ der mit den Basisvariablen assoziierte Gradient der Zielfunktion ist. Die entsprechende Größe für die Superbasisvariablen ist

$$\mathsf{Z}^{\mathrm{T}}\mathbf{g} := \mathbf{g}_{\mathbf{s}} - \mathbf{s}^{\mathrm{T}}\boldsymbol{\pi} = 0 \quad ;$$

im optimalen Lösungspunkt sollte $\mathsf{Z}^{\mathrm{T}}\mathbf{g} = \mathbf{0}$ gelten.

Ist $\mathbf{h}(\mathbf{x}) \neq 0$, so bietet sich, wie in `GAMS/MINOS` implementiert, der auf Robinson (1972,[248]) zurückgehende und in Murtagh & Saunders (1982,[219]) beschriebene projizierende Lagrange-Algorithmus an. In diesem iterativen Verfahren werden die Nebenbedingungen linearisiert; die Folge der Probleme mit nichtlinearer Zielfunktion, aber mit linearen Nebenbedingungen, wird dann mit dem auf Seite 112 beschriebenen reduzierten Gradienten-Verfahren gelöst.

Zu Beginn der k-ten Iteration sei $\mathbf{x}_k$ ein Schätzwert für die nichtlinearen Variablen und $\boldsymbol{\lambda}_k$ ein Schätzwert für die mit den nichtlinearen Nebenbedingungen assoziierten Lagrange-Multiplikatoren (oder dualen Werte). Die Linearisierungen ergeben sich zu

$$\mathbf{h}^L(\mathbf{x}, \mathbf{x}_k) = \mathbf{h}(\mathbf{x}_k) + \mathsf{J}(\mathbf{x}_k)(\mathbf{x} - \mathbf{x}_k)$$

bzw. in Kurzform $\mathbf{h}^L = \mathbf{h}_k + \mathsf{J}_k(\mathbf{x} - \mathbf{x}_k)$, wobei $\mathsf{J}_k = \mathsf{J}(\mathbf{x}_k)$ die Jacobi-Matrix im Punkt $\mathbf{x}_k$ bezeichnet; die i-te Zeile der Jacobi-Matrix ist der Gradientenvektor der i-ten nichtlinearen Nebenbedingung. Damit ergibt sich das Minimierungsproblem

$$\min_{\mathbf{x},\mathbf{y}} \left\{ f(\mathbf{x}) + \mathbf{c}^{\mathrm{T}}\mathbf{x} + \mathbf{d}^{\mathrm{T}}\mathbf{y} - \boldsymbol{\lambda}_k\mathrm{T}(\mathbf{x} - \mathbf{x}_k) + \frac{1}{2}\rho(\mathbf{h} - \mathbf{h}^L)^{\mathrm{T}}(\mathbf{h} - \mathbf{h}^L) \right\} \tag{4.4.46}$$

unter den Nebenbedingungen

$$\begin{array}{l} \mathbf{h}^L + \mathsf{A}_1\mathbf{y} \circ \mathbf{b}_1 \\ \mathsf{A}_\mathbf{2}\mathbf{x} + \mathsf{A}_\mathbf{3}\mathbf{y} \circ \mathbf{b}_2 \end{array} \quad , \quad \mathbf{L} \leq \mathbf{x}, \mathbf{y} \leq \mathbf{U}$$

mit passendem skalarem Parameter ρ in der quadratischen Straffunktion $\frac{1}{2}\rho(\mathbf{h}-\mathbf{h}^L)^{\mathrm{T}}(\mathbf{h}-\mathbf{h}^L)$. Ähnlich wie auf Seite 112 werden wieder Schlupfvariablen $\mathbf{s}_1$ und $\mathbf{s}_2$ eingeführt, wobei die rechten Seiten $\mathbf{b}_1$ und $\mathbf{b}_2$ wieder in die Schranken für $\mathbf{s}_1$ und $\mathbf{s}_2$ eingearbeitet wurden. Somit ergibt sich das lineare Gleichungssystem

$$\begin{pmatrix} \mathsf{J}_k & \mathsf{A}_1 \\ \mathsf{A}_\mathbf{2} & \mathsf{A}_\mathbf{3} \end{pmatrix} \begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix} + \begin{pmatrix} \mathbb{1} & 0 \\ 0 & \mathbb{1} \end{pmatrix} \begin{pmatrix} \mathbf{s}_1 \\ \mathbf{s}_2 \end{pmatrix} = \begin{pmatrix} \mathsf{J}_k\mathbf{x}_k - \mathbf{g}_k \\ \mathbf{0} \end{pmatrix}$$

und das reduzierte Gradienten-Verfahren von Seite 112 kann zur Anwendung kommen.

Ein Nachteil des vorgestellten reduzierten Gradientenverfahrens ist die Linearisierung, die das Verfahren damit in die Gruppe der sequentiellen linearen Verfahren einordnet; diese Verfahren linearisieren die Nebenbedingungen und verwenden eine zusammengesetzte Lagrange-Funktion als Zielfunktion, in der z. B. Strafterme auftreten.

Sequentielle lineare Verfahren iterieren während der Liniensuche entlang der Tangentenrichtung. Dagegen bewegt sich das von Drud (1994,[73]) in `CONOPT` verwendete Verallgemeinerte-Reduzierte-Gradienten-Verfahren [engl.: *generalized reduced gradient; GRG*], wie für diese Klasse üblich, entlang der aktuellen, durch die Nebenbedingungen vorgegebenen Hyperflächen. In jeder Iteration der Liniensuche werden die Basisvariablen derartig modifiziert, so dass sich wiederum eine zulässige Lösung ergibt; damit wird die Liniensuche im GRG-Verfahren aufwendiger, was sich aber auszahlt, wenn die Anzahl der (äußeren) Iterationen klein ist. Das in `CONOPT` implementierte GRG-Verfahren lässt sich in Kurzform so beschreiben:

1. Initialisierung des Verfahrens.
2. Bestimmung eines zulässigen Punktes.

3. Berechnung der mit den Nebenbedingungen assoziierten Jacobi-Matrix J.
4. Selektion der Menge von Basisvariablen $\mathbf{x}^{\mathrm{B}}$ derart, dass die Untermatrix von J, die durch die zu zu den Basisvariablen gehörenden Spalten gebildet wird, regulär ist; anschließende Faktorisierung, d. h. LU-Zerlegung der Basismatrix $\mathtt{B}$; die übrigen Variablen werden als Nichtbasisvariablen $\mathbf{x}^{\mathrm{N}}$ fortgeführt.
5. Berechnung der Lagrange-Multiplikatoren $\mathsf{B}^{\mathrm{T}}\boldsymbol{\pi} = \mathbf{g}$.
6. Berechnung des reduzierten Gradienten $\mathbf{r} := \mathbf{g} - \mathsf{B}^{\mathrm{T}}\boldsymbol{\pi}$.
7. Abbruch, wenn der auf die Schranken projizierte Wert von $\mathbf{r}$ hinreichend klein ist; der aktuelle Punkt wird als optimale Lösung angesehen; andernfalls wird mit Schritt 8 fortgefahren.
8. Bestimmung einer Suchrichtung $\mathbf{d}$ für die Nichtbasisvariablen; die Berechnung basiert auf Kenntnis von und erlaubt die Berechnung der Tangentenrichtung.
9. Durchführung einer Liniensuche entlang der Richtung $\mathbf{d}$. Bei jedem Schritt werden die Basisvariablen $\mathbf{x}^{\mathrm{B}}$ so modifiziert, dass $\mathbf{h}(\mathbf{x}^{\mathrm{B}}, \mathbf{x}^{\mathrm{N}}) = \mathbf{b}$ erfüllt ist, wobei die faktorisierte Darstellung von B in einem Pseudo-Newton-Verfahren verwendet wird.
10. Fortsetzung mit Schritt 3.

4.4.4 Sequentielle Quadratische Programmierung

Sequentielle Quadratische Programmierung [engl.: *sequentiell quadratic programming; SQP*] ist eine sehr effizientes Optimierungsverfahren [267] zur Lösung des Problems (4.4.38). Ein weit verbreitetes, auf SQP aufbauendes und vielfach verwendetes Software-Paket ist das von Gill *et al.* (1997,[102]) entwickelte Programm `SNOPT`. Die Grundidee des Verfahrens besteht darin, (4.4.38) mit Hilfe einer Folge von quadratischen Optimierungsproblemen zu lösen. Das Unterproblem in Iteration k erscheint als

$$\min_{\Delta\mathbf{x}} \left\{ \tfrac{1}{2}\Delta\mathbf{x}^{\mathrm{T}}\mathsf{H}_k\Delta\mathbf{x} + \boldsymbol{\nabla} f(\mathbf{x}_k)^{\mathrm{T}}\Delta\mathbf{x} \right\} \quad , \quad \Delta\mathbf{x} \in \mathbb{R}^n \tag{4.4.47}$$

$$\begin{aligned} \mathsf{J}_2(\mathbf{x}_k)^{\mathrm{T}}\Delta\mathbf{x} + \mathbf{F}_2(\mathbf{x}_k) &= 0 \quad , \\ \mathsf{J}_3(\mathbf{x}_k)^{\mathrm{T}}\Delta\mathbf{x} + \mathbf{F}_3(\mathbf{x}_k) &\geq 0 \quad , \end{aligned}$$

wobei der Index k die Größen bezeichnet, die zu Beginn der Iteration k bereits bekannt sind, und $\Delta\mathbf{x}$ der zu bestimmende Korrekturvektor ist. Zur Lösung dieses Unterproblem [siehe z. B. Gill *et al.* (1981, Abschnitt 6.5.3)] werden die Nebenbedingungen linearisiert und die Taylorreihe der Zielfunktion nach dem quadratischen Glied abgeschnitten; der konstante Term $f(\mathbf{x}_k)$ braucht nicht weiter betrachtet zu werden. Die notwendigen Bedingungen, die sich aus der zu (4.4.47) gehörenden Lagrange-Funktion ableiten, lauten

$$\mathsf{H}_k\Delta\mathbf{x} + \boldsymbol{\nabla} f(\mathbf{x}_k) - \mathsf{J}_2(\mathbf{x}_k)\tilde{\lambda}_{k+1} - \mathsf{J}_3(\mathbf{x}_k)\tilde{\mu}_{k+1} = \mathbf{0} \quad .$$

Bezeichnet $\boldsymbol{\lambda}_k$ den Vektor der Lagrange-Multiplikatoren (der Einfachheit wegen wird nicht zwischen den Lagrange-Multiplikatoren $\boldsymbol{\lambda}$ und $\boldsymbol{\mu}$ für Gleichungen und Ungleichungen unterschieden), der zu Beginn der Iteration k bekannt ist, und sind $\Delta\mathbf{x}_k$, $\tilde{\boldsymbol{\lambda}}_k$ und $\tilde{\boldsymbol{\mu}}_k$ die Lösungen von (4.4.47), so folgt die nächste Iteration aus

$$\begin{pmatrix} \mathbf{x}_{k+1} \\ \boldsymbol{\lambda}_{k+1} \\ \boldsymbol{\mu}_{k+1} \end{pmatrix} = \begin{pmatrix} \mathbf{x}_k \\ \boldsymbol{\lambda}_k \\ \boldsymbol{\mu}_k \end{pmatrix} + \alpha_k \begin{pmatrix} \Delta\mathbf{x}_k \\ \Delta\boldsymbol{\lambda}_k \\ \Delta\boldsymbol{\mu}_k \end{pmatrix} \quad , \quad \begin{pmatrix} \Delta\boldsymbol{\lambda}_k = \tilde{\boldsymbol{\lambda}}_k - \boldsymbol{\lambda}_k \\ \Delta\boldsymbol{\mu}_k = \tilde{\boldsymbol{\mu}}_k - \boldsymbol{\mu}_k \end{pmatrix} \quad ,$$

wobei α_k ein geeigneter Dämpfungsfaktor ist.

Für die Lösung der quadratischen Unterprobleme sei auf Gill *et al.* (1981, Abschnitt 5.3.2) und Fletcher (1987, [87], Kapitel 10) verwiesen.

4.4.5 Innere-Punkte-Methoden

Innere-Punkte-Methoden sind spezielle Algorithmen zur Lösung nichtlinearer beschränkter Optimierungsprobleme. Ausgehend von einer zulässigen Startlösung, die im Inneren $\mathcal{S}$ des zulässigen Bereichs S liegen muss, bestimmen diese Algorithmen iterativ die Lösung, und nähern sich in $\mathcal{S}$ asymptotisch der bei linearen Programmen auf dem Rand liegenden Lösung an. Mit Hilfe der notwendigen und hinreichenden Bedingungen für die Existenz lokaler Optima, d. h. der KKT-Bedingungen [265], wird dieses Problem auf die Lösung eines nichtlinearen Gleichungssystems reduziert. Dieses kann z. B. mit Hilfe des Newton-Verfahrens gelöst werden. In der linearen Programmierung eignen sich IPM insbesondere für große, dünn besetzte Systemmatrizen oder solche, die nahezu entartet sind; siehe hierzu auch Anhang A.3.

4.5 Gemischt-ganzzahlige nichtlineare Optimierung

4.5.1 Einführende Bemerkungen

MINLP-Probleme wie (1.8.1) gehören zu den schwierigsten Optimierungsproblemen überhaupt, denn sie gehören zur Klasse der $\mathcal{NP}$-vollständigen Probleme. Dies bedeutet, dass zur Zeit kein Algorithmus bekannt ist, der (1.8.1) in polynomialer Zeit (polynomial in der Problemgröße anwachsend) löst. Würde einer erfunden, so könnten damit auch andere Probleme in $\mathcal{NP}$ in polynomialer Zeit gelöst werden.

Obwohl MILP-Probleme kombinatorische Optimierungsprobleme mit einer gewöhnlich exponentiell wachsenden Anzahl von ganzzahlig-zulässigen Punkten sind, wurden hierzu erfolgreich effiziente B&B-Verfahren entwickelt, die auf LP-Relaxierung beruhen und in endlich vielen Schritten Optimalität beweisen. Nichtlineare, kontinuierliche Optimierungsprobleme (NLP-Probleme) können dagegen nicht in einer endlichen Anzahl von Schritten gelöst werden, sondern nur iterativ [24]. Dennoch lassen sich (zumindest konvexe) NLP-Probleme in gewisser Weise einfacher lösen als MILP-Probleme, da sich unter Verwendung von *sequentieller quadratischer Optimierung* meist ein Konvergenzverhalten zweiter Ordnung einstellt und somit die Lösung recht schnell bestimmt werden kann.

MINLP-Probleme vereinigen die Schwierigkeiten beider Teilklassen: MILP und NLP, aber sie besitzen darüber hinaus noch einige Eigenschaften, die einzigartig in dem Sinn sind, dass sie weder bei NLP- noch bei MILP-Problemen auftreten. Während bei konvexen NLP-Problemen mit der Bestimmung eines lokalen Minimums bereits das globale Minimum bestimmt ist, ist dies bei MINLP-Problemen z. B. nicht der Fall, wie man sich anhand des Optimierungsproblems

$$\min\ z \quad , \quad z = \alpha_1 + \alpha_2$$

unter den Nebenbedingungen

$$\alpha_1^2 + \alpha_2^2 \leq 1.96 \quad , \quad \alpha_1, \alpha_2 \in \mathbb{N}_0$$

leicht klar machen kann. Der zulässige Bereich S ist der abgeschlossene Kreis mit Radius $R = 1.4$. Da S konvex und die Zielfunktion linear ist, handelt es sich gemäß Theorem 6 auf Seite 112 um ein konvexes Optimierungsproblem. Unter Vernachlässigung der Ganzzahligkeitsbedingungen lautet die optimale Lösung ($\alpha_1 = \alpha_1 = \sqrt{0.98}$). Berücksichtigt man aber die Ganzzahligkeitsbedingung $\alpha_1, \alpha_2 \in \mathbb{N}$, so hat man $(1, 0)$ und $(0, 1)$ als optimale Lösungspunkte.

Lösungsalgorithmen, die das Optimierungsproblem (1.8.1) exakt, d. h. bis auf jede vorgegebene ε-Schranke lösen können, werden in der *Globalen Optimierung*,, einer speziellen Disziplin der mathematischen Optimierung, entwickelt; siehe dazu Abschnitt 4.6. Spezielle Algorithmen wurden konstruiert, die ausnutzen, dass die Menge des zulässigen Bereichs und die Zielfunktion *konvex* sind. Hierbei wird ausgenutzt, dass die Verbindungsgerade zweier Punkte $f(\mathbf{x}_1)$ und $f(\mathbf{x}_2)$ des Graphen der konvexen Funktion f nie „unterhalb" des Graphen liegt, und die Tangente an einem Punkt $(\mathbf{x}_0, f(\mathbf{x}_0))$ des Graphen stets unterhalb von f liegt, d. h.

$$f(\mathbf{x}) \geq f(\mathbf{x}_0) + f'(\mathbf{x}_0)(\mathbf{x} - \mathbf{x}_0) \quad , \quad \forall \mathbf{x} \quad . \tag{4.5.1}$$

Das Problem (1.8.1) kann unter bestimmten Voraussetzungen mit deterministischen Methoden exakt, mit heuristischen Methoden dagegen bestenfalls approximativ gelöst werden. Allen deterministischen Verfahren ist gemeinsam, dass sie sich einer vollständigen Enumeration eines Such-Baums bedienen bzw. Regeln implementieren, um die Suche durch Unterbäume zu begrenzen oder ganz zu vermeiden. Deterministische Verfahren erbringen den Nachweis, dass eine bestimmte gefundene Lösung optimal ist. Heuristische Verfahren sind nicht in der Lage, diesen Nachweis zu erbringen. In vielen Fällen wurden bei nichtkonvexen Problemen nur Heuristiken angewendet.

Ein offensichtliches deterministisches Verfahren besteht darin, sämtliche Kombinationen U diskreter Variablen $\mathbf{y}_i$ im Problem zu fixieren, das übrigbleibende, durch $\mathbf{y}_i$ parametrisierte nichtlineare Problem zu lösen [damit wird ein Paar $\mathbf{x}_i, z_i = f(\mathbf{x}_i, \mathbf{y}_i)$ bestimmt], und dann von allen Problemen dasjenige mit kleinstem z_i auszuwählen (dieses sei mit i^* bezeichnet), so dass die Lösung des Problems durch das Tripel $(\mathbf{x}^* = \mathbf{x}_{i^*}, \mathbf{y} = \mathbf{y}_{i^*}, z_i^* = f(\mathbf{x}_{i^*}, \mathbf{y}_{i^*}))$ repräsentiert wird. Dieses Verfahren funktioniert prinzipiell, wenn U nur endlich viele Elemente enthält und das globale Minimum der nichtlinearer Unterprobleme bestimmt werden kann, ist aber nicht sehr effizient.

4.5.2 Exakte Verfahren zur Lösung von konvexen MINLP-Problemen

Deterministische Verfahren (für konvexe MINLP-Probleme) lassen sich in die folgenden drei Hauptklassen untergliedern:

- *Branch & Bound*; siehe z. B. Gupta & Ravindran (1985,[124]),
- *Bendersdekomposition* (GBD); siehe z. B. Geoffrion (1972,[99]),
- *Äußere Approximation* (OA); siehe z. B. Duran & Grossmann (1986,[75]) oder Viswanathan & Grossmann (1990,[286]).

Das B&B-Verfahren für MINLP-Probleme basiert auf denselben Ideen wie das LP-basierte B&B-Verfahren für MILP-Probleme. Der erste Schritt besteht darin, dasjenige relaxierte Problem zu lösen, das entsteht, wenn die Ganzzahligkeitsbedingung diskreter Variablen vernachlässigt wird. Liefert das relaxierte Problem eine ganzzahlig-zulässige Lösung, so ist dies bereits die optimale Lösung und das Verfahren terminiert. Andernfalls liefert die Lösung des relaxierten Problems eine untere Schranke (allerdings nur eine sichere, wenn das globale Minimum des NLP-Problems bestimmt werden kann) für das ursprüngliche Optimierungsproblem und der Such-Baum wird weiter aufgebaut. Hat man eine ganzzahlig-zulässige Lösung bestimmt, so liefert diese eine obere Schranke für das ursprüngliche Problem. Ein wesentlicher Nachteil der B&B-Idee bei MINLP-Problemen ist die Schwierigkeit, dass bei der Auswertung der Knoten der nachfolgenden Generation kaum die Struktur ihrer Vorgängergeneration ausgenutzt werden kann wie dies bei MILP-Problemen unter Verwendung des dualen Simplexverfahrens der Fall ist.

Das Dekompositionsverfahren von Geoffrion (1972,[99]) unterteilt die Variablen in zwei Mengen, die schwierige – das sind bei MINLP-Problem meist die ganzzahligen Variablen – und nichtschwierige Variablen enthalten. Dann wird eine Folge von NLP-Unterproblemen (erzeugt durch Fixierung der Binärvariablen $\mathbf{y}^k$) und MILP-Masterproblemen im Raum der „schwierigen" Variablen gelöst. Die Unterprobleme liefern obere Schranken für das ursprüngliche Problem, während das MILP-Masterproblem neue Kombinationen der Binärvariablen $\mathbf{y}^k$ für ein weiteres NLP-Unterproblem liefert. Wenn das ursprüngliche Problem konvex ist, so liefert das Masterproblem eine monoton wachsende Folge von unteren Schranken. Das Verfahren terminiert bei Gleichheit der besten oberen und der unteren Schranke bzw. wenn sich beide Schranken um weniger als einen vorgegebenen Wert unterscheiden.

Äußere-Approximations-Verfahren [engl.: *outer approximation methods*] verwenden ebenfalls eine Folge von NLP Unterproblemen (erzeugt durch Fixierung der Binärvariablen $\mathbf{y}^k$) und MILP Masterproblemen. Der wesentliche Unterschied besteht in der Definition der Masterprobleme. Bei OA werden die Masterprobleme durch „äußere Approximationen" (Linearisierungen) der nichtlinearen Nebenbedingungen in *den* Punkten erzeugt, die die optimale Lösungen der NLP-Teilprobleme sind. Bei konvexen MINLP-Problemen wird damit eine Obermenge des zulässigen Bereichs erzeugt, so dass die MILP-Masterprobleme wieder eine monoton wachsende Folge von unteren Schranken liefern; dieses Verfahren ist detailliert im Anhang A.4 beschrieben. Als Abbruchkriterium wird die Gleichheit von oberen und unteren Schranken verwendet. Während das GBD-Masterproblem kleiner in der Anzahl der Variablen und Nebenbedingungen ist, liefert der OA-Algorithmus schärfere untere Schranken und benötigt weniger Iterationen bis zur Konvergenz. Für konvexe NLP-Problemen kann sogar Optimalität bewiesen werden.

Beide Verfahren, sowohl der GBD als auch der OA-Algorithmus sind inzwischen um heuristische Erweiterungen ergänzt worden, die die Behandlung nichtkonvexer MINLP-Probleme erlauben, allerdings keinen Optimalitätsnachweis liefern, sondern nur zulässige Punkte berechnen; dennoch finden sich in der Praxis erfolgreiche Anwendungen [154].

4.6 Globale Optimierung

Das globale Minimum des Problems (1.8.1) kann ohne weitere Annahmen an Konvexität nur mit Methoden der globalen Optimierung ([89], [138], [136], [139], [175], [269], [270], [91], [193], [90], [213]) bestimmt werden; die in der Publikation von Misener & Floudas

Abbildung 4.8 Tangentialebenen an eine konvexe Menge. Der zulässige Bereich eines konvexen Optimierungsproblems kann durch zunächst unendlich viele Tangentialebenen beschrieben werden. Für MINLP-Probleme, in denen es nur eine endliche Anzahl von Kombinationen der ganzzahligen Variablen existiert, kann gezeigt werden, dass bereits eine endliche Anzahl von Tangentialebenen ausreicht.

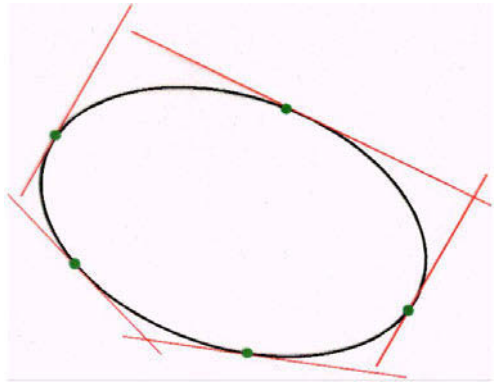
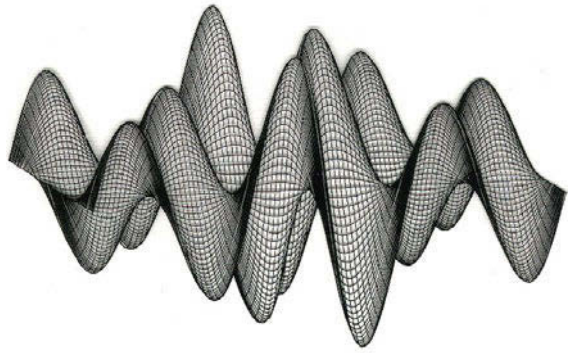


Abbildung 4.9 Diese Abbildung zeigt die Fläche einer nichtkonvexen zweidimensionalen Funktion mit vielen lokalen Minima und Maxima.



(2012,[213]) enthaltene Literaturübersicht sei besonders empfohlen. Bei den deterministischen Verfahren – und nur die sollen betrachtet werden – wird zu jeder vorgegebenen ε-Schranke eine obere und untere Schranke z^o und z^u für die Zielfunktion bestimmt, sodass schließlich $z^u \leq f(\mathbf{x}, \mathbf{y}) \leq z^o$ und $z^o - z^u \leq \varepsilon$ gilt. Die obere Schranke z^o ergibt sich dabei als Zielfunktionswert $z^o = f(\mathbf{x}, \mathbf{y})$ eines zulässigen Punktes $(\mathbf{x}, \mathbf{y})$ oder besser noch eines lokalen Minimums $(\mathbf{x}_*, \mathbf{y}_*)$, das mit einem Lösungsverfahren zur Bestimmung stationärer Punkte in einem kontinuierlichen nichtlinearen Problem oder einer Heuristik gewonnen wurde; die Bestimmung der oberen Schranke ist relativ einfach. Zur Bestimmung der unteren Schranke z^u eignen sich z. B.

- *Intervallmethoden* ([175] oder [239]), auf die hier aber nicht weiter eingegangen

werden soll, oder

- *konvexe Relaxierung* unter Verwendung konvexer Unterschätzern und konvexifizierter Mengen,

die mit einem räumlichen B&B-Verfahren kombiniert werden.

Eine konvexe Funktion f_u heißt *konvexer Unterschätzer* [engl.: *convex underestimator*] zu einer gegebenen Funktion f auf der Menge S, wenn $f_u(\mathbf{x}) \leq f(\mathbf{x})$ für alle $\mathbf{x} \in S$. Als Beispiel sei die nichtkonvexe Funktion $f(x_1, x_2) = x_1 x_2$ auf dem Intervall $[X_1^-, X_1^+] \times [X_2^-, X_2^+]$ betrachtet; sie besteht aus nur einem bilinearen Term und kann durch die Variable y und den folgenden vier Ungleichungen

$$\begin{array}{rcl} X_1^- x_2 + X_2^- x_1 - X_1^- X_2^- & \leq & y \leq -X_1^+ x_2 - X_2^- x_1 + X_1^- X_2^+ \\ X_1^+ x_2 + X_2^+ x_1 - X_1^+ X_2^+ & \leq & y \leq -X_1^- x_2 - X_2^+ x_1 + X_1^- X_2^+ \end{array}$$

linear konvex unterschätzt werden [siehe z. B. Al-Khayyal & Falk (1983,[11]), Al-Khayyal (1990,[10]) und McCormick (1976,[206])]. Dabei kann der Fehler höchstens

$$\max_{x_1, x_2 \in [X_1^-, X_1^+] \times [X_2^-, X_2^+]} (x_1 x_2 - y) = \frac{1}{4} \left(X_1^+ - X_1^-\right) \left(X_2^+ - X_2^-\right)$$

betragen. Für rationale Terme x_1/x_2, die im Intervall $[X_1^-, X_1^+] \times [X_2^-, X_2^+]$ keinen Vorzeichenwechsel haben, ergeben sich entsprechend die linearen Ungleichungen

$$\begin{array}{rcll} y & \geq & X_1^-/x_2 + x_1/X_2^+ - X_1^-/X_2^+ & \text{falls } X_1^- \geq 0 \\ y & \geq & x_1/X_2^+ + X_1^- x_2/X_2^- X_2^+ + X_1^-/X_2^- & \text{falls } X_1^- < 0 \\ y & \geq & X_1^+/x_2 + x_1/X_2^- - X_1^-/X_2^+ & \text{falls } X_1^+ \geq 0 \\ y & \geq & x_1/X_2^- - X_1^+ x_2/X_2^- X_2^+ + X_1^-/X_2^- & \text{falls } X_1^- < 0 \quad . \end{array}$$

Schließlich können univariate konkave Funktionen $f(x)$, also Funktionen mit $f''(x) < 0$ auf dem Intervall $[X^-, X^+]$, bestmöglich durch

$$f(x) \geq f(X^-) + \frac{f(X^+) - f(X^-)}{X^+ - X^-} \left(x - X^-\right)$$

linear konvex unterschätzt werden. Allgemeine, zweimal stetig-differenzierbare nichtkonvexe Funktionen $f(\mathbf{x})$ können für $\mathbf{x} \in \mathbb{R}^n$ im Intervall $[\mathbf{X}^-, \mathbf{X}^+]$, wie in Maranas & Floudas (1994,[276]) abgeleitet, durch

$$F(\mathbf{x}) := f(\mathbf{x}) - \sum_{i=1}^{n} \alpha_i \left(X_i^+ - x_i\right) \left(x_i - X_i^-\right)$$

konvex unterschätzt werden, wobei $F(\mathbf{x}) \leq f(\mathbf{x})$ und

$$\alpha_i \geq -\frac{1}{2} \min_{i, \mathbf{x} \in [\mathbf{X}^-, \mathbf{X}^+]} \lambda_i \left\{\mathsf{H}(\mathbf{x})\right\} \geq 0 \tag{4.6.2}$$

gilt, $\mathsf{H}(\mathbf{x})$ die Hesse-Matrix von $f(\mathbf{x})$ ist und $\lambda_i \left\{\mathsf{H}(\mathbf{x})\right\}$ der i^{te} Eigenwert von $\mathsf{H}(\mathbf{x})$ ist; zu beachten ist: Es muss jeweils der kleinste Eigenwert berechnet werden, den $\mathsf{H}(\mathbf{x})$ auf dem Intervall $[\mathbf{X}^-, \mathbf{X}^+]$ annimmt. Für die maximale Distanz gilt die Beziehung

$$d_{\max} := \max_{\mathbf{x}\in[\mathbf{X}^-,\mathbf{X}^+]} (f(\mathbf{x}) - F(\mathbf{x})) = \frac{1}{4}\sum_{i=1}^{n} \alpha_i \left(X_i^+ - X_i^-\right)^2 \tag{4.6.3}$$

und $F(\mathbf{x})$ ist genau dann konvex auf dem Intervall $[\mathbf{X}^-, \mathbf{X}^+]$, wenn $\mathsf{H}_F(\mathbf{x}) + 2\Delta$ positiv definit ist für alle $\mathbf{x} \in [\mathbf{X}^-, \mathbf{X}^+]$, wobei $\mathsf{H}_F(\mathbf{x})$ die Hesse-Matrix von F ist und $\Delta := diag\{\alpha_i\}$ diagonale Shiftmatrix genannt wird. Das hier skizzierte Verfahren heißt *nichtuniformes Diagonales-Shift-Verfahren* ([5], [6] und [9], Abschnitte 3.4 und 3.5; wird statt der α_i-Werte ein globales α, also $\Delta := \alpha\mathbb{1}$ verwendet, so spricht man von einem *uniformen Diagonalen-Shift-Verfahren* – in diesem Fall ist $d_{\max}$ proportional zu α.

Die Eigenwerte können exakt bzw. mit *Methoden der Intervallarithmetik* – siehe z. B. [239] oder [175] – berechnet werden. Dieser Schritt ist bedeutend für die Effizienz des Verfahrens; die α_i sollten schließlich so klein wie nach (4.6.2) gerade noch möglich gewählt werden, um auch $d_{\max}$ so klein und die Unterschätzer so genau wie möglich werden zu lassen. Im nachfolgenden Beispiel

$$\min_{-1\le x_1\le 2, -1\le x_2\le 1} f(x_1, x_2) \quad , \quad f(x_1, x_2) := \cos x_1 \sin x_2 - \frac{x_1}{x_2^2}$$

sei wenigsten angedeutet, worauf hierbei zu achten ist. Die Hesse-Matrix folgt daraus – im allgemeinen Fall z. B. mit einem automatischen Differenzierungsalgorithmus – zu

$$\mathsf{H}_f(\mathbf{x}) := \begin{pmatrix} -\cos x_1 \sin x_2 & -\sin x_1 \cos x_2 + \frac{2x_2}{\left(x_2^2+1\right)^2} \\ -\sin x_1 \cos x_2 + \frac{2x_2}{\left(x_2^2+1\right)^2} & -\cos x_1 \sin x_2 + \frac{2x_1\left(x_2^2+1\right)^2 - 8x_1x_2^2\left(x_2^2+1\right)^2}{\left(x_2^2+1\right)^4} \end{pmatrix} \quad ;$$

für die Anwendung der Intervall-Arithmetik ist wichtig und notwendig, dass $\mathsf{H}_f(\mathbf{x})$ analytisch berechnet werden kann. Wertet man diese Matrix unter Berücksichtigung der angegebenen Schranken der Variablen x_1 und x_2 aus, so folgt die Intervall-Abschätzung

$$\mathsf{H}_f(\mathbf{x}) \subseteq [\mathsf{H}_f] = \begin{pmatrix} [-0.84148, +0.84148] & [-3.00000, +2.84148] \\ [-3.00000, +2.84148] & [-40.84148, +32.84148] \end{pmatrix} \quad .$$

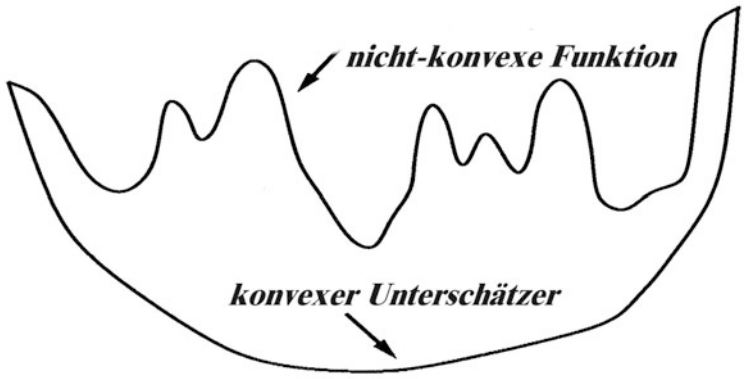
Diese Elemente kann man aber noch weiter verfeinern, in dem man für jedes Element über den zulässigen Bereich das globale Minimum bzw. Maximum bestimmt. Damit erhält man dann die optimale Intervall-Hesse-Matrix

$$[\mathsf{H}_f]_* = \begin{pmatrix} [-0.84148, +0.84148] & [-1.52288, +1.38086] \\ [-1.52288, +1.38086] & [-2.00608, +4.00181] \end{pmatrix} \quad .$$

Hier sind folgende Ergebnisse relevant. Der exakte kleinste Eigenwert für $(x_1, x_2) \in [-1, 2] \times [-1, 1]$ ist $\lambda^e_{\min}(\mathsf{H}_f) = -2.3934$; diesen kann man z. B. genähert bestimmen, indem man für ein sehr feines Gitter jeweils die Eigenwerte der Matrix $\mathsf{H}_f(\mathbf{x})$ ausrechnet. Der kleinste Eigenwert von Intervallmatrizen kann z. B. mit dem Verfahren von Hertz [133] berechnet werden; die Grundidee ist dabei, aus der Intervall-Hesse-Matrix eine endliche Untermenge reeller Matrizen so zu konstruieren, dass der kleinste Eigenwert über all diese Matrizen der Untermenge gleich dem kleinsten Eigenwert der Intervall-Matrix ist. Damit erhält $\lambda^e_{\min}([\mathsf{H}_f]) = -41.0652$ und $\lambda^e_{\min}([\mathsf{H}_f]_*) = -3.0817$. Damit ist $\alpha = 1.1967$ nach (4.6.2) ausreichend ist, um die Konvexität des Unterschätzers

$$F(\mathbf{x}) := f(\mathbf{x}) - \alpha\,(2 - x_1)\,(x_1 + 1) - \alpha\,(1 - x_2)\,(x_2 + 1)$$

Abbildung 4.10 Nichtkonvexe Funktion und ein konvexer Unterschätzer. Der gezeigte konvexe Unterschätzer ist nicht unbedingt der beste Unterschätzer. Bestmöglich ist die konvexe einhüllende Funktion der gegebenen nichtkonvexen Funktion.



zu garantieren.

Eine konvexe Menge M_c heißt Konvexifizierung einer gegebenen Menge M, wenn $M_c \supseteq M$. Die implizit durch die Ungleichungen[21] $\mathbf{g}(\mathbf{x}, \mathbf{y}) \geq 0$ beschriebene zulässige Menge S wird in einem B&B-Verfahren in Teilmengen S^i räumlich geteilt; in Abschnitt 11.1 wird dies anhand eines Beispiels näher erläutert. Über die Konvexifizierung S_c^i wird das konvexe Minimierungsproblem $\min_{(\mathbf{x},\mathbf{y})\in S_c^i} f_c^i(\mathbf{x}, \mathbf{y})$ gelöst und mit Hilfe eines Verfahren, wie z. B. der oben und im Anhang A.4 beschriebenen Äußeren Approximation die Lösung z_c^i berechnet, aus der die untere Schranke $z^u := \min_i z_c^i$ folgt. Durch immer weitere Verfeinerung der Teilung und dadurch bessere Anpassung der konvexen Unterschätzer $f_c^i(\mathbf{x}, \mathbf{y})$ an die Teilung wird so eine monoton wachsende Folge von unteren Schranken z^u erzeugt. Die Verfeinerung wird so lange fortgeführt, bis schließlich $z^o - z^u \leq \varepsilon$ gilt.

Es lässt sich zeigen, dass diese Verfahrensklasse bis auf jede ε-Schranke gegen das globale Minimum konvergiert, allerdings nur in der Theorie, da die unvermeidbaren Rundungsfehler, die bei der praktischen Realisierung eines Verfahrens auf einem Rechner auftreten, dabei nicht berücksichtigt werden; bei der Lösung praktischer Probleme sollte man sich daher möglichst auf Genauigkeiten von $\varepsilon > 10^{-6}$ beschränken. Dass der Einfluss von Rundungsfehlern erheblich sein und schlimme Auswirkungen haben kann, ja sogar zu völlig falschen Aussagen führen kann, ist schon häufig anhand von praktisch relevanten Problemen demonstriert worden. In Bezug auf Rundungsfehler kann es bei der Globalen Optimierung sinnvoll sein, die Ergebnisse mit Methoden der Intervallarithmetik zu überprüfen.

Der zur Lösung von nichtkonvexen NLP- und MINLP-Problemen, die lediglich quadratisch in Constraints und Zielfunktion sind, eignet sich in besonderer Weise der von Misener & Floudas (2012,[213]) entwickelte Solver GLOMIQO (Global Mixed-Integer Quadratic Optimizer). Hier wird das Problem zunächst reformuliert; falls möglich werden Variablen aus Gleichungsbedingungen eliminiert. Beispielsweise wird aus

[21] Gleichungen $\mathbf{h}(\mathbf{x}, \mathbf{y}) = 0$ werden durch das Ungleichungspaar $-\delta \leq \mathbf{h}(\mathbf{x}, \mathbf{y}) \leq \delta$ mit beliebig vorgegebenem $\delta > 0$ ersetzt. Diese Vorgehensweise sollte aber nur eingesetzt werden, wenn es nicht anders geht; besonders effizient ist sie nämlich nicht.

$$Q_{ij}x_ix_j + A_kx_k = B$$

unter der Annahme $A_k \neq 0$ die Variable x_k durch

$$x_k = \frac{Q_{ij}x_ix_j - B}{A_k}$$

im Modell ersetzt; falls untere und obere Schranken X_k^- und X_k^+ für x_k bekannt sind, werden die Ungleichungen

$$X_k^- \leq \frac{Q_{ij}x_ix_j - B}{A_k} \leq X_k^+$$

zum Problem hinzugefügt. Bilineare Terme x_ix_j werden durch Einführung einer zusätzlichen Variablen

$$y_{ij} := x_i + x_j$$

ersetzt und führen auf

$$x_ix_j = \frac{1}{2}\left(y_{ij}^2 - x_i^2 - x_j^2\right) \quad .$$

Zwar enthält das Problem nun drei zusätzlichen quadratische Terme, aber diese enthalten vorteilhafterweise nur eine Variable. Zwei Besonderheiten in GLOMIQO sind die Erzeugung von *RLT-Bedingungen* und eine graphentheoretische Analyse zur Identifikation konvexer Teilstrukturen, die sich positiv im Zusammenhang mit konvexen Unterschätzern auswirken. RLT steht für *Reformulation-Linearization Techniques*; die RLT-Bedingungen werden bei Bedarf dynamisch zum Problem hinzugefügt. GLOMIQO berücksichtigt zum Beispiel alle Variablen x_i, die irgendwo im Modell nichtlinear auftreten, und multipliziert diese mit allen linearen Gleichungsbedingungen mit Zeilenvektor $\mathbf{A}_m$ und Spaltenvektor $\mathbf{x}$

$$\mathbf{A}_m\mathbf{x} = \sum_j A_{mj}x_j = b_m \quad ,$$

die nur kontinuierliche Variablen x_j enthalten, d. h.

$$\sum_j \left(A_{mj}x_j - b_m\right)x_i = \sum_j A_{mj}x_jx_i - b_mx_i = 0 \quad ; \quad \forall\{im\} \quad .$$

Ungleichungsbedingungen

$$b_m^- \leq \mathbf{A}_m\mathbf{x} \leq b_m^+$$

werden ähnlich behandelt und führen auf die zusätzlichen Bedingungen

$$\left(\mathbf{A}_m\mathbf{x} - b_m^+\right)\left(x_i - X_i^-\right) \leq 0 \quad ; \quad \forall\{im\} \quad ,$$

$$\left(\mathbf{A}_m\mathbf{x} - b_m^+\right)\left(X_i^+ - x_i\right) \leq 0 \quad ; \quad \forall\{im\} \quad ,$$

$$\left(b_m^- - \mathbf{A}_m\mathbf{x}\right)\left(x_i - X_i^-\right) \leq 0 \quad ; \quad \forall\{im\} \quad ,$$

$$\left(b_m^- - \mathbf{A}_m\mathbf{x}\right)\left(X_i^+ - x_i\right) \leq 0 \quad ; \quad \forall\{im\} \quad .$$

Hierbei ist zu beachten, dass diese Ungleichungen nicht alle *a priori*, sondern nur bei Bedarf innerhalb eines Branch&Bound Verfahrens erzeugt werden.

5 Die Kunst guter Modellierung

In diesem Kapitel wird die Modellierung verschiedener spezieller Konstrukte vorgestellt, die in praktischen Problemen auftreten. Dazu zählen logische Relationen, die sich mit Hilfe von Binärvariablen beschreiben lassen, aber auch bestimmte nichtlineare Strukturen, die in gemischt-ganzzahlige Formulierungen transformiert werden können. Zudem werden Methoden behandelt, mit denen sich die Rechenzeit ganzzahliger Probleme erheblich reduzieren lässt. Hierzu zählen eine gute Modellbildung, die Verwendung spezieller Verzweigungstechniken und eine Kontrolle der Branch&Bound-Strategien.

Während sich bei der linearen Programmierung die Modellbildung zwar auch positiv auf die Rechenzeit auswirkt, ist bei den meisten praktischen gemischt-ganzzahligen Problemen eine gute Modellbildung für die Lösung geradezu notwendig. Bei MILP-Problemen kann die Qualität und Güte eines Modells anhand der LP-Relaxierung und der konvexen Hülle entschieden werden. Neben einigen Aspekten effizienter Modellbildung, die vom Modellentwickler beeinflusst werden können, werden in diesem Kapitel automatische Reformulierungsverfahren [283] für Optimierungsprobleme mit binären Variablen und *Preprocessing*-Techniken behandelt, die zu schärferen Modellformulierungen führen.

In diesem Kapitel wird betont, wie wichtig es ist, Modellstrukturen mit Bedacht zu wählen, oder spezielle Strukturen wie *Cliquen* und *Covers* zu erkennen, Modelle zu reformulieren und daraufhin zu untersuchen, wie sich bestimmte gemischt-ganzzahlige Ungleichungen verschärfen lassen. Durch Presolving, sinnvolle Skalierung und strukturausnutzende Verzweigungsregeln für SOS-1- und SOS-2-Mengen sowie halbstetige Variablen kann die Qualität der Lösung und die Rechenzeit erheblich verbessert werden.

5.1 Modellierung logischer Zusammenhänge

In manchen Fällen ist es sinnvoll, verschiedene Teile eines Modells mit Hilfe der Symbole der mathematischen Logik darzustellen, die sich dann wiederum in algebraische *Nebenbedingungen* übersetzen lassen. Zu diesen Relationen der mathematischen Logik zählen die Disjunktion ODER ($\vee$), die Konjunktion UND ($\wedge$), die Negation ($\neg$) bzw. die Relation NICHT [engl.: NOT], die *Implikation* ($\Longrightarrow$), und die *Äquivalenz* ($\Longleftrightarrow$). Mit Hilfe dieser Relationen lassen sich mehrere logische Ausdrücke L_i, verknüpfen, wie z. B. „wenn L_1 ODER L_2, dann L_3"

$$L_1 \vee L_2 \Rightarrow L_3 \quad .$$

Die Nützlichkeit dieses Konzepts lässt sich anhand einer zweistufigen chemischen Produktionsanlage demonstrieren, die Vorprodukte und Verkaufsprodukte herstellt und dabei der Regel folgt: wenn eines der Verkaufsprodukte S_1, S_2 oder S_3 produziert wird, dann muss mindestens eines der Vorprodukte P_1 oder P_2 hergestellt werden. Mit Hilfe der *logischen Variablen*[1] L_1^S, L_2^S, L_3^S, L_1^P und L_2^P kann diese Regel in die Form

[1] Logische Variablen können nur die Werte wahr (W) und falsch (F) [engl.: *true* und *false*, bzw. T und F] annehmen.

J. Kallrath, *Gemischt-ganzzahlige Optimierung: Modellierung in der Praxis*,
DOI 10.1007/978-3-658-00690-7_5, © Springer Fachmedien Wiesbaden 2013

$$(L_1^S \vee L_2^S \vee L_3^S) \Longrightarrow (L_1^P \vee L_2^P) \tag{5.1.1}$$

gebracht werden. Logische Ausdrücke lassen sich mit Hilfe der *DeMorgan-Gesetze*

$$\neg(L_1 \vee L_2) \Leftrightarrow \neg L_1 \wedge \neg L_2 \quad , \quad \neg(L_1 \wedge L_2) \Leftrightarrow \neg L_1 \vee \neg L_2 \tag{5.1.2}$$

in äquivalente Ausdrücke umformulieren, die dann vielleicht einfacher zu überschauen sind. Nützlich sind auch die Distributivgesetze

$$L_1 \vee (L_2 \wedge L_3) \quad \Leftrightarrow \quad (L_1 \vee L_2) \wedge (L_1 \vee L_3) \tag{5.1.3}$$

$$L_1 \wedge (L_2 \vee L_3) \quad \Leftrightarrow \quad (L_1 \wedge L_2) \vee (L_1 \wedge L_3) \tag{5.1.4}$$

und die Äquivalenz

$$(L_1 \vee L_2) \Longrightarrow L_3 \quad \Leftrightarrow \quad (L_1 \Longrightarrow L_3) \wedge (L_2 \Longrightarrow L_3) \quad . \tag{5.1.5}$$

Eine Vertiefung erfahren derartige logische Zusammenhänge beim Erfüllbarkeitsproblem [engl.: *satisfibility problem*] in Abschnitt 5.1.5.

5.1.1 Ganzzahlige Variablen und logische Bedingungen

Kann eine Maschine sich jeweils in genau einem von sechs Zuständen befinden, so könnte man versucht sein, dies mit einer ganzzahligen Variablen α, die der Bedingung $1 \leq \alpha \leq 6$ genügt, abzubilden; $\alpha = i$ bedeutet dann, dass der Zustand i gewählt ist. Diese Modellformulierung ist jedoch unpraktisch, da vielleicht zusätzlich noch Implikationen

$$\alpha = 3 \Longrightarrow \ldots, \quad \alpha \neq 3 \Longrightarrow \ldots, \quad \alpha = 5 \Longrightarrow \ldots, \quad \alpha \neq 5 \Longrightarrow \ldots \quad .$$

der Art „wenn Zustand 3 oder 5 gewählt ist, dann ....” etc. zu berücksichtigen sind. Wesentlich eleganter ist es, das Problem mit binären Variablen δ_i zu formulieren, die wie folgt definiert sind

$$\delta_i = \begin{cases} 1 \text{ , wenn die Maschine im Zustand } i \text{ ist} \\ 0 \text{ , sonst} \end{cases} \quad , \quad i = 1, 2, ..., 6$$

und mit Hilfe der Bedingung

$$\sum_{i=1}^{6} \delta_i = 1 \tag{5.1.6}$$

verknüpft sind, die garantiert, dass die Maschine genau einen Zustand annehmen muss.

Jetzt lassen sich die Implikationen, z. B.

$$\delta_3 = 1 \Longrightarrow \text{„es wird zusätzliches Rohmaterial benötigt”}$$

leicht in der Form

$$\begin{aligned} \delta_3 &= 1 \Longrightarrow \ldots, \quad \delta_3 = 0 \Longrightarrow \ldots \\ \delta_5 &= 1 \Longrightarrow \ldots, \quad \delta_5 = 0 \Longrightarrow \ldots \end{aligned}$$

formulieren.

Wir halten fest, dass ganzzahlige Variablen selten gebraucht werden, um logische Relationen, sondern eher um zählbare Größen wie z. B. Container, Mitarbeiter, etc. abzubilden. Sollte der Name und die Zahl des Zustandes explizit benötigt werden, so kann die Variable α mit Hilfe von

$$\alpha = \sum_{i=1}^{6} i\delta_i$$

mit den Binärvariablen verknüpft werden.

5.1.2 Transformation logischer Aussagen in arithmetische Ausdrücke

Angenommen, eine Produktionsanlage kann keines oder einige bzw. alle Produkte aus einer Liste von n Produkten herstellen. Zur Beschreibung dieser Situation sei die logische Variable L_p eingeführt, die den Wert *wahr* annimmt, wenn wir entscheiden, das Produkt p (positive Entscheidung) zu produzieren, und *falsch*, wenn wir entscheiden, p nicht zu produzieren (negative Entscheidung). Die Produktion der einzelnen Produkte mag voneinander abhängen oder nicht. Mit den logischen Variablen führen wir assoziierte Binärvariablen $\delta_1, \delta_2, \ldots, \delta_n$

$$\delta_p := \begin{cases} 1\ , p \text{ wird produziert, d. h., } L_p = \mathit{wahr} \\ 0\ , p \text{ wird nicht produziert, d. h., } L_p = \mathit{falsch} \end{cases}$$

ein. Die Werte *wahr* und *falsch*, die von den logischen Variablen angenommen werden können, entsprechen in natürlicher Weise den Werten 1 und 0 der binären Variablen. Letztere bieten aber den Vorteil, dass sie die gewohnten arithmetischen Operationen erlauben. So repräsentiert z. B. der Ausdruck

$$\delta_1 + \delta_2 + \delta_3 + \delta_4$$

eine Größe, die jeden Wert zwischen 0 und 4 annehmen kann, und Auskunft darüber gibt, wie viele positive Entscheidungen getroffen bzw. wie viele Produkte hergestellt wurden.

Im Folgenden wird eine Übersetzung zwischen diversen logischen und arithmetischen Ausdrücken gegeben. Sei δ eine Binärvariable, die die Entscheidung oder logische Variable L repräsentiert, dann ist die Negation $\neg L$ durch

$$\delta' = 1 - \delta$$

gegeben. Soll L der Wert *falsch* zugewiesen werden, so lautet die entsprechende algebraische Darstellung $\delta = 0$.

5.1.3 Logische Ausdrücke mit zwei Argumenten

Logische Ausdrücke, die zwei Variablen L_1 und L_2, von denen jede die Werte *wahr* und *falsch* annehmen kann, verknüpfen, lassen sich algebraisch in der Form

Verknüpfung		Nebenbedingung
ODER	$L_1 \vee L_2$	$\delta_1 + \delta_2 \geq 1$
UND	$L_1 \wedge L_2$	$\delta_1 + \delta_2 = 2$
Negation (UND)	$\neg(L_1 \wedge L_2)$	$\delta_1 + \delta_2 \leq 1$
Negation (ODER)	$\neg(L_1 \vee L_2)$	$\delta_1 = 0, \delta_2 = 0$
Implikation	$L_1 \Longrightarrow L_2$	$\delta_1 \leq \delta_2$
Äquivalenz	$L_1 \Longleftrightarrow L_2$	$\delta_1 = \delta_2$

(5.1.7)

darstellen. Die Übersetzungen lassen sich mit Hilfe aller Kombinationen der Variablen δ_1 und δ_2 und der entsprechenden Wahrheitstabelle überprüfen. Als Beispiel hierzu sei die oder-Verknüpfung betrachtet; an die beiden logischen Variablen L_1 und L_2 soll die Bedingung gestellt sein, dass $L_1 \vee L_2$ *wahr* wird. Die nachfolgende Wahrheitstabelle

L_1	L_2	$L_1 \vee L_2$	δ_1	δ_2	$\delta_1 + \delta_2 \geq 1$
w	f	w	1	0	erfüllt; $1 \geq 1$
w	w	w	1	1	erfüllt; $2 \geq 1$
f	w	w	0	1	erfüllt; $1 \geq 1$
f	f	f	0	0	verletzt; $0 \geq 1$!!

beweist die Richtigkeit der Übersetzung, die Ungleichung $\delta_1 + \delta_2 \geq 1$ ist genau dann erfüllt, wenn mindestens eine der beiden Binärvariablen den Wert 1 hat, also L_1 oder L_2 *wahr* ist, d. h. $\delta_1 + \delta_2 \geq 1$ erzwingt wie gewünscht, dass mindestens eine Binärvariable den Wert annimmt bzw. mindestens eine der Entscheidungen L_1 oder L_2 akzeptiert wird, also möglicherweise auch beide. Soll erzwungen werden, dass nur eine Entscheidung akzeptiert wird (*exklusives oder*), so wird man die Bedingung $\delta_1 + \delta_2 = 1$ wählen.

Hier wird einmal mehr deutlich, dass während der Modellierungsarbeit und in Zusammenarbeit mit einem Auftraggeber auf subtile Feinheiten zu achten ist. Das Wort *oder* wird umgangssprachlich eher im Sinne von „*mindestens eines von beiden*" verwendet und kann bei Übersetzung in ein Modell zu erheblichen Unterschieden führen, wenn es eigentlich in der Bedeutung *eines von beiden* (*exklusives oder*) verwendet werden soll.

Die Konjunktion von L_1 und L_2, $L_1 \wedge L_2$ (d. h. „$L_1 = w$ *und* $L_2 = w$") führt zu

$$\begin{aligned} \delta_1 &= 1 \\ \delta_2 &= 1 \end{aligned}$$

oder, kürzer gesagt,

$$\delta_1 + \delta_2 = 2 \quad . \tag{5.1.8}$$

Gleichung (5.1.8), hier der Vollständigkeit wegen aufgeführt, mag als überflüssig erscheinen, da sie trivialerweise $\delta_1$und δ_2 zwingt, die Werte 1 anzunehmen und somit die direkte Eliminierung dieser Variablen möglich wäre.

Wie die Übersetzung von $\wedge$ und $\vee$ zeigt, liefert die Summe einiger Binärvariablen eine Größe, die angibt, wie viele positive Entscheidungen realisiert wurden. Teilmengen von $\{1, 2, ..., n\}$ können so ausgewählt werden, dass sie spezielle Gruppen von Entscheidungen verknüpfen. So repräsentiert z. B. $\delta_3 + \delta_6$ zwei gleichzeitig anstehende Entscheidungen.

Die umgekehrten Bedingungen zu $\vee$ und $\wedge$, nämlich $\neg(L_1 \vee L_2)$ und $\neg(L_1 \wedge L_2)$ lassen sich mit Hilfe der DeMorgan Gesetze (5.1.2) formulieren. Wegen $\neg(L_1 \wedge L_2) = \neg L_1 \vee \neg L_2$ und anschließender Verwendung des in (5.1.7) gegebenen Ausdrucks für ODER gilt

$$1 - \delta_1 + 1 - \delta_2 \geq 1$$

bzw.

$$\delta_1 + \delta_2 \leq 1 \quad .$$

Der Ausdruck $\neg(L_1 \vee L_2) = \neg L_1 \wedge \neg L_2$ wird in gleicher Weise behandelt.

Die soweit entwickelten Relationen erlauben es, komplizierte Beispiele aus einfacheren zu entwickeln. Gewöhnlich ist es sinnvoll, zur Darstellung von $\neg L$ eine Variable, z. B. δ' einzuführen, die später wieder durch $\delta' = 1 - \delta$ substituiert wird. Beispielsweise kann die Aussage „ist Entscheidung 1 positiv, dann ist Entscheidung 2 negativ", in der Form

$$\delta_1 \leq \delta_2' \tag{5.1.9}$$

übersetzt werden, wobei δ_2' mit $\neg L_2$ korrespondiert. Daraufhin (5.1.9) kann als

$$\delta_1 \leq 1 - \delta_2 \quad \text{oder} \quad \delta_1 + \delta_2 \leq 1$$

umgeschrieben werden. Man beachte, dass dieser Ausdruck derselbe ist, den wir für die Modellierung von $\neg(L_1 \wedge L_2)$ verwendet haben, der wiederum ein äquivalenter Ausdruck für $\neg L_1 \vee \neg L_2$ ist. Die logische Äquivalenz wird folgendermaßen ersichtlich: $\delta_1 = 1$ impliziert $\delta_2 = 0$. Ist jedoch $\delta_2 = 1$, dann folgt aus (5.1.9) sofort $\delta_1 = 0$, was nahelegt, dass wir auch die Aussage „wenn die Entscheidung 2 positiv ausfällt, kann die Entscheidung 1 nicht positiv ausfallen" modellieren. Dies erscheint vernünftig, denn wenn Entscheidung 2 gewählt wurde, dann müssen wir verhindern, dass Entscheidung 1 ebenfalls getroffen wird, da sonst ein widersprüchliches Ergebnis, nämlich die Wahl von Entscheidung 2 nicht genommen werden kann. Dies lässt sich auch mit Hilfe der entsprechenden Wahrheitstabelle überprüfen.

5.1.4 Multivariate logische Ausdrücke

Die grundlegende Idee zur Umformung logischer Ausdrücke mit mehr als zwei Argumenten in arithmetische Relationen besteht in der Rückführung auf logische Ausdrücke mit zwei Variablen.

Soll der logische Ausdruck $L_1 \wedge (L_2 \vee L_3)$ den Wert *wahr* annehmen, so wird dies durch die Bedingungen

$$\delta_1 = 1 \quad , \quad \delta_2 + \delta_3 \geq 1$$

sichergestellt. Um den Ausdruck $L_1 \vee (L_2 \wedge L_3)$ zu modellieren, verwenden wir zuerst (5.1.3) und erhalten $(L_1 \vee L_2) \wedge (L_1 \vee L_3)$. Dies führt uns unverzüglich zu

$$\delta_1 + \delta_2 \geq 1 \quad , \quad \delta_1 + \delta_3 \geq 1 \quad . \tag{5.1.10}$$

Durch Addition der beiden Ungleichungen (5.1.10) folgt die Ungleichung

$$2\delta_1 + \delta_2 + \delta_3 \geq 2 \quad , \tag{5.1.11}$$

welche ebenfalls den ursprünglichen logischen Ausdruck $L_1 \vee (L_2 \wedge L_3)$ darstellt; dies zeigt uns, dass es im allgemeinen keine eindeutige Transformation gibt.

Zu beachten ist jedoch, dass (5.1.10) „schärfer" ist als (5.1.11). Hinsichtlich aller ganzzahligen Lösungen sind (5.1.11) und (5.1.10) äquivalent, aber (5.1.11) kann nichtganzzahlige Lösungen enthalten, die nicht für (5.1.10) gültig sind. Um dies zu sehen, betrachte man die LP-Relaxierung von (5.1.10) und (5.1.11). Während $(\delta_1, \delta_2, \delta_3) = (0.8, 0.3, 0.1)$ (5.1.11) erfüllt, ist die zweite Ungleichung von (5.1.10) verletzt; (5.1.10) eliminiert also mehr nichtganzzahlige Kombinationen von Variablen. Da (5.1.11) aus Addition der beiden Ungleichungen in (5.1.10) erzeugt wurde, kann es umgekehrt nicht sein, dass es nichtganzzahlige Lösungen für (5.1.10) gibt, die aber ganzzahlig für (5.1.11) sind.

Nun sind wir in der Lage, den logischen Ausdruck (5.1.1) für das erwähnte zweistufige Produktionsbeispiel aus der Chemie in einen arithmetischen Ausdruck zu transformieren. Zunächst wenden wir (5.1.5) an und erhalten

$$(L_1^S \Longrightarrow L) \wedge (L_2^S \Longrightarrow L) \wedge (L_3^S \Longrightarrow L) \quad ,$$

wobei wir $L_1^P \vee L_2^P$ durch L mit $L \Longrightarrow L_1^P \vee L_2^P$ ersetzen. Assoziieren wir nun eine Binärvariable δ mit L, so folgen aus Tabelle (5.1.7) die Ungleichungen

$$\delta \leq \delta_1^P + \delta_2^P \quad , \quad \delta_1^S \leq \delta \quad , \quad \delta_2^S \leq \delta \quad , \quad \delta_3^S \leq \delta \quad .$$

Zum Abschluss dieses Abschnitts seien noch einige nützliche Transformationen aufgelistet, die drei Variablen enthalten:

Relationen	Nebenbedingungen
$L_1 \Longrightarrow (L_2 \wedge L_3)$	$\delta_1 \leq \delta_2 \quad , \quad \delta_1 \leq \delta_3$
$L_1 \Longrightarrow (L_2 \vee L_3)$	$\delta_1 \leq \delta_2 + \delta_3$
$(L_1 \wedge L_2) \Longrightarrow L_3$	$\delta_1 + \delta_2 \leq 1 + \delta_3$
$(L_1 \vee L_2) \Longrightarrow L_3$	$\delta_1 \leq \delta_3 \quad , \quad \delta_2 \leq \delta_3$
$L_1 \wedge (L_2 \vee L_3)$	$\delta_1 = 1 \quad , \quad \delta_2 + \delta_3 \geq 1$
$L_1 \vee (L_2 \wedge L_3)$	$\delta_1 + \delta_2 \geq 1 \quad , \quad \delta_1 + \delta_3 \geq 1$

. (5.1.12)

Schließlich seien noch einige allgemeinere logische Ausdrücke mit n Variablen betrachtet. Mit den Bezeichnungen

$$\bigvee_{i=1}^{n} L_i := L_1 \vee L_2 \vee ... \vee L_n \quad , \quad \bigwedge_{i=1}^{n} L_i := L_1 \wedge L_2 \wedge ... \wedge L_n$$

lassen sich die folgenden Transformationen ableiten:

Disjunktion	$\bigvee_{i=1}^{n} L_i$	:	$\sum_{i=1}^{n} \delta_i \geq 1$
Konjunktion	$\bigwedge_{i=1}^{n} L_i$	:	$\sum_{i=1}^{n} \delta_i = n$
Implikation	$\bigwedge_{i=1}^{k} L_i \Longrightarrow \bigvee_{i=k+1}^{n} L_i$	:	$\sum_{i=k+1}^{n} \delta_i - \sum_{i=1}^{k} \delta_i \geq 1 - k$
	mindestens k aus n	:	$\sum_{i=1}^{n} \delta_i \geq k$
	genau k aus n	:	$\sum_{i=1}^{n} \delta_i = k$
	höchstens k aus n	:	$\sum_{i=1}^{n} \delta_i \leq k$

. (5.1.13)

Schließlich verallgemeinern wir einige Relationen, die den Äquivalenz-Operator $\Longleftrightarrow$ und $n+1$ Variablen umfassen. Um den Ausdruck $L_{n+1} \Longleftrightarrow \bigvee_{i=1}^{n} L_i$ zu modellieren, verwenden wir die Ungleichungen

$$\sum_{i=1}^{n} \delta_i \geq \delta_{n+1} \quad , \quad \delta_{n+1} \geq \delta_k \quad , \quad k = 1, \ldots, n \quad .$$

Der Ausdruck $L_{n+1} \Longleftrightarrow \bigwedge_{i=1}^{n} L_i$ ist darstellbar durch die Ungleichungen

$$-\sum_{i=1}^{n} \delta_i + \delta_{n+1} \geq 1 - n \quad , \quad \delta_{n+1} \leq \delta_k \quad , \quad k = 1, \ldots, n \quad .$$

Zu beachten ist, dass bei der Modellierung logischer Ausdrücke diese in die Form

$$A \quad \text{UND} \quad B \quad \text{UND} \quad C \quad .....$$

gebracht werden müssen, weil derartige Ausdrücke in eine Reihe von Ungleichungen in Binärvariablen übersetzt werden, die *alle* erfüllt werden müssen. Auf diese Weise transformiert sich unser Modell in

Nebenbedingung $'A'$ UND Nebenbedingung $'B'$ UND Nebenbedingung $'C'$ ..... .

5.1.5 Das Erfüllbarkeitsproblem

Das z. B. in Papadimitriou & Steiglitz (1982,[230], S. 314ff) beschriebene Erfüllbarkeitsproblem [engl.: *satisfiability problem, SAT-Problem*] ist das erste kombinatorische Problem, für das die $\mathcal{NP}$-Vollständigkeit[2] nachgewiesen wurde, d. h. es wurde gezeigt, dass, wenn man das SAT-Problem in polynomialer Zeit lösen kann, damit alle Probleme in $\mathcal{NP}$ in polynomialer Zeit löst. Jedes andere Problem in $\mathcal{NP}$ kann polynomial auf das SAT-Problem abgebildet werden; siehe z. B. Garey & Johnson (2000,[98]). Das 3-SAT-Problem ist eine immer noch $\mathcal{NP}$-vollständige Einschränkung des SAT-Problems, bei dem nun jede Bedingung aus drei logischen Elementartermen [engl.: *literal*] besteht.

Zur Veranschaulichung dieses Problems kann man eine Gesetzesvorschrift betrachten, die regelt, ob ein bestimmter Mensch in einem Land einwanderungsberechtigt ist. Er ist einwanderungsberechtigt, wenn er eine Reihe von Voraussetzungen erfüllt, wobei sich die Voraussetzungen zum Teil gegenseitig ausschließen. Zu beantworten ist die Frage, ob es eine Kombination von Voraussetzungen gibt, die eine Einwanderung ermöglichen, ob also alle Bedingungen erfüllbar sind. Eine ähnliche Fragestellung findet sich in Küchlin & Sinz (2000,[186]) bei der Kaufauswahl und Produktion von Kfz-Modellen, die komponentenweise definiert werden oder bei einem schwerkranken Patienten, hinsichtlich einer anstehenden operativen Behandlung und der Verträglichkeit verschiedener Maßnahmen und Verabreichung von Medikamenten. Schließlich, und das überrascht nicht sehr, eignet sich das Erfüllbarkeitsproblem beim Design elektrischer Schaltkreise. An den Eingängen des Schaltkreises kann man eine bestimmte Spannung anlegen ODER dies nicht tun. So ergeben sich die Voraussetzungen bzw. Eingangswerte W (wahr) oder F (falsch). Die elektrischen Bauteile des Schaltkreises verhalten sich wie logische Schaltelemente (Gatter) und unterstützen die Operationen UND, ODER und NICHT. Jedes Gatter wertet mehrere Eingaben in bestimmter Weise aus und liefert in seinem Ausgang ein Ergebnis, das vielleicht wiederum als Eingang in einem nachgeschalteten Gatter verwendet wird. Ein NICHT-Gatter wandelt so z. B. die Eingabe W in F, F dagegen in W um. Ein UND-Gatter dagegen liefert nur das Ergebnis W, wenn beide Eingänge W sind. Der gesamte Schaltkreis hat einen Ausgang, und die entscheidende Frage im Zusammenhang mit dem SAT-Problem ist, ob es zu diesem Schaltkreis ein Kombination von Eingaben gibt, sodass das Ausgangsergebnis W sein kann. Es zeigt sich, dass es sehr schwierig wird, diese Frage zu beantworten, wenn der Schaltkreis viele Gatter hat. Abschließend empfohlen seien die Artikel von Biere *et al.* (1999,[36]) aus dem Gebiet der Hardware-Verifikation, Lynce & Marques-Silva (2006,[200]) mit einer Anwendung aus der Bioinformatik sowie Marques-Silva (2008,[17]) und seinen Überblick über Anwendungen des SAT-Problems.

Im SAT-Problem soll also überprüft werden, ob es für einen gegeben logischen Ausdruck passende Wertzuweisungen zu den logischen Variablen gibt, sodass dieser logische Ausdruck den Wert W annimmt. Sei $X = \{x_1, x_2, ..., x_n\}$ eine Menge von n logischen bzw. binären Variablen. Ein logischer Elementarterm y_i ist entweder eine Variable x_i oder deren Negation $\bar{x}_i$. Eine Bestimmung [engl.: *clause*] L_j ist eine Disjunktion logischer Elementarterme, z. B. $x_1 \vee x_2 \vee \bar{x}_3$. Die Formel $L = L_1 \wedge L_2 \wedge ... \wedge L_m$ stelle eine Konjunktion von m Bestimmungen dar. Die L ist genau dann erfüllbar, wenn es eine Abbildung $t : X \to \{0, 1\}^n$ gibt, die simultan allen Bestimmungen L_j in L genügt, d. h. es soll entschieden werden, ob für eine gegebene Instanz (X, L) eine Wahrheitswertzuordnung für X existiert, die L erfüllt. Im weiteren Verlauf nehmen wir an, dass jede

[2] Die Begriffe der Komplexitätstheorie wie z. B. $\mathcal{NP}$-Vollständigkeit oder polynomiale Algorithmen sind auf Seite 303 erklärt.

logische Formel in konjunktiver Normalform vorliegt, d. h. sie ist eine Konjunktion von Disjunktionstermen. Enthält jede Bestimmung höchstens k logische Elementarausdrücke, so wird das Problem k-SAT-Problem genannt. Das 2-SAT-Problem kann in polynomialer Zeit gelöst werden. In einer mathematisch-algebraischer Formulierung identifizieren wir die Wahrheitswerte W und F mit 1 und 0. Dann ist das Komplement $\bar{x}_i$ von x_i einfach $1 - x_i$ und die Disjunktion erscheint als Addition. Daher verlangen wir nun, dass in jeder Bestimmung L_j mindestens ein wahrer logischer Elementarterm existiert, d. h.:

$$\sum_{x_i \in L_j} x_i + \sum_{\overline{x}_i \in L_j} (1 - x_i) \geq 1 \quad , \quad j = 1, ..., m \quad .$$

Damit kann das Problem nun wie folgt beschrieben werden. Bestimme x derart, dass gilt:

$$\sum_{i \in L_j^+} x_i - \sum_{i \in L_j^-} x_i \geq 1 - |L_j^-| \quad , \quad j = 1, ..., m$$

$$x_i \in \{0, 1\} \quad , \quad i = 1, ..., n \quad ,$$

wobei L_j^+ die Menge der positiven logischen Elementarterme und L_j^- die Menge ihrer Komplemente für die Bestimmung L_j ist.

Allgemeine SAT-Probleme lassen sich zwar mit Hilfe ganzzahliger Modelle beschreiben. Für SAT-Probleme gibt es spätestens seit dem Verfahren von Davis & Putnam (1960,[65]) effizientere Lösungsverfahren als die ganzzahlige Optimierung; als Einstieg mag Hölldober *et al.* (2011,[134]) mit vielen weiteren Literaturhinweisen dienen.

5.2 Logische Bedingungen auf Nebenbedingungen

Um logische Operationen auf Nebenbedingungen $\mathbf{Ax} \circ \mathbf{B}$ anzuwenden, sind zunächst untere und obere Schranken von $\mathbf{Ax} - \mathbf{B}$ zu bestimmen [siehe z. B. Brearley *et al.* (1975,[44]) oder McKinnon & Williams (1989,[207]) für eine tiefgehendere Behandlung]. Einführend sei das folgende Beispiel betrachtet, wobei logische Operationen auf Relationen der Form

$$A_1 x_1 + A_2 x_2 + ... + A_n x_n \circ B \tag{5.2.1}$$

anwendet werden sollen, die zunächst als

$$A_1 x_1 + A_2 x_2 + ... + A_n x_n - B \circ 0$$

umgeschrieben werden, wobei $\circ$ die Relationen $=, \leq,$ oder $\geq$ darstellt, und die Variablen $x_1, x_2, \ldots, x_n$ entweder kontinuierlich oder ganzzahlig und mit unteren und oberen Schranken L_i bzw. U_i versehen sind. Unter der Voraussetzung $A_i \geq 0$ folgen obere (untere) Schranken von $\mathbf{Ax} - \mathbf{B}$ zu

$$U = \sum_{i=1}^{n} A_i U_i - B \quad , \quad L = \sum_{i=1}^{n} A_i L_i - B \quad ;$$

im Falle $A_i < 0$ müssen U_i und L_i lediglich vertauscht werden. Nun können einige Spezialfälle betrachtet werden.

5.2.1 Logische Bedingungen auf einzelnen Variablen

Die Bedingungen „fällt Entscheidung 1 positiv aus, dann muss die Variable x größer sein als irgendeine positive Konstante C" und „wenn Entscheidung 1 negativ ausfällt, dann ist die Variable x gleich 0" werden durch die Ungleichungen

$$x \geq C\delta \quad , \quad x \leq U\delta$$

modelliert, wobei U eine obere Schranke von x ist.

In der ersten Ungleichung nehmen wir an, dass $C > L$ ist, wobei L eine untere Schranke für x ist, andernfalls ist die Ungleichung $x \geq C\delta$ redundant. Um die Ungleichung $x \leq U\delta$ zu verstehen, sei das folgende Beispiel betrachtet.

Wenn die Produktion von *Xyrene* in einer Schicht beginnt, fallen Rüstkosten in Höhe von £1000 an, unabhängig davon, wieviel *Xyrene* produziert wird. Die maximale Menge von *Xyrene*, die in einer Schicht produziert werden kann, beträgt 500 Tonnen. Die variablen Produktionskosten von *Xyrene* betragen £100/Tonne. Die gesamten Produktionskosten C lassen sich nun mit Hilfe der binären Variable δ,

$$\delta = \begin{cases} 1 \text{ , wenn die Produktion Xyrene in einer Schicht gestartet wird} \\ 0 \text{ , andernfalls} \end{cases}$$

und der kontinuierlichen Variable x darstellen, die die produzierte Menge von *Xyrene* beschreibt. Damit ergeben sich die Nebenbedingungen

$$x \leq 500\delta \quad , \quad C = 100x + 1000\delta \quad .$$

5.2.2 Logische Bedingungen auf Nebenbedingungen

Der Ausdruck „falls Entscheidung 1 getroffen wird, dann muss $\sum_{i=1}^{n} A_i x_i \leq B$ eingehalten werden", wird modelliert als

$$\sum_{i=1}^{n} A_i x_i - B \leq U(1-\delta) \quad . \tag{5.2.2}$$

Ähnlich wird der Ausdruck „falls Entscheidung 1 gewählt wird, muss $\sum_{i=1}^{n} A_i x_i \geq B$ eingehalten werden" durch die Ungleichung

$$\sum_{i=1}^{n} A_i x_i - B \geq L(1-\delta) \quad . \tag{5.2.3}$$

modelliert. Soll dagegen bei Wahl der Entscheidung 1 die Gleichung $\sum_{i=1}^{n} A_i x_i = B$ eingehalten werden, so werden zwei Ungleichungen

$$\sum_{i=1}^{n} A_i x_i - B \leq U(1-\delta) \quad , \quad \sum_{i=1}^{n} A_i x_i - B \geq L(1-\delta)$$

benötigt. Zu beachten ist, dass $\delta = 1$ die Ungleichungen

$$\sum_{i=1}^{n} A_i x_i - B \leq 0 \quad , \quad \sum_{i=1}^{n} A_i x_i - B \geq 0$$

impliziert, was, wie gefordert, äquivalent zu $\sum_{i=1}^{n} A_i x_i = B$ ist. Der Fall $\delta = 0$ führt zu

$$\sum_{i=1}^{n} A_i x_i - B \leq U \quad , \quad \sum_{i=1}^{n} A_i x_i - B \geq L$$

was, in Einklang zu der Definition von L und U, immer erfüllt ist.

Dazu nun ein Beispiel: Eine Ölfirma erforscht ein neues Gebiet im Hinblick auf geeignete Bohrplätze. Das Gebiet ist in sechs Teilgebiete eingeteilt, in denen mit gleicher Wahrscheinlichkeit Öl vermutet wird. In naher Zukunft wird eine Klassifikation über die seismologische Aktivität des Gebietes erwartet, die klärt, ob es eine gute, mittelgute, oder schlechte Aussicht hat, ein geeignetes Erdölfördergebiet zu werden.

Wenn die Angabe lautet, dass das Gebiet eine schlechte Aussicht hat, wird die Firma Bohrplätze auf wenigstens einem oder höchstens zwei Teilgebieten aufnehmen. Bei mittelguten Aussicht sollen Bohrplätze für mindestens zwei und höchstens vier, bei guten Aussicht wenigstens drei und höchstens fünf Teilgebiete aufgenommen werden.

Das Problem wird wie folgt dargestellt. Seien δ_1, δ_2 und δ_3 Binärvariablen, die über schlechte, mittelgute und gute Aussicht entscheiden, z. B.,

$$\delta_{1(2,3)} = \begin{cases} 1 \text{ , wenn das Gebiet eine schlechte (mittelgute, gute) Aussicht hat} \\ 0 \text{ , andernfalls} \end{cases}$$

und für $i = 1, 2, ..., 6$

$$\alpha_i = \begin{cases} 1 \text{ , wenn Bohrtätigkeit in Teilgebiet } i \text{ aufgenommen werden soll} \\ 0 \text{ , andernfalls} \end{cases} .$$

Die Gleichung

$$\delta_1 + \delta_2 + \delta_3 = 1$$

besagt, dass genau eine der Aussichten zutreffen wird. Die Forderung, dass wenigstens in einem Gebiet die Bohrtätigkeit aufgenommen wird, wenn die Aussichten schlecht sind, kann durch

$$\sum_{i=1}^{6} \alpha_i \geq \delta_1$$

abgebildet werden. Ist die Aussicht schlecht, folgt $\delta_1 = 1$ und wenigstens eine Bohrtätigkeit wird aufgenommen; andernfalls ist die Ungleichung nicht aktiv, da auch der Fall eintreten kann, dass gar keine Bohrtätigkeit aufgenommen wird. Die obere Grenze (höchstens zwei von sechs) wird modelliert als

$$\sum_{i=1}^{6} \alpha_i - 2 \leq 4(1 - \delta_1) \quad ,$$

welche als

$$\sum_{i=1}^{6} \alpha_i \leq 6 - 4\delta_1$$

umgeschrieben werden kann.

Ist $\delta_1 = 1$, dann ist die Anzahl der aufgenommenen Bohrungen höchstens zwei, andernfalls ist die Ungleichung nicht aktiv, da die Maximalanzahl der aufgenommenen Möglichkeiten sechs sein könnte.

Die anderen Fälle können ähnlich modelliert werden: für mittelgute Aussicht folgt $\delta_2 = 1$; somit erzwingt die Ungleichung

$$\sum_{i=1}^{6} \alpha_i \geq 2\delta_2 \quad ,$$

dass mindestens zwei Bohrungen aufgenommen werden, andernfalls, d. h. für $\delta_2 = 0$ ist die Ungleichung nicht aktiv, da die Minimalanzahl der aufgenommenen Bohrungen Null sein darf. Andererseits impliziert im Falle $\delta_2 = 1$ die Ungleichung

$$\sum_{i=1}^{6} \alpha_i \leq 6 - 2\delta_2 \quad ,$$

dass die Anzahl der aufgenommenen Bohrungen höchstens vier sein kann, andernfalls ist die Ungleichung nicht aktiv, da die Maximalanzahl der aufgenommenen Bohrungen sechs sein könnte. Schließlich führt der Fall guter Aussichten, d. h. $\delta_3 = 1$, mit mindestens drei und maximal fünf Bohrungen auf die Bedingung

$$3\delta_3 \leq \sum_{i=1}^{6} \alpha_i \leq 6 - \delta_3 \quad .$$

5.2.3 Disjunktive Mengen von Implikationen

Disjunktive Ausdrücke der Form „fällt Entscheidung 3 negativ aus, dann können auch die Entscheidungen 1 und 2 nicht positiv ausfallen", können durch die Ungleichungen

$$\delta_1 + \delta_2 \leq 2\delta_3 \tag{5.2.4}$$

modelliert werden. Unter numerischen Gesichtspunkten ist es jedoch vorteilhafter, statt dessen die beiden Ungleichungen

$$\delta_1 \leq \delta_3 \quad , \quad \delta_2 \leq \delta_3 \tag{5.2.5}$$

zu verwenden. Die zweite Formulierung schließt mehr fraktionale Werte für die Binärvariablen in einer LP-Relaxierung aus und ist in diesem Sinne *schärfer* [engl.: *tighter*]; siehe Jeroslow & Lowe (1984, 1985; [148], [149]) für weitere Details. Während im Falle $\delta_3 = 0.5$, δ_1 und δ_2 in (5.2.4) einzeln noch die Werte 0 und 1 annehmen bzw. in der LP-Relaxierung die Werte $[0, 1] \times [0, 1]$ annehmen können, erlaubt (5.2.5) nur die ganzzahligen Werte $\delta_1 = \delta_2 = 0$, bzw. in der LP-Relaxierung nur die Wertekombinationen $[0, 0.5] \times [0, 0.5]$. Die Idee, effiziente Formulierungen zu entwickeln, besteht darin, bereits mit der LP-Relaxierung eines MILP-Problems, einige offensichtlich unerwünschte Lösungen, in denen fraktionale Variablen auftreten, auszuschließen und zu vermeiden.

Ein weiteres, anspruchsvolleres Beispiel für disjunktive Implikationen ist das folgende, in dem verlangt wird, dass genau eine von m Ungleichungen

$$\sum_{j=1}^{n} A_{ij} x_j \leq B_i \quad , \quad i = 1, 2, ..., m \tag{5.2.6}$$

erfüllt sein muss. Hierbei kann (5.2.2) in einer erweiterten Form verwendet werden, wobei zunächst Binärvariablen $\delta_1, \delta_2, ..., \delta_m$ eingeführt werden, die indizieren ob Ungleichung i erfüllt sein muss. *Genau eine* der Ungleichungen zu erfüllen, wird mit

$$\sum_{i=1}^{m} \delta_i = 1 \tag{5.2.7}$$

erzwungen, während mit U_i als Obergrenze für $\sum_{j=1}^{n} A_{ij}x_j - B_i$ die Ungleichungen

$$\sum_{j=1}^{n} A_{ij}x_j - B_i \leq U_i(1-\delta_i) \quad , \quad i = 1, 2, ..., m \tag{5.2.8}$$

für $\delta_i = 1$ zur Erfüllung der Ungleichung i führen.

Es gibt allerdings eine alternative, von Jeroslow & Lowe (1984,[148]) entwickelte Formulierung, die besonders für größere Probleme nützlich ist, da die LP-Relaxierung bereits die konvexe Hülle liefert und deshalb vorzuziehen ist. Hierbei sind die zusätzlichen Variablen $y_{ij} \geq 0$ erforderlich, die mit den ursprünglichen Variablen x_j durch

$$x_j = \sum_{i=1}^{m} y_{ij} \quad , \quad j = 1, 2, ..., n \tag{5.2.9}$$

verknüpft sind. Die Formulierung setzt

$$A_{ij} \geq 0, \quad i = 1, 2, ..., m, \quad j = 1, 2, ..., n$$

voraus und führt auf

$$\sum_{j=1}^{n} A_{ij}y_{ij} - B_i\delta_i \leq 0 \quad , \quad i = 1, 2, ..., m \quad , \tag{5.2.10}$$

wobei, wie zuvor, $\delta_1, \delta_2, \ldots, \delta_m$ Binärvariablen sind. Die Gleichung (5.2.7) erzwingt wieder, dass genau eine der Binärvariablen den Wert 1 annimmt, die im Folgenden mit δ_r bezeichnet wird. Für alle $i \neq r$ gilt also $\delta_m = 0$, damit

$$\sum_{j=1}^{n} A_{ij}y_{ij} = 0 \quad , \quad \forall i \neq r$$

und wegen $A_{ij} \geq 0$ somit $y_{ij} = 0$ für alle $i \neq r$ und j. Für $i = r$ folgt aus (5.2.9)

$$x_j = y_{rj} \quad , \quad j = 1, 2, ..., n \quad .$$

Betrachten wir (5.2.10) genauer, so bleibt lediglich

$$\sum_{j=1}^{n} A_{rj}y_{rj} \leq B_r \tag{5.2.11}$$

übrig, wobei wir noch y_{rj} durch x_j ersetzen können. Es ist sichergestellt, dass genau eine Nebenbedingung von (5.2.6) erfüllt ist, aber mit veränderten y_{ij} Variablen, von denen nur eine einzige, nämlich $x_j = y_{rj}$ von Null verschieden sein kann.

5.3 Modellierung von Nicht-Null-Variablen

Hier soll der Fall betrachtet werden, in der eine diskrete Variable σ alle ganzzahligen Werte zwischen 1 und 10 annehmen kann, aber aus irgendeinem Grund nicht den Wert 4. Derartige Variablen können mittels der auf Seite 11 eingeführten diskreten Variablen behandelt werden, d. h.

$$\sigma = \sum_{k=1}^{3} k\delta_k + \sum_{k=4}^{9}(k+1)\delta_k \quad , \quad \sum_{i=1}^{9}\delta_k = 1 \quad .$$

Sie können aber auch mit Hilfe der disjunktiven Ungleichungen

$$\sigma \leq 3 \quad \vee \quad \sigma \geq 5$$

abgebildet werden. Die Variable σ könnte mit Hilfe der in Abschnitt 5.2.3 beschriebenen Methode zur Behandlung disjunktiver Methoden modelliert werden, wenn wir zuerst eine (kontinuierliche oder ganzzahlige) *freie*, d. h. hinsichtlich des Vorzeichens unbeschränkte, Nicht-Null-Variable ς einführen, die mit der ursprünglichen Variable σ gemäß

$$\sigma - 4 = \varsigma$$

verknüpft ist und den folgenden Bedingungen genügt:

$$-Z_1 \leq \varsigma \leq Z_2 \quad , \quad \varsigma \neq 0 \quad , \quad Z_1 = 4 \quad , \quad Z_2 = 6 \quad .$$

Nicht-Null-Variablen wurden erstmalig von Hajian *et al.* (1996,[125]) betrachtet. Handelt es sich bei ς um eine kontinuierliche Variable, so wird die Punktbedingung $\varsigma \neq 0$ mit Hilfe eines kleinen positiven Parameters $\Delta > 0$ durch $\varsigma \notin [-\Delta, +\Delta]$ approximiert

$$\varsigma \leq -\Delta \quad \vee \quad \varsigma \geq \Delta \quad .$$

Führt man ferner zwei Binärvariablen $\delta_1, \delta_2 \in \{0, 1\}$ ein, so können die Ideen aus Abschnitt 5.2.3 für den Fall $m = 2$, $n = 1$ zweier disjunktiver Ungleichungen

$$\varsigma \leq -\Delta \quad \vee \quad \varsigma \geq \Delta$$

angewendet werden. Da wir (5.2.8) benutzen wollen, formen wir dies äquivalent so um, dass nur $\leq$ Ungleichungen auftreten und erhalten

$$\varsigma \leq -\Delta \quad \vee \quad -\varsigma \leq -\Delta \quad . \tag{5.3.1}$$

Wenn (5.2.8) auf (5.3.1) angewendet wird, folgen die beiden Ungleichungen

$$\varsigma + \Delta \leq U_1(1 - \delta_1) \quad , \quad -\varsigma + \Delta \leq U_2(1 - \delta_2)$$

mit den oberen Schranken $U_1 = Z_2 + \Delta$ und $U_2 = Z_1 + \Delta$. Dies führt zu den Beziehungen

$$\varsigma + (Z_2 + \Delta)\delta_1 \leq Z_2 \quad , \quad \varsigma - (Z_1 + \Delta)\delta_2 \geq -Z_1 \quad , \quad \delta_1 + \delta_2 = 1 \tag{5.3.2}$$

oder in der algebraisch äquivalenten Form zu,

$$\varsigma + \Delta\delta_1 \leq \delta_2 Z_2 \quad , \quad \varsigma - \Delta\delta_2 \geq -\delta_1 Z_1 \quad , \quad \delta_1 + \delta_2 = 1 \quad . \tag{5.3.3}$$

Aus (5.3.2) und (5.3.3) folgen, wie für die Nicht-Null-Variable ς zu erwarten ist, die Implikationen:

$$\delta_1 = 1 \quad \Longrightarrow \delta_2 = 0 \quad , \quad -Z_1 \leq \varsigma \leq -\Delta$$

und

$$\delta_2 = 1 \quad \Longrightarrow \delta_1 = 0 \quad , \quad \Delta \leq \varsigma \leq Z_2 \quad .$$

Die Bedingungen (5.3.2) oder (5.3.3) sind zwar algebraisch äquivalent, können aber zu verschiedenen Rechenzeiten in B&B-Verfahren führen. Deshalb ist es ratsam, beide Formulierungen auf ihre numerische Tauglichkeit hin zu prüfen.

Im folgenden Beispiel wird ersichtlich, wie nützlich Nicht-Null-Variablen sein können. Eine Temperaturvariable möge die in Grad Celsius gemessene Temperatur beschreiben, wobei aus physikalisch-technischen Gründen Temperaturen um den Gefrierpunkt des Wassers ausgeschlossen sein sollen. In diesem Falle wird die Temperaturvariable als eine Nicht-Null-Variable behandelt; der zugehörige Parameter $\Delta = 1°\mathrm{C}$ scheint sinnvoll.

5.4 Modellierung von Mengen paarweise verschiedener Werte

Ein einfaches Beispiel für eine Situation, in denen die Variablenwerte der optimalen Lösung paarweise verschieden sein müssen [engl.: *all-different relation*] ist der Fall einer Gruppe von Verkäufern, von denen jeder eine andere Stadt besuchen muss (es sei nicht angenommen, dass jede Stadt von einem Verkäufer besucht werden muss; möglicherweise gibt es mehr Städte als Verkäufer). In diesem Falle ist es wieder möglich, mit Hilfe der Binärvariablen δ_{ij}, die indizieren, ob Verkäufer i die Stadt j besucht, und der Ungleichung

$$\sum_i \delta_{ij} \leq 1 \quad , \quad \forall j \tag{5.4.1}$$

zu erzwingen, dass keine Stadt von mehr als einem Verkäufer besucht werden kann; gleichsam garantiert die Gleichung

$$\sum_j \delta_{ij} = 1 \quad , \quad \forall i \quad ,$$

dass jeder Verkäufer genau eine Stadt besucht.

Dieser Fall mit diskreten Objekten, hier Städten, ist einfach, da verschiedene Städte einfach durch verschiedene Indizes abgebildet werden können und man eine Binärvariable einfügen kann, die die *all-different relation* durch die Ungleichung (5.4.1) verwirklicht.

Weitaus schwieriger ist der kontinuierliche Fall, in dem eine Menge kontinuierlicher Variablen $x_i \geq 0$ zusammen mit der Anforderung betrachtet wird, dass für jedes Indexpaar $i \neq j$ die Bedingungen $x_i \neq x_j$ erfüllt sein müssen. Hier hilft es, die durch

$$\varsigma_{ij} = x_i - x_j \quad , \quad \forall i, j \mid i \neq j$$

definierten Nicht-Null-Variablen – siehe hierzu Abschnitt 5.3 – einzuführen und die Nebenbedingungen (5.3.2) und (5.3.3) anzuwenden. Das folgende Beispiel zeigt eine nützliche Anwendung der *all-different* Relation. Eine Firma möchte einige elektronische Geräte produzieren, deren Funktionalität im Wesentlichen durch drei Frequenzen f_1, f_2 und f_3 bestimmt ist. Zu beachten ist dabei, dass Resonanzen niedriger Ordnung vermieden werden müssen. Sollen alle Resonanzen bis zur Ordnung 4 vermieden werden, so kann dies durch die Ungleichungen

$$f_i \neq n f_j \quad , \quad \forall i,j \mid i \neq j \quad , \quad n = 1,2,3,4$$

beschrieben werden, wobei die $f_i \geq 0$ kontinuierliche Variablen sind, die Frequenzen beschreiben. Mit Hilfe der Nicht-Null-Variablen

$$\varsigma_{ij} = f_i - n f_j \quad , \quad \forall i,j \mid i \neq j \quad , \quad n = 1,2,3,4$$

kann dieses Problem beschrieben werden.

5.5 Modellierung von Betragstermen

In Modellen können manchmal Ausdrücke der Form $|x_1 - x_2|$ auftreten. So kann z. B. der Produktionsausfall bei Änderung der Produktionsraten x eines Chemiereaktors der Betragsdifferenz $|x_1 - x_2|$

$$|x_1 - x_2| = \begin{cases} x_1 - x_2 \text{ , wenn } x_1 \geq x_2 \\ x_2 - x_1 \text{ , wenn } x_1 < x_2 \end{cases}$$

proportional sein, wobei x_1 die Produktionsrate im ersten Zeitintervall und x_2 die Produktionsrate im zweiten Zeitintervall darstellt.

Während x_1 und x_2 nichtnegative Variablen sind und $|x_1 - x_2|$ stets nichtnegativ ist, unterliegt die Differenz $x_1 - x_2$ keiner Vorzeichenbeschränkung. Wir können $|x_1 - x_2|$ sowohl als nichtlineare Funktion oder alternativ wegen des *wenn*-Ausdrucks als einen logischen Ausdruck interpretieren. Sei

$$l(\mathbf{x}) = L_1 x_1 + \ldots + L_k x_k$$

irgendein linearer Ausdruck, der als $|l(\mathbf{x})|$ in unserem Modell erscheint. In diesem Falle führen wir zwei nichtnegative Variablen $a^+ \geq 0$ und $a^- \geq 0$ ein. Damit können nun $|l(\mathbf{x})|$ und $l(\mathbf{x})$ bei jedem Auftreten durch

$$|l(\mathbf{x})| = a^+ + a^- \tag{5.5.1}$$

bzw.

$$l(\mathbf{x}) = a^+ - a^- \tag{5.5.2}$$

ersetzt werden. Der Term $a^+ + a^-$ stellt exakt den Betrag $|l(\mathbf{x})|$ dar, wenn mindestens eine der Variablen a^+ und a^- den Wert 0 annimmt, also

$$a^+ a^- = 0 \tag{5.5.3}$$

erfüllt ist. Wird der absolute Wert $|l(\mathbf{x})|$ selbst benötigt, so kann dieser durch eine zusätzliche nichtnegative Variable, z. B. $a \geq 0$, und der Gleichung

$$a = a^+ + a^- \tag{5.5.4}$$

verfügbar gemacht werden; in diesem Fall wird man bei jedem Auftreten $|l(\mathbf{x})|$ durch a ersetzen und die Gleichung (5.5.4) hinzufügen.

Der Term $a^+ - a^-$ könnte uns an den Gebrauch der im Kapitel 2 eingeführten *freien Variablen* erinnern, von denen nur eine der Variablen a^+ oder a^- einen positiven Wert annehmen konnte. Im vorliegenden Fall der Betragsfunktion ist dies anders. Da $a^+ + a^-$ und $a^+ - a^-$ linear unabhängig sind, können sowohl a^+ als auch a^- Basisvariablen werden und somit von Null verschiedene Werte annehmen. Die nichtlineare Bedingung (5.5.3) kann entweder durch SOS-1-Mengen [siehe Abschnitt 5.7] oder durch eine Binärvariable δ und den beiden Ungleichungen

$$\begin{array}{rcl} a^+ & \leq & M\delta \\ a^- & \leq & M(1-\delta) \end{array}$$

modelliert werden, wobei M eine obere Schranke für $|l(\mathbf{x})|$ darstellt; $\delta = 0$ führt zu $a^+ = 0$, während $\delta = 1$ zu $a^- = 0$ führt. Es ist also mindestens eine der beiden Variablen a^+ und a^- gleich Null.

Bleibt zu klären, wie die Konstante M gewählt werden soll. Im Hinblick auf numerische Effizienz sollte M, wie z. B. in Abschnitt 5.10.1.2 begründet, so klein wie möglich gewählt werden. Zunächst gelten wegen $x_j \geq 0$ die Ungleichungen

$$\begin{array}{rcccl} a^+ & \leq & |l(\mathbf{x})| & \leq & |L_1| \max x_1 + \ldots + |L_k| \max x_k \\ a^- & \leq & |l(\mathbf{x})| & \leq & |L_1| \max x_1 + \ldots + |L_k| \max x_k \end{array} \quad .$$

Schärfere obere Schranken lassen sich durch eine differenzierte Betrachtung der Variablen in $l(\mathbf{x})$ mit positiven und negativen Koeffizienten bestimmen. Bezeichnen $\mathcal{L}_1$ und $\mathcal{L}_2$ die Index-Mengen der Variablen mit positiven und negativen Koeffizienten l_j, so gilt

$$|l(\mathbf{x})| \leq \max \left\{ \sum_{j \in \mathcal{L}_1} |L_j| \max x_j, \sum_{j \in \mathcal{L}_2} |L_j| \max x_j \right\} \quad . \tag{5.5.5}$$

Das Verfahren sei an folgendem Beispiel

$$\max \quad Z = |x_1 - 2x_2|$$

unter den Nebenbedingungen

$$x_1 \leq 3 \quad , \quad x_2 \leq 4$$

veranschaulicht. Dieses Problem nimmt die folgende Gestalt

$$\max \quad Z = a^+ + a^-$$

unter den Nebenbedingungen

$$\begin{array}{c} x_1 \leq 3 \quad , \quad x_2 \leq 4 \\ x_1 - 2x_2 = a^+ - a^- \end{array}$$

und

$$\begin{array}{rcl} a^+ & \leq & M\delta \\ a^- & \leq & M(1-\delta) \end{array}$$

an. Bei Verwendung von (5.5.5) erhalten wir für A

$$M = \max \{|1| \cdot 3, |-2| \cdot 4\} = 8$$

In diesem Beispiel lautet die Lösung: $x_1 = 0$, $x_2 = 4$, $Z = 8$, $\delta = 0$, $a^+ = 0$ und $a^- = 8$; die obere Schranke ist mit $M = 8$ also optimal, d. h. so klein wie möglich.

5.6 Behandlung und Transformation nichtlinearer Probleme

Einige spezielle nichtlineare Optimierungsprobleme lassen sich in lineare gemischt-ganzzahlige Programme mit binären Variablen transformieren. Der Vorteil solcher Transformationen besteht darin, dass nichtlineare Optimierungsprobleme meist nur hinreichende Bedingungen für lokale Optima zulassen, d. h. die zugehörigen Algorithmen nicht notwendigerweise das globale Optimum liefern. Hingegen liefern (exakte) Algorithmen der linearen gemischt-ganzzahligen Programmierung das globale Optimum oder ermöglichen bei vorzeitigem Abbruch zumindest eine Aussage über die Qualität der gefundenen Lösung. In Abschnitt 6.3 wird sogar ein Beispiel behandelt, bei dem ein scheinbar nichtlineares Problem komplett als lineares Programm gelöst wird.

5.6.1 Nichtlineare binäre Optimierungsprobleme

Das folgende nichtlineare Problem

$$\max\ Z = \delta_1^2 + \delta_2\delta_3 - \delta_3^3 \quad , \quad -2\delta_1 + 3\delta_2 + \delta_3 \leq 3 \quad , \quad \delta_i \in \{0,1\}$$

könnte aus einem Problem mit logischen Ausdrücken resultieren und kann in einem ersten Schritt mit Hilfe der für alle Binärvariablen und alle $k \in \mathbb{N}$ gültigen Beziehung

$$\delta_i^k = \delta_i \quad , \quad \delta_i \in \{0,1\} \quad , \quad i = 1,2,3 \tag{5.6.1}$$

vereinfacht werden und verdeutlicht, wie im allgemeinen Fall binärer Optimierungsprobleme vorgegangen werden kann. In einem zweiten Schritt, in dem

$$\delta_i \in \{0,1\} \rightarrow \delta_{23} \in \{0,1\} \quad , \quad i = 1,2$$

festgestellt wird, wird eine Binärvariable δ_{23} eingeführt, so dass gilt:

$$\delta_{23} = \begin{cases} 1, \, x_2 = x_3 = 1 \\ 0, \, \text{sonst} \end{cases}$$

und

$$\begin{array}{lcl} \delta_{23} & \geq & \delta_2 + \delta_3 - 1 \\ \delta_{23} & \leq & \delta_2 \\ \delta_{23} & \leq & \delta_3 \end{array} \quad .$$

Mit den Variablen

$$\delta_i \in \{0,1\} \quad , \quad \delta_{23} \in \{0,1\}$$

schreibt sich die zu maximierende Zielfunktion als

$$Z = \delta_1 + \delta_{23} - \delta_3$$

und die Nebenbedingungen erscheinen in der Form

$$\begin{array}{rcl} -2\delta_1 + 3\delta_2 + \delta_3 & \leq & 3 \\ \delta_2 + \delta_3 - \delta_{23} & \leq & 1 \\ -\delta_2 + \delta_{23} & \leq & 0 \\ -\delta_3 + \delta_{23} & \leq & 0 \end{array} \quad .$$

Das nichtlineare Produkt x_2x_3 wurde hier mit Hilfe einer zusätzlichen Binärvariablen δ_{23} und drei zusätzlichen Ungleichungen eliminiert. Allerdings braucht δ_{23} gar nicht als Binärvariable deklariert zu werden; es genügt die Relaxierung $0 \leq \delta_{23} \leq 1$. Der Fall $x_2 = x_3 = 1$ impliziert wegen $\delta_2 + \delta_3 - \delta_{23} \leq 1$ nämlich $\delta = 1$. Ist dagegen $\delta_2 = 0$ oder $\delta_3 = 0$, so folgt sofort $\delta_{23} = 0$.

Diese Eigenschaft überträgt sich auch auf das Produkt

$$\delta_p = \prod_{k=1}^{K} \delta_k$$

von K binären Variablen $\delta_k \in \{0, 1\}$; hier genügt die relaxierende Ungleichung

$$0 \leq \delta_p \leq 1$$

nebst den zusätzlichen Ungleichungen

$$\begin{aligned} K\delta_p - \sum_{k=1}^{K} \delta_k &\leq 0 \\ \delta_p &\leq \delta_k \ ; \ k = 1, ..., K \quad , \end{aligned}$$

um sicherzustellen, dass $\delta_p \in \{0, 1\}$.

5.6.2 Quadratische Optimierungsprobleme in binären Variablen

In der quadratischen Optimierung [engl.: *quadratic programmierung; QP*] beschäftigt man sich mit Optimierungsproblemen der Form

$$\min \frac{1}{2}\mathbf{x}^{\mathrm{T}}\mathsf{G}\mathbf{x} + \mathbf{g}^{\mathrm{T}}\mathbf{x}$$

unter den Nebenbedingungen

$$\mathsf{A}^{\mathrm{T}}\mathbf{x} \geq \mathbf{b} \quad , \quad \mathbf{x} \geq \mathbf{0} \quad , \quad \mathbf{x} \in \mathbb{R}^n \quad .$$

Bilineare Terme, also Produkte von Variablen, treten nur in der Zielfunktion auf, nicht aber in den Nebenbedingungen. Anwendungen finden sich in der Portfoliotheorie oder bei der Preisbildung. Nimmt man z. B. an, dass die Pro-Stück-Kosten C_p beim Einkauf oder Verkauf linear gemäß

$$C_p := A_p x_p + B_p \tag{5.6.2}$$

von der verkauften Stückzahl x_p abhängen, so führt dies zu dem akkumulierten Preis

$$\sum_{i=1}^{n} C_p x_p = \sum_{i=1}^{n} A_p x_p^2 + B_p x_p \quad .$$

Diskrete Stückzahlen x_p können mit Hilfe von Binärvariablen dargestellt werden

$$x_p := \sum_{k=0}^{K} 2^k \delta_{pk} \quad ,$$

wodurch wir ein binäres quadratisches Problem erhalten. Betrachtet man dagegen statt Stückzahlen x_p – wie beispielsweise – in der chemischen Industrie auch kontinuierliche Verkaufsmengen x_p, so erhält man ein kontinuierliches QP.

Je nach Wahl von A_p kann hiermit eine Rabattstruktur oder ein Kostenwachstum beschrieben werden. Bei kleinen Stückzahlen oder Mengen kann mit negativen Werten von A_p eine Rabattierung beschrieben werden; hierbei ist sicherzustellen, dass C_p keine negativen Werte annimmt – der Zusammenhang (5.6.2) wird also nur für beschränkte Werte von gelten. Möchte man ein nur beschränkt zum Verkauf anstehendes Gut möglichst vielen Käufern zukommen lassen, so könnte man bei größeren Abnahmemengen die Kosten ansteigen lassen; dies wird durch positive Werte von A_p realisiert. Schließlich seien noch die in Abschnitt 11.4 beschriebenen Preisstrukturen mit Stufenrabatten als mögliche Alternative erwähnt, die dann allerdings in der Modellierung binäre Variablen erfordert.

Ein Spezialfall von QP-Problemen mit nur binären Variablen sind quadratische Zuordnungsprobleme; sie spielen eine Rolle in der Standortplanung, im Scheduling oder auch bei der Auslegung elektrischer Schaltkreise. Jedes QP-Problem, in dem nur binäre Variablen $\boldsymbol{\delta}$ auftreten, kann als konvexes QP-Problem umgeschrieben werden; äquivalent zur Konvexität des Problems ist die Bedingung $\boldsymbol{\delta}^{\mathrm{T}}\mathsf{G}\boldsymbol{\delta} > 0$ für alle $\boldsymbol{\delta} \neq \mathbf{0}$ bzw. die Forderung, dass alle Eigenwerte der Matrix G positiv sind. Hierzu wird zum quadratischen Term $\frac{1}{2}\boldsymbol{\delta}^{\mathrm{T}}\mathsf{G}\boldsymbol{\delta}$ der Zielfunktion die Summe $\sum_{i=1}^{n} \mu\delta_i^2$ mit hinreichend großem $\mu > 0$ hinzu addiert und anschließend wieder subtrahiert; die Zielfunktion nimmt dann die Gestalt

$$\begin{aligned} \frac{1}{2}\boldsymbol{\delta}^{\mathrm{T}}\mathsf{G}\boldsymbol{\delta} + \boldsymbol{\delta}^{\mathrm{T}}\boldsymbol{\delta} &= \frac{1}{2}\boldsymbol{\delta}^{\mathrm{T}}\mathsf{G}'\boldsymbol{\delta} + \mathbf{g}^{\mathrm{T}}\boldsymbol{\delta} - \sum_{i=1}^{n} \mu\delta_i^2 \\ &= \frac{1}{2}\boldsymbol{\delta}^{\mathrm{T}}\mathsf{G}'\boldsymbol{\delta} + \mathbf{g}^{\mathrm{T}}\boldsymbol{\delta} - \sum_{i=1}^{n} \mu\delta_i = \frac{1}{2}\boldsymbol{\delta}^{\mathrm{T}}\mathsf{G}'\boldsymbol{\delta} + (\mathbf{g} - \mu\mathbf{e})^{\mathrm{T}}\boldsymbol{\delta} \end{aligned}$$

an. Für hinreichend großes μ wird $\mathsf{G}' = \mathsf{G} + 2\mu\mathbb{1}$ schließlich positiv definit; der Abzugsterm wurde linear umgeschrieben, wobei ausgenutzt wurde, dass wegen (5.6.1) für jede Komponente δ_i des Vektors die Beziehung $\delta_i = \delta_i^2$ gilt, da δ_i eine binäre Variable ist; der Vektor $\mathbf{e} \in \mathbb{R}^n$ ist ein Vektor, dessen Komponenten alle den Wert 1 haben. Wie auf S. 50 in [137] gezeigt, kann aber auch jedes QP-Problem mit symmetrischer Matrix G, in dem nur binäre Variablen auftreten, als konkaves binäres QP-Problem umgeschrieben werden; das Problem ist dann äquivalent zur Minimierung einer konkaven Funktion über dem Einheitswürfel.

5.6.3 Behandlung von stückweise linearen Funktionen

Gegeben seien eine stetige, stückweise lineare Funktion $f(x)$ und eine Menge S von Stützpunkten $[X_i, F_i = f(X_i)]$, die $f(x)$ vollständig charakterisieren, also

$$f(x) := g_i(x) := A_i(x - x_i) + F_i \quad \text{falls} \quad X_i \leq x \leq X_{i+1} \quad , \tag{5.6.3}$$

wobei $g_i(x)$ das i-te lineare Segment der Funktion $f(x)$ bezeichnet und der Koeffizient A_i den Wert

$$A_i := \frac{F_{i+1} - F_i}{X_{i+1} - X_i} = C_i\,(F_{i+1} - F_i) \quad , \quad C_i := \frac{1}{X_{i+1} - X_i}$$

hat. Unter diesen Voraussetzungen kann ein beliebiger Argumentwert x mit $X_1 \le x \le X_S$ als Konvexkombination

$$x = \sum_{i=1}^{S} \lambda_i X_i \quad , \quad \sum_{i=1}^{S} \lambda_i = 1 \quad , \quad \lambda = (\lambda_1, ..., \lambda_S) \in (\mathbb{R}_0^+)^S$$

geschrieben werden. Die λ_i sind zwar i. Allg. nicht eindeutig, aber falls die λ_i so gewählt werden, dass im Falle $X_i \le x \le X_{i+1}$

$$x = \lambda_i X_i + \lambda_{i+1} X_{i+1}$$

und

$$\lambda_i + \lambda_{i+1} = 1$$

gilt, so folgt

$$f := f(x) = \sum_{i=1}^{S} \lambda_i F_i \quad , \quad \sum_{i=1}^{S} \lambda_i = 1 \quad , \quad \boldsymbol{\lambda} = (\lambda_1, ..., \lambda_S) \ \in \ \left(\mathbb{R}_0^+\right)^S \quad .$$

Setzt man nämlich $x = \lambda_i X_i + \lambda_{i+1} X_{i+1}$ in (5.6.3) ein, so folgt zunächst

$$X_i \le x \le X_{i+1}$$

und damit, weil das passende Segment bekannt ist,

$$\begin{aligned}
f(x) &= \frac{F_{i+1} - F_i}{X_{i+1} - X_i}(\lambda_i X_i + \lambda_{i+1} X_{i+1} - X_i) + F_i \\
&= \frac{-F_{i+1}\lambda_i X_i - F_{i+1}\lambda_{i+1} X_{i+1} + F_{i+1} X_i + F_i \lambda_i X_i + F_i \lambda_{i+1} X_{i+1} - F_i X_{i+1}}{X_i - X_{i+1}} \\
&= \frac{\left(-F_{i+1}\lambda_i + F_{i+1} + F_i\lambda_i\right) X_i + \left(-F_{i+1}\lambda_{i+1} + F_i\lambda_{i+1} - F_i\right) X_{i+1}}{X_i - X_{i+1}} \\
&= \frac{\left[-F_{i+1}(1-\lambda_{i+1}) + F_{i+1} + F_i\lambda_i\right] X_i + \left[-F_{i+1}\lambda_{i+1} + F_i(1-\lambda_i) - F_i\right] X_{i+1}}{X_i - X_{i+1}} \\
&= \frac{\left[F_{i+1}\lambda_{i+1} + F_i\lambda_i\right] X_i - \left[F_{i+1}\lambda_{i+1} + F_i\lambda_i\right] X_{i+1}}{X_i - X_{i+1}} \\
&= F_i\lambda_i + F_{i+1}\lambda_{i+1} \quad .
\end{aligned}$$

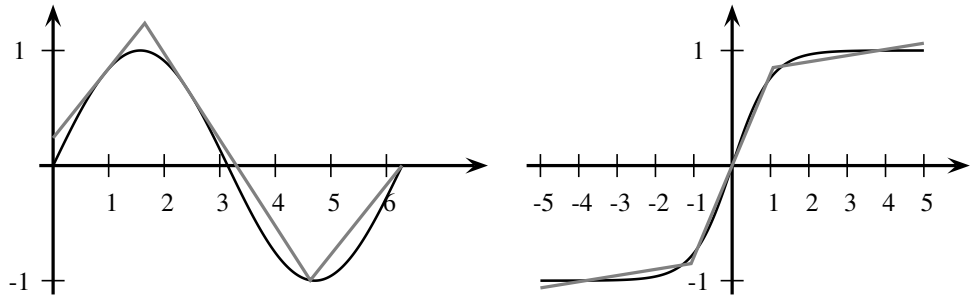
Vorausgesetzt wird also, dass höchstens zwei λ_i positiv sind und falls λ_j und λ_k positiv sind, die Beziehung $|j-k| = 1$ erfüllt ist. Diese Bedingung an den Vektor λ führt auf den in kommerziellen Paketen zur Lösung gemischt-ganzzahliger Programme häufig realisierten Variablentyp SOS-2 (*Special Ordered Sets of type 2*); SOS-1-Variablen genügen der Bedingung, dass höchstens ein λ_i von Null verschieden ist. Die SOS-2-Variablen λ könnten formal durch $S-1$ binäre Variablen δ_i

$$\delta_i = \begin{cases} 1, \ X_i \le x \le X_{i+1} \\ 0, \text{ sonst} \end{cases}$$

und den Restriktionen

$$\sum_{i=1}^{S-1} \delta_i = 1 \quad ,$$

Abbildung 5.1 Optimale stückweise lineare Approximatoren für sin$x$ und tanh$x$ mit $\delta = 0.24$ bzw. $\delta = 0.063$ und nur 4 Stützstellen.



sowie

$$\begin{aligned} \lambda_1 &\leq \delta_1 \quad , \\ \lambda_i &\leq \delta_{i-1} + \delta_i \quad ; \quad i = 2, ..., S-1 \quad , \\ \lambda_S &\leq \delta_{S-1} \end{aligned}$$

realisiert werden. Aus $\delta_j = 1$ folgt zwingend $\lambda_i = 0$ für $i \neq j$ und $i \neq j+1$.

Ist die Funktion $f(x)$ keine stückweise lineare, sondern eine beliebige stetige Funktion, kann die vorgestellte Vorgehensweise übernommen werden, wobei die X_i nun Interpolationsstützstellen und die λ_i Interpolationskoeffizienten sind. Rebennack & Kallrath (2013a,[241]; 2013b,[242]) berechnen minimale Stützstellensysteme derart, dass die Approximation der nichtlinearen Funktion $f(x)$ durch stückweise stetige linearer Funktionen $\ell(x)$ einer vorgegebenen Genauigkeit δ genügt. Die Approximation ist hier allgemeiner als die Interpolation, da nicht $\ell(X_i) = f(X_i)$ gefordert wird, d. h. in den Stützstellen wird auf die Übereinstimmung von Approximator und Funktionswert verzichtet. Statt der Approximation kann so auch ein stückweise stetiger linearer Über- oder Unterschätzer konstruiert werden. Die Berechnung dieser Stützstellensysteme erfordert, dass nichtkonvexe nichtlineare Hilfsprobleme global optimal gelöst werden.

Wie gezeigt, können nichtlineare Zusammenhänge durch lineare Ungleichungen und binäre Variablen beschrieben werden. Für die Praxis relevant ist, dass die nichtlinearen Zusammenhänge direkt in den B&B-Prozess mit einbezogen werden können, was zu erheblichen Rechenzeiteinsparungen führen kann; daher auch die Einführung der in Abschnitt 5.7 behandelten SOS-2-*Mengen*.

Eine numerisch interessante Formulierung für $n = 1+2^k$ gegebene Stützstellen bietet Vielma & Nemhauser (2011,[285]). Statt n SOS-2-Variablen werden nur k Binärvariablen und $2k$ lineare Ungleichungen benötigt, d. h. die Anzahl der Binärvariablen *und* die Anzahl der zusätzlichen Ungleichungen wachsen nur logarithmisch in der Anzahl der Stützstellen. Betrachten wir dazu ein kleines Beispiel mit 5 Stützstellen, also $k = 2$, und den nichtnegativen Variablen $\lambda_0, \lambda_1, \ldots, \lambda_4$, die den SOS-2-Variablen entsprechen, aber nun nicht als solche deklariert werden, sondern lediglich der Gleichung

$$\sum_{j=0}^{4} \lambda_j = 1$$

genügen. Mit Hilfe der beiden Binärvariablen δ_1 und δ_2 sowie der $2k = 4$ Ungleichungen

$$\begin{aligned} \lambda_0 + \lambda_1 &\leq \delta_1 \\ \lambda_2 &\leq \delta_2 \\ \lambda_3 + \lambda_4 &\leq 1 - \delta_1 \\ \lambda_0 + \lambda_4 &\leq 1 - \delta_2 \end{aligned}$$

können mit jeder (δ_1,δ_2)-Kombination genau die für die möglichen Interpolationsintervalle zulässigen Paare $(\lambda_j, \lambda_{j+1})$, $j = 0, 3$, erzeugt werden:

δ_1	δ_2	auf Null fixiert	verbleibende Ungleichungen	$(\lambda_j, \lambda_{j+1})$
0	0	$\lambda_0, \lambda_1, \lambda_2$	$\lambda_3 + \lambda_4 \leq 1$	(λ_3, λ_4)
1	0	$\lambda_2, \lambda_3, \lambda_4$	$\lambda_0 + \lambda_1 \leq 1$	(λ_0, λ_1)
0	1	$\lambda_0, \lambda_1, \lambda_4$	$\lambda_2 \leq 1, \lambda_3 \leq 1$	(λ_2, λ_3)
1	1	$\lambda_0, \lambda_3, \lambda_4$	$\lambda_1 \leq 1, \lambda_2 \leq 1$	(λ_1, λ_2)

.

Schließlich können wir mit einer neuen kontinuierlichen Variable z und der Ungleichung

$$z \geq g_i(x) := A_i(x - X_i) + F_i \quad , \quad \forall i = 1, \ldots, S-1 \quad , \tag{5.6.4}$$

zeigen, dass lineare beschränkte Optimierungsprobleme mit stückweise linearen, konvexen (konkaven) Zielfunktionen als LP-Probleme (nur kontinuierliche Variablen) minimiert (maximiert) werden können. In (5.6.4) beschreibt $g_i(x)$ gerade das i-te lineare Segment der Zielfunktion $f(x)$, die bei Minimierung durch $\min z$ ersetzt. Ähnlich wie beim Beweis des Eckensatzes der linearen Programmierung, siehe z. B. ([224], Satz 1.1.5, S. 48), gilt auch hier das Argument, dass falls

$$z > g_i(x) \quad , \quad \forall i = 1, \ldots, S-1$$

gilt, man noch einen Index i_* finden kann, sodass man mit

$$z_* = g_{i_*}(x) := A_{i_*}(x - X_i) + F_{i_*}$$

eine bessere Lösung erhält (Zielfunktionswert liegt auf einem Segment) – hier wird die Konvexität benötigt. Da es nur endlich viele Segmente gibt, wird auf einem dieser Segmente der minimale Zielfunktionswert angenommen. Formuliert man die konvexe stückweise lineare Funktion $f(x)$ mit SOS-2-Variablen, so erfüllen die λ Variablen bereits in der LP-Relaxierung automatisch die SOS-2-Bedingungen.

5.6.4 Produkte von Binärvariablen

Einige nichtlineare Funktionen, die Binärvariablen als Argumente enthalten, können auch durch MILP-Formulierungen abgebildet werden. Das erste Beispiel für derartige Funktionen sind Binärpotenzen. Jedes Mal, wenn wir auf δ^k treffen mit $k \in \mathbb{N}_0$, können wir δ^k durch δ ersetzen, da $\delta^k = \delta$ für Binärvariablen stets erfüllt ist. Modelle, die Produkte

$$p = \prod_{k=1}^{K} \delta_k \quad , \quad \delta_k \in \{0, 1\} \tag{5.6.5}$$

von K Binärvariablen δ_k beinhalten, können mit Hilfe der Ungleichungen

$$p \leq \delta_k \quad , \quad \forall k \quad ; \quad -p + \sum\nolimits_{k=1}^{K} \delta_k \leq K - 1 \quad ,$$

in MILP-Formulierung umgewandelt werden. Für den speziellen Fall $K = 2$ wird das Produkt $\delta_1 \delta_2$ durch die drei Ungleichungen

$$p \leq \delta_1 \quad , \quad p \leq \delta_2 \quad , \quad -p + \delta_1 + \delta_2 \leq 1$$

ersetzt, von denen die beiden ersten $p = 0$ implizieren, wenn eine der Binärvariablen 0 ist. Die dritte sichert, dass p den Wert 1 annimmt, wenn alle Faktoren den Wert 1 haben.

5.6.5 Produkte binärer und einer kontinuierlichen Variablen

Produkte der Form

$$y = x \prod_{k=1}^{K} \delta_k \quad ,$$

mit K Binärvariablen δ_k und einer kontinuierlichen oder ganzzahligen nichtnegativen Variablen x sowie einer oberen Schranke X^+ von x, können durch die Ungleichungen

$$\forall k : y \leq X^+ \delta_k \quad , \quad y \leq x \quad , \quad y \geq x - X^+ \left(K - \sum_{k=1}^{K} \delta_k \right) \tag{5.6.6}$$

dargestellt werden. Die erste Ungleichung von (5.6.6) impliziert ($\delta_k = 0 \Rightarrow y = 0$) und ($y > 0 \Rightarrow \sum_{k=1}^{K} \delta_k = K$), während die zweite und dritte Ungleichung ($\sum_{k=1}^{K} \delta_k = K \Rightarrow y = x$) und ($y = 0 \Rightarrow \sum_{k=1}^{K} \delta_k < K$) liefern.

5.7 Strukturierte Mengen - Special Ordered Sets

Im Abschnitt (5.1.4) wurden Mengen von Entscheidungen, von denen höchstens K Entscheidungen ausgewählt werden konnten, mit n Binärvariablen δ_i und der Ungleichung

$$\delta_1 + \delta_2 + ... + \delta_n \leq K$$

modelliert. Für $K = 1$ schließen sich die Entscheidungen gegenseitig aus – ein häufig auftretender Fall, der, falls die Binärvariablen mit einer Ordnungsstruktur assoziiert sind, mit Hilfe spezieller Mengen mit Ordnungsstruktur [engl.: *special ordered sets*, SOS-1, SOS-2] modelliert werden. Diese erlauben bei B&B besondere Verzweigungstechniken und können zu erheblichen Rechenzeitersparnissen führen. SOS-1-Mengen wurden von Beale & Tomlin (1970,[30]) eingeführt und später von Hummeltenberg (1984,[140]) und Wilson (1990,[293]) untersucht; sie sind besonders geeignet zur Modellierung diskreter Variablen (vergl. S. 11). Mit Hilfe von SOS-2-Mengen können stückweise lineare Funktionen modelliert werden, die wiederum zur Approximation nichtlinearer Terme in der Zielfunktion verwendet werden können [siehe Abschnitt 5.7.2].

5.7.1 Strukturierte Variablenmengen vom Typ 1 (SOS-1-Mengen)

Eine *speziell geordnete Menge vom Typ* 1 [engl.: *special ordered set of type* 1], kurz SOS-1-Menge genannt, ist eine geordnete Menge beliebiger Variablen, von denen jedoch höchstens eine von Null verschieden sein darf. In den meisten Fällen von SOS-1-Mengen werden die Variablen Binärvariablen δ_i sein. Häufig tritt die Variante auf, in denen genau eine Variable von Null verschieden sein muss, was durch die Konvexitätsbedingung

$$\sum_{i=1}^{n} \delta_i = 1 \tag{5.7.1}$$

modelliert werden kann. Betrachtet man die Variablen $\delta_1, \ldots, \delta_n$ als Elemente einer SOS-1-Menge $\{\delta_1, \ldots, \delta_n\}$ und fügt die Konvexitätsgleichung (5.7.1) hinzu, so können diese nur die Werte 0 und 1 annehmen; hierbei ist es nicht nötig die Variablen ausdrücklich als Binärvariablen zu deklarieren. Da es in diesem Zusammenhang häufig zu einem Missverständnis kommt, sei hier betont, dass das Konzept von SOS-1-Mengen nichts mit Konvexitätsbedingungen zu tun hat; im Abschnitt 9.2 werden wir auf ein Beispiel treffen, in dem keine solche Beziehung zwischen den Elementen einer SOS-1-Menge besteht.

Eine beim Konzept von SOS-1-Mengen wichtige Eigenschaft ist die Existenz von Ordnungsbeziehungen zwischen den Variablen. Es ist möglich, dass die Modelldaten eine natürliche, durch den Index vorgegebene Ordnungsstruktur besitzen. Diese Ordnungsstruktur wird durch eine Referenzbedingung [engl.: *reference row*]

$$x = \sum_{i=1}^{n} X_i \delta_i$$

repräsentiert, wobei die X_i Gewichte (Zeiten, Kapazitäten, etc.) sind, die durch δ_i gewählt werden sollen. Die Referenzbedingung und die im Index i monoton wachsenden oder fallenden Koeffizienten X_i etablieren die erwähnte Ordnung zwischen den Variablen δ_i.

Das Konzept und der Nutzen von SOS-1-Mengen kann mit Hilfe des folgenden Beispiels verdeutlicht werden. Bei einem Netzwerkdesignproblem möchte eine Firma die optimale Topologie und die Kapazitäten von Rohrleitungen, die die Anlagen eines Netzwerks verbinden, bestimmen. Die Rohrleitungen bzw. deren Durchmesser können nur in diskreten Größen $C_1, ..., C_n$ ausgewählt und gekauft werden. Diese Kapazitäten seien im folgenden Sinne geordnet: Je größer der Index, desto größer die Kapazität, z. B.,

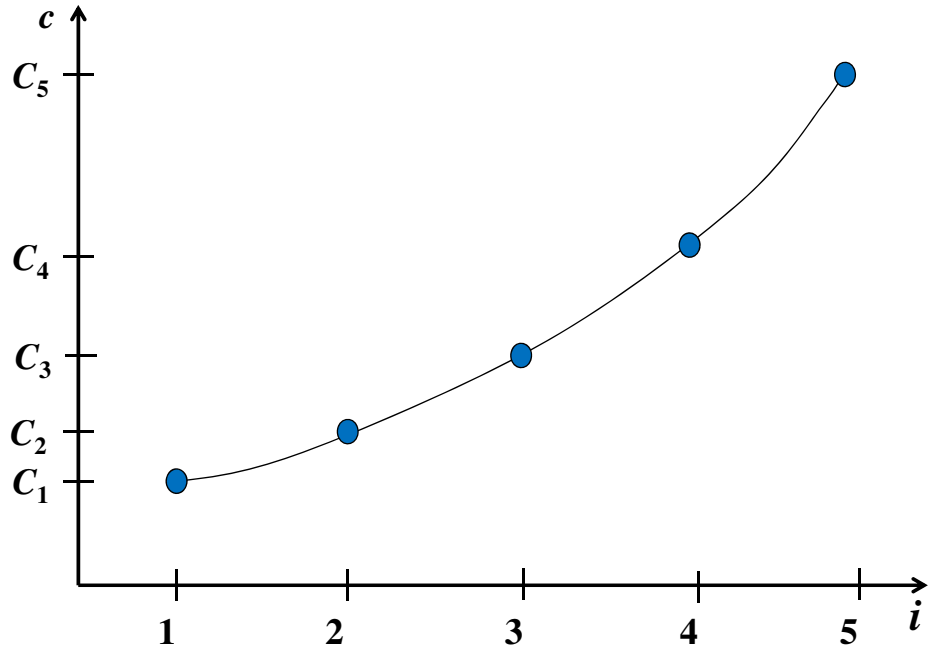
$$j > i \quad \Rightarrow \quad C_j > C_i \quad .$$

Eine derartige Situation ist in Figur 5.2 dargestellt; die als Linie unterlegte Funktion soll den nichtlinearen Zusammenhang zwischen C_i und i andeuten. Zur Auswahl der richtigen Kapazität führen wir n Binärvariablen δ_i

$$\delta_i := \begin{cases} 1 \text{ , wenn die Größe } C_i \text{ für die Rohrleitung ausgewählt wird} \\ 0 \text{ , andernfalls} \end{cases}$$

ein. Fasst man die Binärvariablen zu einer SOS-1-Menge zusammen, braucht man diese Variablen nicht länger als Binärvariablen zu deklarieren. Die Konvexitätsbedingung (5.7.1) stellt sicher, das genau ein Wert der Menge $\{C_1, ..., C_n\}$ ausgewählt wird. Der Wert der diskreten Variablen c, also die aktuell gewählte Größe, kann durch

Abbildung 5.2 SOS-1-Variablen: Selektion einer Kapazität.



$$c = \sum_{i=1}^{n} C_i \delta_i \tag{5.7.2}$$

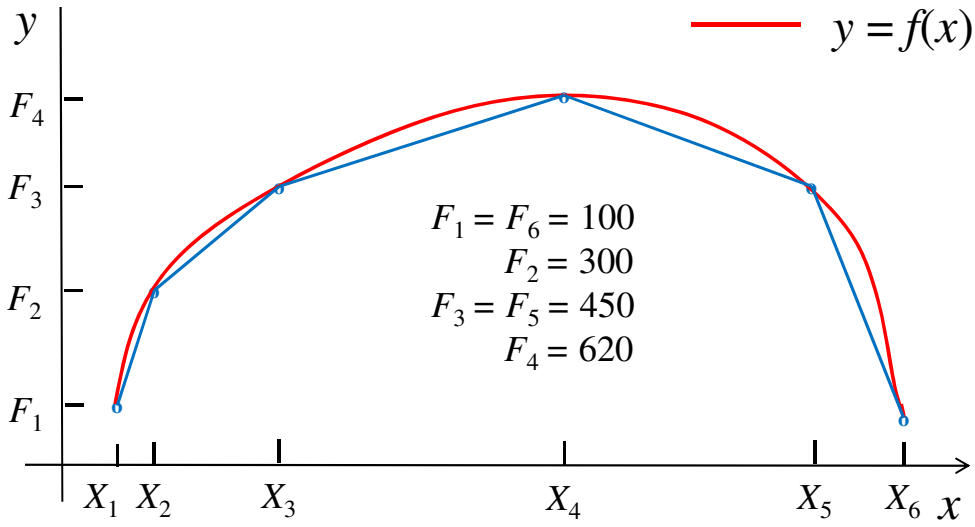
berechnet werden; die Gleichung (5.7.2) eignet sich hervorragend als Referenzbedingung.

Im Abschnitt 5.13.3 wird gezeigt, wie mit Hilfe von SOS-1-Mengen bestimmte Probleme schneller als mit Hilfe von Binärvariablen gelöst werden können, wenn im B&B-Verfahren die Ordnungsrelation ausgenutzt werden kann.

5.7.2 Strukturierte Variablenmengen vom Typ 2 (SOS-2-Mengen)

Eine SOS-2-Menge [engl.: *special ordered set of type* 2] ist eine geordnete Menge von Variablen, gewöhnlich als $\{\lambda_1, \lambda_2, ..., \lambda_n\}$ bezeichnet, von denen höchstens zwei von Null verschiedene Werte annehmen dürfen; wenn zwei Variablen von Null verschieden sind, dann müssen sie benachbarte Indizes haben. Analog zu SOS-1-Mengen kann auch von den Variablen einer SOS-2-Menge $\{\lambda_1, \lambda_2, \ldots, \lambda_n\}$ gefordert sein, der Konvexitätsbedingung (5.7.1) zu genügen. Eine relaxierte Lösung im Modell könnte $\lambda_2 = 0.75$ und $\lambda_3 = 0.25$ (alle übrigen λ's haben den Wert Null) sein oder, wenn nur eine einzige Variable verschiedenen von Null ist, $\lambda_3 = 1$. SOS-2-Mengen werden meist zur Modellierung nichtlinearer Ausdrücke verwendet. Ein derartiges Beispiel ist in Figur 5.3 dargestellt.

Abbildung 5.3 Modellierung einer nichtlinearen Funktion mit Hilfe von 6 Stützstellen X_1 bis X_6 und SOS-2-Variablen.



In diesem Fall hängt die Funktion $y = f(x)$ nichtlinear von x ab. Da LP- und MILP-Modelle nur lineare Beziehungen zwischen den Variablen erlauben, wird hier eine Approximation erforderlich sein. Hierzu seien in Abbildung 5.3 die fünf geraden Liniensegmente betrachtet, die die vorgegebene Kurve durch fünf Segmente mit linearen Interpolanten approximiert; durch mehr Segmente kann die Genauigkeit erhöht werden.

Betrachtet man nun nur die lineare Interpolation, so kann diese mit Hilfe von SOS-2-Mengen exakt beschrieben werden, wobei jeweils ein Segment auswählt wird, in dem linear interpoliert wird, alle anderen Segmente aber *ausgeschaltet* werden. Hier sei an die Definition von SOS-2-Mengen erinnert: „sind zwei Variablen von Null verschieden, dann müssen sie hinsichtlich der Indizes benachbart angeordnet sein". Die beiden benachbarten Variablen eignen sich für diese Interpolation. Die Segmente in Abbildung 5.3 sind als SOS-2-Menge $\{\lambda_1, \lambda_2, ..., \lambda_6\}$ mit der Konvexitätsbedingung

$$\lambda_1 + \lambda_2 + ... + \lambda_6 = 1$$

modelliert worden. Die Referenzbedingung

$$l = \lambda_1 + 2\lambda_2 + ... + 6\lambda_6$$

etabliert eine monotone Ordnung im Problem. Die Kosten werden durch die Gleichung

$$y = 100\lambda_1 + 300\lambda_2 + 450\lambda_3 + 620\lambda_4 + 450\lambda_5 + 100\lambda_6$$

beschrieben. Somit könnte sich eine typische Lösung wie folgt ergeben:

$$\lambda_1 = 0 \quad , \quad \lambda_2 = 0.75 \quad , \quad \lambda_3 = 0.25 \quad , \quad \lambda_4 = \lambda_5 = \lambda_6 = 0$$

$$l = 2 \cdot 0.75 + 3 \cdot 0.25 = 2.25$$

und

$$y = 300 \cdot 0.75 + 450 \cdot 0.25 = 337.50 \quad ;$$

diese Lösung zeigt in Abbildung 5.3 bereits eine gute Approximation. Die im Sinne der SOS-2-Definition unzulässige Lösung $\lambda_1 = 0.5$, $\lambda_3 = 0.5$ liefert dagegen z. B. eine schlechte Approximation; daher der Bedarf für benachbarte, von Null verschiedene Variablen. Die Approximationsgüte könnte durch eine größere Anzahl von Punkten – Gitterpunkte oder Stützstellen [engl.: *breakpoints*] in der graphischen Darstellung genannt – in der SOS-2-Menge verbessert werden, aber mit zunehmender Punktzahl wird auch die Rechenzeit ansteigen. Zu beachten ist auch, dass die Segmente die Kurve in Bereichen geringer Krümmung besser approximieren als dort, wo eine starke Krümmung vorherrscht; deshalb ist eine adaptive, nicht gleichmäßige Verteilung bzw. Dichte der Stützstellen ratsam. Zu beachten ist außerdem, dass die Kurve durch Segmente systematisch überschätzt oder unterschätzt wird. Es bleibt dem Modellierer überlassen, zu entscheiden, ob für den Kontext der Anwendung eine Überschätzung oder Unterschätzung vorzuziehen ist; wie die Anwendung im Abschnitt 11.4 zeigt, erfordert die Verwendung von SOS-2-Mengen stets ein hohes Maß an Erfahrung.

Hier sei nun kurz zusammengefasst, wie sich mit SOS-2-Mengen nichtlineare Relationen durch lineare Interpolation approximieren lassen. Für eine nichtlineare Funktion $y = f(x)$ wird zunächst eine Menge von n Stützpunkten X_i definiert, zu denen die gehörigen Funktionswerte $F_i = f(X_i)$ berechnet werden und die aus n Variablen λ_i bestehende SOS-2-Menge definiert wird. Tritt irgendwo in unserem Modell $f(x)$ auf, so schreiben wir stattdessen y, fügen die im Argument x interpolierende[3] Gleichung

$$x = \sum_{i=1}^{n} X_i \lambda_i \tag{5.7.3}$$

hinzu, interpolieren im Funktionswert

$$y = \sum_{i=1}^{n} F_i \lambda_i \tag{5.7.4}$$

und berücksichtigen schließlich die Konvexitätsbedingung

$$\sum_{i=1}^{n} \lambda_i = 1 \quad .$$

Mit Hilfe von (5.7.3) als Referenz-Bedingung wird eine monotone Ordnung in unserem Modell etabliert. Für ein B&B-Verfahren wäre (5.7.4) keine gute Referenzbedingung, da die Koeffizienten F_i nicht unbedingt eine Reihenfolge oder eine Monotonie etablieren. Nur wenn $f(x)$ eine monotone Funktion ist, könnte sie als eine Referenzbedingung verwendet werden. Es lohnt sich daher immer, die Referenzbedingung sorgfältig zu wählen. Je ausgeprägter eine Monotoniebedingung ist, um so wirksamer ist die Indextrennung, was sich positiv auf das B&B-Verfahren auswirkt.

[3] Im allgemeinen bezeichnet *Interpolation* den Prozess, Zwischenwerte einer Größe zu berechnen, die für eine diskrete Menge von Argumentwerten bekannt und gegeben ist. In unserem Fall, da die Variablen λ_i Elemente einer SOS-2-Menge sind, wird genau zwischen zwei benachbarten Werten interpoliert.

Zu beachten ist, dass nichtlineare Funktionen, die in der Zielfunktion vorkommen, leicht und sicher mit SOS-2-Mengen modelliert werden können. Problematisch sind jedoch Gleichungen, die nichtlineare Funktionen enthalten. Da die nichtlinearen Funktionen durch lineare Interpolation ersetzt wurden, könnte es leicht passieren, dass Probleme unzulässig werden und keine Lösung haben.

Über die hier beschriebene Methode hinaus, die es ermöglicht, spezielle nichtlineare Optimierungsprobleme mit Hilfe von LP- oder MILP-Techniken zu lösen, gibt es spezielle Algorithmen zur Lösung von NLP- und MINLP-Problemen, von denen einige in Abschnitt 4.4 beschrieben sind.

SOS-1- und SOS-2-Mengen stehen in Zusammenhang mit dem früheren Konzept der *Separablen Programmierung*, welches 1963 von Miller ([212]) entwickelt wurde. Wenn die zu approximierende nichtlineare Funktion $f(x)$ bzw. die daraus resultierende stückweise lineare Funktion konvex ist, erfüllt die Lösung der LP-Relaxierung bereits die SOS-2-Mengenbedingung für die Variablen λ_i. Ist $f(x)$ nicht konvex, benötigt man die Verzweigungsregeln im Kapitel 5. Zwar kann auch eine allgemeine IP-Formulierung mit Binärvariablen angewendet werden, um die zu den SOS-2-Mengen analoge Relationen zu modellieren, aber diese sind bei weitem nicht so effizient wie die strukturausnutzenden Verzweigungsregeln von SOS-2-Mengen.

Manchmal lassen sich auch nichtlineare, von zwei Variablen abhängige Ausdrücke so transformieren, dass sie mit linearen Modellen beschrieben werden können. Hierzu sei das Produkt $y = x_1 x_2$ zweier Variablen x_1 und x_2 betrachtet. Mittels der Logarithmusfunktion können wir $x_1 x_2$ durch die lineare Relation $\log y = \log x_1 + \log x_2$ ausdrücken. Es werden nun weitere Variablen $u_1 = \log x_1$ und $u_2 = \log x_2$ eingeführt. Mit Hilfe entsprechender SOS-2-Mengen werden u_1 mit x_1 bzw. u_2 mit x_2 verknüpft; es folgt $\log y = u_1 + u_2$, woraus wiederum y berechnet wird. Wenn jedoch x_1 und x_2 über einen relativ grossen Wertebereich variieren, können die Ergebnisse infolge der unterschiedlichen Skalierung [siehe Abschnitt 5.11.2] der Variablen und ihrer Logarithmen recht ungenau werden. Eine alternative Methode zur Beschreibung von Produkten ist im Abschnitt 5.7.3 beschrieben, weitere Transformationen in Rebennack & Kallrath (2013,[242]).

5.7.3 Strukturierte Variablenmengen - Verknüpfte Mengen

Ketten verknüpfter strukturierter Variablenmengen [engl.: *linked ordered sets*] wurden von Beale & Forrest (1976,[28]) eingeführt und sind später in Beale & Daniel (1980,[27]) weiter entwickelt worden. Mit ihrer Hilfe können durch ein Paar verknüpfter SOS-1-Mengen Produktterme approximiert werden. Angenommen, es soll der Produktterm

$$z = x \cdot g(y) \tag{5.7.5}$$

modelliert werden, wobei $g(y)$ irgendeine gegebene nichtlineare Funktion einer kontinuierlichen Variablen y ist, und y nur einen der n Werte

$$Y_1, Y_2, ..., Y_n$$

annehmen kann. Außerdem sei x eine kontinuierliche Variable mit bekannten (endlichen) oberen und unteren Schranken

$$-\infty < X^- \leq x \leq X^+ < \infty \quad .$$

Alle Kombinationen möglicher Werte von x, y und z können durch Einführung von $2n$ nichtnegativen Variablen

$$\begin{aligned} \lambda_{11}, \lambda_{12}, ..., \lambda_{1n} &\geq 0 \\ \lambda_{21}, \lambda_{22}, ..., \lambda_{2n} &\geq 0 \end{aligned}$$

definiert, erfasst und durch die Gleichungen

$$\sum_{j=1}^{n} \lambda_{1j} + \sum_{j=1}^{n} \lambda_{2j} = 1 \tag{5.7.6}$$

$$y = \sum_{j=1}^{n} Y_j(\lambda_{1j} + \lambda_{2j}) \tag{5.7.7}$$

und den zusätzlichen Bedingungen

$$\lambda_{ij} \neq 0 \Rightarrow \lambda_{rs} = 0 \quad \forall\, s \neq j, \quad , \quad i, r \in \{1, 2\}, \quad j \in \{1, 2, ..., n\} \tag{5.7.8}$$

verknüpft werden. Dann folgt x aus

$$x = \sum_{j=1}^{n} X^{-}\lambda_{1j} + \sum_{j=1}^{n} X^{+}\lambda_{2j} \tag{5.7.9}$$

und z ist gegeben durch

$$z = \sum_{j=1}^{n} [X^{-}g(Y_j)]\lambda_{1j} + \sum_{j=1}^{n} [X^{+}g(Y_j)]\lambda_{2j} \quad . \tag{5.7.10}$$

Die Bedingungen (5.7.8) können durch eine Modifikation der Verzweigungsbedingungen im B&B-Verfahren explizit berücksichtigt werden [siehe Abschnitt 4.3.2.3]. Die Bedingungen (5.7.6), (5.7.7) und (5.7.8) garantieren, dass y einen der gültigen n Werte annimmt; ist z. B. $\lambda_{1k} + \lambda_{2k} = 1$, dann wird y den Wert Y_k annehmen. Die Bedingungen (5.7.6) und (5.7.9) stellen sicher, dass x einen Wert auf einer Geraden annimmt, die zwischen X^{-} und X^{+} auf dem Segment liegt; ist z. B. $\lambda_{1k} + \lambda_{2k} = 1$ ist, dann ist x durch $\lambda_{1k}X^{-} + \lambda_{2k}X^{+}$ gegeben und kann so alle Werte zwischen X^{-} und X^{+} annehmen. Schließlich führen (5.7.6), (5.7.10) und (5.7.8) dazu, dass der Produktterm für z richtig berechnet wird. Dies wird deutlich, wenn die Konstante $g(Y_k)$ als gemeinsamer Faktor (alle mit $j \neq k$ assoziierten Terme sind Null) faktorisiert wird.

Das folgende Beispiel mit der Funktion

$$g(y) = e^{y} + y^{2}$$

verdeutlicht das Konzept verknüpfter SOS-2-Mengen. Seien $n = 4$, $X^{-} = 3.4$, $X^{+} = 8.5$ und 2.0, 2.5, 3.0 und 3.8 die möglichen Werte für y. Damit lautet der Wertebereich der Funktion $g(y)$ {11.39, 18.43, 29.09, 59.14}. Führt man die λ-Variablen zusammen mit der Gleichung

$$\lambda_{11} + \lambda_{12} + \lambda_{13} + \lambda_{14} + \lambda_{21} + \lambda_{22} + \lambda_{23} + \lambda_{24} = 1$$

ein, so folgen daraus

$$y = 2.0(\lambda_{11} + \lambda_{21}) + 2.5(\lambda_{12} + \lambda_{22}) + 3.0(\lambda_{13} + \lambda_{23}) + 3.8(\lambda_{14} + \lambda_{24}) \quad , \tag{5.7.11}$$

$$x = 3.4 \sum_{i=1}^{4} \lambda_{1i} + 8.5 \sum_{i=1}^{4} \lambda_{2i} \tag{5.7.12}$$

und

$$\begin{aligned} z \quad = \quad & 3.4(11.39\lambda_{11} + 18.43\lambda_{12} + 29.09\lambda_{13} + 59.14\lambda_{14}) \\ & + 8.5(11.39\lambda_{21} + 18.43\lambda_{22} + 29.09\lambda_{23} + 59.14\lambda_{24}) \quad . \end{aligned} \tag{5.7.13}$$

In einer LP-Relaxierung könnte eine typische Lösung $\lambda_{13} = 0.4$, $\lambda_{23} = 0.6$ sein mit allen anderen λ Variablen Null, was $y = 3$ in (5.7.11), $x = 6.46$ in (5.7.12) und $z = 3.4(0.4)29.09 + 8.5(0.6)29.09 = 187.92$ in Gleichung (5.7.13) impliziert.

Tritt die Variable y in mehreren Produkttermen auf, so kann für jeden Produktterm eine SOS-2-Menge für die λ Variablen definiert werden; daher auch die Bezeichnung *linked ordered sets*.

5.7.4 Strukturierte Variablenmengen - Familien von SOS-Mengen

Wie im Abschnitt 5.7.3 beschrieben wurde, können in einem Modell mehrere SOS-2-Mengen auftreten. SOS-1- oder SOS-2-Mengen können auch in Familien verwendet werden, bei denen es Verknüpfungen zwischen den einzelnen Mengen gibt. Zu den häufigsten Anwendungsfällen zählen:

1. *Zeitliche Ordnungsbeziehungen* [engl.: *precedence constraints*]. In diesem Fall werden Zeiten modelliert, zu denen Ereignisse vorkommen dürfen. Seien zum Beispiel zwei Ereignisse betrachtet, deren Zeiten t_1 und t_2 durch

   $$t_1 = \lambda_1 + 2\lambda_2 + ... + n\lambda_n$$

   und

   $$t_2 = \mu_1 + 2\mu_2 + ... + n\mu_n$$

   gegeben sind, wobei $\{\lambda_1, \lambda_2, ..., \lambda_n\}$ und $\{\mu_1, \mu_2, ..., \mu_n\}$ entsprechend definierte SOS-1-Mengen sind. Die Forderung, dass Ereignis 2 mindestens eine Zeiteinheit später als Ereignis 1 beginnt, kann durch die Ungleichung

   $$t_1 + 1 \leq t_2$$

   beschrieben werden. Somit werden zwei SOS-1-Mengen verknüpft.

2. *Vermeidung zeitlicher Überlappungen*: Wie im vorangegangenen Beispiel können zwei SOS-1-Mengen verwendet werden, um die Zeiten zu modellieren, zu welchen zwei Ereignisse vorkommen dürfen. Wenn es verlangt ist, dass die zwei Zeiten um mindestens eine Einheit abweichen, dann fordern wir die verknüpfte Einschränkung

   $$t_1 \neq t_2 \quad .$$

3. Die SOS-2-Mengen modellieren eine Verknüpfung zwischen anderen Mengen im Sinne der *verknüpften strukturierten Mengen*.

Die beiden ersten Beispiele enthalten SOS-1-Mengen, das dritte Beispiel nur SOS-2-Mengen, es können aber auch SOS-1- und SOS-2 Mengen gemischt auftreten. Spezielle *Schnitte* können dem Modell hinzugefügt werden, um das B&B-Verfahren zu beschleunigen. Für eine vollständige Diskussion dieser Thematik sei auf Wilson (1990,[293]) verwiesen; in Abschnitt 5.8 wird diskutiert, wie Modellformulierungen durch das Hinzufügen zusätzlicher Ungleichungen verbessert werden können.

5.8 Verbesserte Modellformulierungen: Logische Ungleichungen

Die meisten exakten Verfahren zur Lösung von MILP-Problemen beginnen mit der LP-Relaxierung, bei der die Ganzzahligkeitsbedingungen der Variablen einfach ignoriert werden. Der Unterschied zwischen dem Wert der optimalen Lösung dieses LP-Problems und dem optimalen Lösungswert des MILP-Problems wird Ganzzahligkeitslücke [engl.: *integrality gap*] Δ genannt, die schon eingehend auf Seite 95 diskutiert wurde. Es ist wichtig, Δ so klein wie möglich zu halten, da dies das B&B-Verfahren beschleunigt. *Zulässige Ungleichungen* [engl.: *valid inequalities*, *cuts*] sind ein Mittel dazu.

Als Beispiel sei die Bedingung „Produkt 1 kann nur produziert werden, wenn die Produkte $2, 3, \ldots, n$ produziert wurden" betrachtet. Die Ungleichung

$$(n-1)\delta_1 \leq \delta_2 + \delta_3 + ... + \delta_n$$

genügt völlig, um diese Bedingung zu modellieren, aber sie ist numerisch schwächer als die $n-1$ Ungleichungen

$$\delta_1 \leq \delta_i \quad , \quad i = 2, \ldots, n \quad . \tag{5.8.1}$$

Im vorigen Fall mag δ_1 bei einer LP-Relaxierung immer noch den Wert $(n-2)/(n-1)$ annehmen, wenn eine der verbleibenden δ Variablen gleich Null ist. Im Gegensatz dazu würde (5.8.1) die Wertzuweisung $\delta_1 = 0$ erzwingen.

Die Abbildung von Implikationen bietet ebenfalls Raum für Modellverschärfungen. In der (5.1.12) wurde die Implikation $\Longrightarrow$ benutzt, um eine Entscheidung mit einer anderen in der Form $L_1 \Longrightarrow (L_2 \wedge L_3)$ bzw. durch

$$\delta_1 \leq \delta_2 \quad , \quad \delta_1 \leq \delta_3$$

zu verknüpfen. Es ist allerdings immer noch möglich, Produkt 2 *und* Produkt 3 zu produzieren, ohne dass Produkt 1 produziert wurde. Ist die Produktion von Produkt 2 *und* Produkt 3 nur möglich, wenn Produkt 1 produziert wurde, dann muss die Implikation $\Longrightarrow$ durch die Äquivalenz $\Longleftrightarrow$ ersetzt und das Modell um die Ungleichung

$$1 + \delta_1 \geq \delta_2 + \delta_3$$

erweitert werden.

Denkbar ist, dass, wenn statt der Äquivalenz ($\Longleftrightarrow$) nur die Implikation ($\Longrightarrow$) modelliert wird, das Modell nicht geschlossen ist und die Möglichkeit eines Fehlers in der Modelllogik bleibt. Allgemein ist es nützlich, bei einer Implikation ($\Longrightarrow$) zu betrachten, was gelten soll, wenn die linke Seite der Implikation **nicht** gilt und untersucht, ob nicht vielleicht doch eher die Äquivalenzrelation ($\Longleftrightarrow$) gefordert ist. Es ist wünschenswert, eine Äquivalenz ($\Longleftrightarrow$), wann immer möglich, anstelle einer Implikation ($\Longrightarrow$) zu benutzen. Die Zielfunktion mag in einigen Fällen die Implikation durch Ausschluss logischer

Unmöglichkeiten zu einer Äquivalenz führen, aber darauf sollte man sich nicht unbedingt verlassen. Betrachten wir die Formulierung der *Implikationstabelle* (5.1.12), so können wir daraus die folgende *Äquivalenztabelle*

Relation	Ungleichungen
$L_1 \Longleftrightarrow (L_2 \wedge L_3)$	$\delta_1 \leq \delta_2 \quad , \quad \delta_1 \leq \delta_3 \quad , \quad 1 + \delta_1 \geq \delta_2 + \delta_3$
$L_1 \Longleftrightarrow (L_2 \vee L_3)$	$\delta_2 \leq \delta_1 \quad , \quad \delta_3 \leq \delta_1 \quad , \quad \delta_1 \leq \delta_2 + \delta_3$
$(L_1 \wedge L_2) \Longleftrightarrow L_3$	$\delta_3 \leq \delta_2 \quad , \quad \delta_3 \leq \delta_1 \quad , \quad \delta_1 + \delta_2 \leq 1 + \delta_3$
$(L_1 \vee L_2) \Longleftrightarrow L_3$	$\delta_1 \leq \delta_3 \quad , \quad \delta_2 \leq \delta_3 \quad , \quad \delta_1 + \delta_2 \geq \delta_3$

ableiten. Der Ausdruck $\bigwedge_{i=1}^{k} L_i \Longrightarrow \bigvee_{i=k+1}^{n} L_i$ lässt uns fragen, ob nicht eher $\bigwedge_{i=1}^{k} L_i \Longleftrightarrow \bigvee_{i=k+1}^{n} L_i$ gefordert ist. Falls dem so ist, so kann dies durch die Ungleichungen

$$\sum_{i=k+1}^{n} \delta_i - \sum_{i=1}^{k} \delta_i \geq 1 - k \quad ; \quad \delta_i \geq \delta_j \quad , \quad \begin{array}{l} i = 1, \ldots, k \\ j = k+1, \ldots, n \end{array}$$

modelliert werden.

5.9 Verbesserte Modellformulierungen: Spezielle Schnitte

Problemabhängig lassen sich spezielle zulässige Ungleichungen [engl.: *valid inequalities*] oder *Schnitte* [engl.: *cuts*] ableiten, die einen Teil des Gebietes $S(LP_{\mathrm{R}})/C_{\mathrm{H}}$ – zulässiger Bereich der LP-Relaxierung abzüglich der konven Hülle – abtrennen und so zu einer schärferen LP-Relaxierung führen.

5.9.1 Schnitte für ganzzahlige und semi-kontinuierliche Variablen

Können wir in unserem Modell Ungleichungen der Form $x + A\alpha \geq B$ mit positiven Konstanten A und B und Variablen $x \in \mathbb{R}_0^+$ und $\alpha \in \mathbb{N}_0$ identifizieren, so können wir das Modell verschärfen, in dem wir die zulässige Ungleichung

$$x \geq [B - (C-1)A]\,(C - \alpha) \quad , \quad C := \left\lceil \frac{B}{A} \right\rceil = ceil\left(\frac{B}{A}\right) \quad , \tag{5.9.2}$$

hinzufügen, wobei C die kleinste ganze Zahl bezeichnet, die größer oder gleich dem Verhältnis B/A ist. Die Ungleichung (5.9.2) wirkt um so effizienter, je mehr sich C und B/A unterscheiden.

Existiert in unserem Modell eine semi-kontinuierliche Variable σ, z. B., $\sigma = 0$ oder $\sigma \geq 1$, die in einer Ungleichung

$$x + A\sigma \geq B \quad \Leftrightarrow \quad \frac{1}{B}x + \frac{A}{B}\sigma \geq 1$$

auftritt, so wirkt die Ungleichung

$$\frac{1}{B}x + \sigma \geq 1 \tag{5.9.3}$$

für unser Modell verschärfend.

Als weiteres Beispiel sei die gemischt-ganzzahlige Ungleichung

$$x \leq C\lambda \quad , \quad 0 \leq x \leq X \quad ; \quad x \in \mathbb{R}_0^+ \quad , \quad \lambda \in \mathbb{N}_0$$

mit der zulässigen, verschärfenden Ungleichung

$$x \leq X - G(K - \lambda) \quad ; \quad K := \left\lceil \frac{X}{C} \right\rceil \quad , \quad G := X - C\,(K-1) \tag{5.9.4}$$

aufgeführt. Die Ungleichung (5.9.4) ist um so nützlicher, je mehr sich K und X/C unterscheiden. Ein spezieller Fall, der jedoch häufig auftaucht, ist $\lambda \in \{0,1\}$. Schließlich sei noch das aus Wolsey (1998,[297], S.129) entnommene Beispiel

$$A_1\alpha_1 + A_2\alpha_2 \leq B + x \quad ; \quad x \in \mathbb{R}_0^+ \quad , \quad \alpha_1, \alpha_2 \in \mathbb{N}$$

erwähnt, das für $B \notin \mathbb{N}_0$ zu der unzulässigen Ungleichung

$$\lfloor A_1 \rfloor\, \alpha_1 + \left(\lfloor A_2 \rfloor\, \alpha_2 + \frac{f_2 - f}{1-f} \right) \leq \lfloor B \rfloor + \frac{x}{1-f} \quad ,$$

mit den Abkürzungen

$$f := B - \lfloor B \rfloor \quad , \quad f_1 := A_1 - \lfloor A_1 \rfloor \quad , \quad f_2 := A_2 - \lfloor A_2 \rfloor$$

führt.

5.9.2 Elimination unerwünschter Kombinationen von Binärvariablen

Betrachtet sei eine Menge von Binärvariablen $\boldsymbol{\delta} \in \{0,1\}^m$ des m-dimensionalen Einheitswürfels. Gesucht sei nun eine zulässige Ungleichung, die genau die Kombination $\boldsymbol{\delta}^i := \{\delta_j^i \mid j = 1, \ldots, m\} \in \{0,1\}^m$ bezüglich der Indexmengen $\mathcal{B}_1^i := \{j \mid \delta_j^i = 1\}$ und $\mathcal{B}_0^i := \{j \mid \delta_j^i = 0\}$ trennt; es sei $\left|\mathcal{B}_1^i\right| + \left|\mathcal{B}_0^i\right| = m$. Es lässt sich zeigen, dass die Ungleichung

$$\sum_{j \in \mathcal{B}_1^i} y_j - \sum_{j \in \mathcal{B}_0^i} y_j \leq \left|\mathcal{B}_1^i\right| - 1 \tag{5.9.5}$$

nur durch $\boldsymbol{\delta}^i$ und durch keine andere Kombination $\boldsymbol{\delta}^k \neq \boldsymbol{\delta}^i$ verletzt wird. Durch Hinzufügen der Ungleichung (5.9.5), sie sei *Binär-Cut* genannt, zu einem bestehenden Problem kann also eine bestimmte Kombination $\boldsymbol{\delta}^i$ von Binärvariablen gezielt abgetrennt werden. Hierzu ein Beispiel: Sei $\boldsymbol{\delta} \in \{0,1\}^3$ und die Kombination $\boldsymbol{\delta}^i = \{0,1,0\}$ soll ausgeschlossen werden. Dann gilt offensichtlich $\mathcal{B}_1^i = \{2\}, \mathcal{B}_0^i := \{1,3\}$, $\left|\mathcal{B}_1^i\right| = 1$ und der passende Binär-Cut lautet

$$y_2 - y_1 - y_3 \leq 0 \quad \Leftrightarrow \quad y_2 \leq y_1 + y_3 \quad .$$

Mehrere Binär-Cuts erlauben die Abtrennung weiterer Kombinationen.

5.10 Preprocessing

Preprocessing-Verfahren bereiten ein Problem und dessen Daten hinsichtlich der späteren Lösung vor; die Methoden können eher datentechnischer oder stärker mathematischer Natur sein. Sie umfassen insbesondere *Presolving-Techniken*,[4] die sprachlich nicht immer sauber von Preprocessing-Verfahren getrennt werden, aber eher tiefer in die mathematische Struktur eines Problems eingreifen und ein gegebenes Modell mit dem Ziel modifizieren, verkürzte Rechenzeiten zu erzielen, ohne dabei den zulässigen Bereich zu verändern. Zwar werden sie auch im Zusammenhang mit reinen LP-Problemen verwendet, aber wesentlich wichtiger sind sie bei MILP-Problemen. Wir wollen die Begriffe *Preprocessing* und *Presolve* eher synonym verwenden. Eine Auswahl verschiedener Preprocessing-Verfahren findet sich bei Johnson *et al.* (1985,[151]); ein neuerer Überblick bei Savelsbergh (1994,[255]) und schließlich der umfassende Überblick über Presolve-Verfahren von Andersen & Andersen (1995,[13]). Im Detail wollen wir beleuchten:

- Presolve (arithmetische Tests bei Nebenbedingungen und Variablen, Ableitung schärferer Schranken),
- Disaggregation von Nebenbedingungen,
- Koeffizientenreduktion sowie Clique- und
- Cover-Identifizierung.

Während des Preprocessings werden redundante Nebenbedingungen und Variablen eliminiert, Variablen und Nebenbedingungen an ihre Ober- oder Untergrenzen fixiert oder einfache Basislösungen bestimmt. Mit Kenntnis dieser Methoden können das Modell und seine Laufzeiteigenschaften erheblich verbessert werden. Aus diesem Grund wird anhand einfacher Beispiele gezeigt, wie Presolve-Verfahren funktionieren.

5.10.1 Presolve

Innerhalb des Presolvings sind besonders zwei Grundoperationen von Bedeutung: *arithmetische Tests* und *Verschärfung von Schranken* [engl.: *bound tightening*].

5.10.1.1 Arithmetische Tests

Mit Hilfe arithmetischer Tests lassen sich Variablen eliminieren und somit die Problemgröße reduzieren. Sind x_1 und x_2 binäre Variablen, so folgt z. B. aus der Ungleichung $3x_1 + x_2 \leq 1$ sogleich $x_1 = 0$.

Trotz sorgfältiger Modellbildung kann ein Modell redundante Nebenbedingungen enthalten. So ist z. B. bei gegebenen oberen (10, 8, 8) und Untergrenzen von (2,3,5) für x_1, x_2 und x_3 die Ungleichung

[4] Die Begriffe *Preprocessing* und *Presolve* werden oft synonym verwendet. Manchmal wird der Ausdruck *Presolve* allgemein für jene Verfahren verwendet, die versuchen, die Größe des Problems zu reduzieren und zu identifizieren, ob das Problem unbeschränkt oder unzulässig ist, ohne es dabei gleich lösen zu müssen. *Preprocessing* schließt *Presolve* ein, aber umfasst auch andere Techniken, die z. B. darauf abzielen, die MILP-Formulierung selbst zu verbessern.

$$2x_1 + x_2 - x_3 \leq 25 \tag{5.10.1}$$

redundant, da die linke Seite der Ungleichung nie den Wert 23 überschreiten kann, was aus folgender Betrachtung ersichtlich wird. Der Maximalwert, den $2x_1+x_2-x_3$ annehmen kann, lässt sich ermitteln, indem die Variablen an ihren oberen bzw. unteren Schranken fixiert werden, d. h.

$$\max_{x_1,x_2,x_3} (2x_1 + x_2 - x_3) = 2 \cdot 10 + 1 \cdot 8 - 1 \cdot 5 = 23 \quad .$$

Die Ungleichung (5.10.1) kann also problemlos entfernt werden.

Als nächstes Beispiel redundanter Bedingungen seien die drei Ungleichungen

$$0 \leq x_1 \leq b_1 \quad , \quad 0 \leq x_2 \leq \infty \quad , \quad x_1 - x_2 \leq b_2 \tag{5.10.2}$$

unter der Voraussetzung $0 < b_1 < b_2$ betrachtet. Da $x_1 \leq b_1 < b_2$ und $x_2 \geq 0$ ist, kann $x_1 - x_2$ maximal den Wert b_1 annehmen (nämlich für $x_2 = 0$). Die Ungleichung $0 \leq x_1 \leq b_1$ ist somit stärker als $x_1 - x_2 \leq b_2$ und $x_1 - x_2 \leq b_2$ kann deshalb entfallen.

Im vorliegenden Maximierungsproblem

$$\max \quad 2x_1 + x_2 - x_3 \quad ,$$

in dem alle Variablen den Kapazitätsungleichungen

$$\begin{array}{rcrcrcl} x_1 & + & x_2 & + & x_3 & \leq & 100 \\ 2x_1 & + & 5x_2 & + & 3x_3 & \leq & 200 \\ -3x_1 & + & x_2 & + & x_3 & \leq & 150 \end{array} \tag{5.10.3}$$

unterliegen und den Wert Null als untere Schranke haben, kann die Variable x_3, da sie in der Zielfunktion mit negativem Koeffizienten, in den $\leq$ Ungleichungen (5.10.3) jedoch mit positiven Koeffizienten auftritt, eliminiert werden, denn sonst würde x_3 ja nur Kapazitäten wegnehmen. In der optimalen Lösung gilt daher $x_3 = 0$.

Die Idee, in einem Problem redundante Bedingungen zu eliminieren, wird in Brearley *et al.* (1975,[44]) diskutiert; weitere Informationen finden sich in Tomlin & Welch (1983a,[275];b,[274]). Es ist nützlich, Redundanzen zu entfernen, wann immer dies möglich ist, da es den Speicherbedarf und die Rechenzeiten verringert. Ein Modell mit Redundanzen wird zwar keine „falschen" Lösungen geben, bietet aber Raum für Verbesserungen. Die meisten kommerziellen Optimierungspakete bieten Möglichkeiten, Redundanzen während der Presolvephase zu eliminieren.

Mit Hilfe arithmetischer Tests lassen sich auch *unzulässige Probleme* schnell als solche identifizieren. In diesen Tests werden die Nebenbedingungen so umgeformt, dass eine ausgewählte Variable x_j alleine auf der linken Seite erscheint, z. B.,

$$x_j \circ B_i \pm \sum_{\forall k, k \neq j} A_{ik} x_k \quad , \quad \forall i$$

wobei das Symbol $\circ$ die Relationen $\leq$ oder $\geq$ darstellt. Im Folgenden wird diese Idee auf die Ungleichungen

$$0 \leq x_1 \leq 1 \quad , \quad 7 \leq x_2 \leq 9 \quad , \quad x_1 + x_2 \leq 5$$

angewendet, wobei x_2 als zu betrachtende Variable ausgewählt wird und wegen $0 \leq x_1$ zu dem Ungleichungssystem

$$\begin{array}{lcl} x_2 & \geq & 7 \\ x_2 & \leq & 9 \\ x_2 & \leq & 5 - x_1 \leq 5 \end{array}$$

führt. Die erste und letzte Ungleichung sind offensichtlich widersprüchlich, und somit ist unser Problem *unzulässig*.

5.10.1.2 Verschärfung von Schranken

Wie wir bereits auf Seite 86 gesehen haben, kann eine Verschärfung der oberen Schranke der Ungleichung (2.2.2) $x_1 + x_2 \leq 3.5$ zu $x_1 + x_2 \leq 3$ dazu führen, dass bereits die Lösung der LP-Relaxierung allen Ganzzahligkeitsbedingungen genügt. In diesem Abschnitt wird gezeigt, wie der zulässige Bereich einer Variablen verkleinert werden kann, indem mehrere Nebenbedingungen gleichzeitig berücksichtigt werden.

Hierzu seien die Ungleichungen

$$L_i \leq \sum_j A_{ij} x_j \leq U_i \quad , \quad \forall i$$

mit den Schranken

$$X_j^- \leq x_j \leq X_j^+ \quad , \quad \forall j$$

betrachtet. Zunächst ist es möglich, iterativ die Extremwerte

$$E_i = \sum_j A_{ij} x_j \quad , \quad \forall i$$

der Ungleichungen zu berechnen, indem die Variablen in Abhängigkeit vom Vorzeichen von A_{ij} an ihren unteren oder oberen Schranken fixiert werden. Für jede Ungleichung wird jeweils eine Variable ausgewählt und E_i verbessert, um damit eine neue Schranke für die ausgewählte Variable abzuleiten. Falls die neue Schranke schärfer ist, wird die vorhandene Schranke ersetzt. Diese Prozedur wird auf alle Ungleichungen angewendet und terminiert, wenn a) während einer kompletten Iteration keine verbesserte Schranke gefunden wird oder b) die Anzahl der Iterationen einen vor-definierten Maximalwert erreicht. Um das Verschärfen von Schranken zu demonstrieren, seien die folgenden Schranken

$$\begin{array}{rcrcr} 0 & \leq & x_1 & \leq & \infty \\ 0 & \leq & x_2 & \leq & 2 \\ 4 & \leq & x_3 & \leq & 10 \end{array} \tag{5.10.4}$$

und die Ungleichungen

$$4 \leq 2x_1 + 4x_2 - 6x_3 \leq 10$$

betrachtet. Eine neue Untergrenze für x_1 lässt sich aus $4 \leq 2x_1 + 4x_2 - 6x_3$ bzw. dem äquivalenten Ausdruck

$$x_1 \geq 2 - 2x_2 + 3x_3$$

ableiten. Setzt man die gegeben Schranken (5.10.4) für x_2 und x_3 ein, z. B., $x_2 = X_2^+ = 2$ und $x_3 = X_3^- = 4$, so folgt die verbesserte untere Schranke $x_1 \geq 10$. Eine verbesserte obere Schranke folgt aus der Betrachtung von $2x_1 + 4x_2 - 6x_3 \leq 10$ bzw. äquivalent dazu

$$x_1 \leq 5 - 2x_2 + 3x_3 \quad .$$

Hierin verwenden wir die verfügbaren Schranken $x_2 = X_2^+ = 0$ und $x_3 = X_3^- = 10$, und folgern $x_1 \leq 35$. Somit lauten die aktuellen Grenzen von x_1

$$10 \leq x_1 \leq 35 \quad .$$

Erhält man für eine Variable x nach diesen Schritten z. B. die Schranken $1.62 \leq x \leq 3.73$, und handelt es sich bei x um eine ganzzahlige Variable, so kann dies weiter zu der Ungleichungskette $2 \leq x \leq 3$ verschärft werden.

Mit Hilfe des Konzepts verschärfter Schranken lässt sich nun auch besser verstehen, warum es wichtig ist, in Ungleichungen der Form

$$y \leq Y\delta \quad , \quad y \geq 0 \quad , \quad \delta \in \{0,1\} \tag{5.10.5}$$

die obere Schranke Y für y so klein wie möglich zu wählen. In Modell der LP-Relaxierung tritt δ in der relaxierten Form $0 \leq \delta \leq 1$ auf. Angenommen, die optimale, d. h. die kleinste obere Schranke für y sei $Y' < Y$ und die zu minimierende Zielfunktion enthalte einen Term $C\delta$, der mit der Entscheidung δ verbundene Rüstkosten beschreibt. Ergibt sich $y = Y'$ in der optimalen Lösung der LP-Relaxierung, d. h. y möchte so groß wie möglich werden, und wurde (5.10.5) verwendet, so folgt wegen der Präsenz des Terms $C\delta$ in der Zielfunktion $\delta = Y'/Y < 1$ (δ möchte so klein wie möglich werden), was zu zwei zusätzlichen Unterproblemen beim B&B-Verfahren führt. Wäre jedoch

$$y \leq Y'\delta \quad , \quad y \geq 0 \quad , \quad 0 \leq \delta \leq 1$$

verwendet worden, so hätte sich automatisch $\delta = 1$ ergeben.

5.10.2 Disaggregation von Nebenbedingungen

Disaggregation zielt darauf, logische Implikationen zwischen binären und anderen Variablen zu finden und diese in Form logischer Ungleichungen dem Modell hinzuzufügen. Dies ist z. B. möglich, wenn Implikationen der folgenden Art gegenwärtig sind: ist eine Binärvariable gleich Null oder Eins, dann soll eine kontinuierliche Variable den Wert ihrer unteren oder oberen Schranke X^- bzw. X^+ annehmen. Es ergeben sich also vier mögliche Implikationen, die auf die folgenden Ungleichungen

$$\begin{array}{lcllcl} \delta = 0 & \Rightarrow & x = X^- & \longrightarrow & x - X^- & \leq & (X^+ - X^-)\delta \\ \delta = 0 & \Rightarrow & x = X^+ & \longrightarrow & X^+ - x & \leq & (X^+ - X^-)\delta \\ \delta = 1 & \Rightarrow & x = X^+ & \longrightarrow & x - X^- & \geq & (X^+ - X^-)\delta \\ \delta = 1 & \Rightarrow & x = X^- & \longrightarrow & X^+ - x & \geq & (X^+ - X^-)\delta \end{array} \quad . \tag{5.10.6}$$

führen. Betrachten wir z. B. die Binärvariable $\delta \in \{0,1\}$ in den Ungleichungen

$$\begin{array}{rrcr} x_1 & & \leq & 100\delta \\ -x_1 & +x_2 & \leq & 20 \end{array} \quad , \quad \begin{array}{rcccr} 0 & \leq & x_1 & \leq & 100 \\ 20 & \leq & x_2 & \leq & 80 \end{array} \quad . \tag{5.10.7}$$

Um zu sehen, ob eine der in (5.10.6) aufgelisteten Implikationen anwendbar ist, wird überprüft, welche Gestalt die Ungleichungen für $\delta = 0$ bzw. $\delta = 1$ annehmen:

$$\delta = 1 \Longrightarrow \left\{ \begin{array}{rrcr} x_1 & & \leq & 100 \\ -x_1 & +x_2 & \leq & 20 \end{array} \right\}$$

und

$$\delta = 0 \Longrightarrow \left\{ \begin{array}{rcr} x_1 & \leq & 0 \\ x_2 & \leq & 20 \end{array} \right\} \Longrightarrow \left\{ \begin{array}{rcr} x_1 & = & 0 \\ x_2 & = & 20 \end{array} \right\} \quad .$$

Während $\delta = 1$ nicht impliziert, dass x_1 oder x_2 an ihren jeweiligen unteren oder oberen Schranken fixiert werden, führt $\delta = 0$ auf x_1 und x_2 an ihren Untergrenzen. (5.10.6) liefert für x_1 die logische Ungleichung $x_1 \leq 0$, die aber keine neuen Informationen liefert und nicht weiterhilft. Dagegen folgt für x_2 die logische Ungleichung

$$x_2 - 60\delta \leq 20 \quad , \tag{5.10.8}$$

die das ursprüngliche Modell erheblich verbessert. Dies wird ersichtlich, wenn man die ursprünglichen Ungleichungen (5.10.7) zu

$$x_2 \leq 20 + x_1 \leq 20 + 100\delta$$

kombiniert. Die zusätzliche logische Ungleichung (5.10.8) führt zu der Verschärfung

$$x_2 \leq 20 + x_1 \leq 20 + 60\delta \quad .$$

5.10.3 Koeffizientenreduktion

Koeffizientenreduktion, ein weiteres Verfahren in der Gruppe der Preprocessing-Techniken, das zu verschärften MILP-Formulierungen führen kann, steht in Verbindung zum Rucksackproblem aus Abschnitt 2.2.7 und sei anhand der folgenden Ungleichung

$$4\delta_1 + 3\delta_2 - 2\delta_3 \leq 6 \tag{5.10.9}$$

erläutert. Allerdings ist (5.10.9) noch nicht in der üblichen Form des Rucksackproblems, da ein Koeffizient negativ ist, kann aber mittels der Transformation $\delta_i^{'} = 1 - \delta_i$ in die Standardrucksackform – mit dann positivem Koeffizienten +2 für δ_3' – überführt werden

$$4\delta_1 + 3\delta_2 + 2\delta_3' \leq 8 \quad . \tag{5.10.10}$$

Da eine solche Transformation stets möglich ist, kann also ohne Beschränkung der Allgemeinheit davon ausgegangen werden, dass alle Koeffizienten A_{ij} positiv sind und die zu betrachtende Ungleichung als Rucksackbedingung in binären Variablen erscheint. Für jede Ungleichung $\sum_j A_{ij}\delta_j \leq b_i$ (im Beispiel wird nur eine betrachtet) bezeichne

$$S_i := \sum_j A_{ij}$$

die Summe aller Koeffizienten. Sinnvollerweise kann angenommen werden, dass $S_i > b_i$ ist, da die Ungleichung andernfalls redundant ist. Ist A_{im} der maximale Koeffizient in der Ungleichung i, so schreiben wir die ursprüngliche Ungleichung in der separierten Form

$$A_{im}\delta_m + \sum_{j \neq m} A_{ij}\delta_j \leq b_i \quad . \tag{5.10.11}$$

Der Fall $A_{im} > b_i$ kann ebenfalls ausgeschlossen werden, da in diesem Fall notwendigerweise $\delta_m = 0$ gelten muss. Aus der Definition von S_i folgt

$$\sum_{j \neq m} A_{ij}\delta_j \leq \min\{S_i - A_{im}, b_i\} \quad . \tag{5.10.12}$$

Um die Ungleichung zu verschärfen, sei auf beiden Seiten der Term $(S_i - b_i)\,\delta_m$ hinzugefügt; nach entsprechender Anordnung der Terme nimmt (5.10.11) die Gestalt

$$\begin{aligned} (S_i - b_i)\,\delta_m + \sum_{j \neq m} A_{ij}\delta_j &\leq b_i + (S_i - b_i)\,\delta_m - A_{im}\delta_m \\ &= (S - A_{im} - b_i)\delta_m + b_i \end{aligned} \tag{5.10.13}$$

an. Zur Verschärfung der Ungleichung sei nun der Minimalwert der rechten Seite bestimmt. In der Annahme, dass $S_i - A_{im} < b_i$ ist, wird der Minimalwert, $S - A_{im}$, für $\delta_m = 1$ erreicht (man bemerke, dass der Term in Klammern infolge unserer Annahme negativ ist). Somit folgt schließlich die Bedingung

$$(S_i - b_i)\,\delta_m + \sum_{j \neq m} A_{ij}\delta_j \leq S_i - A_{im} \quad , \tag{5.10.14}$$

die die ursprüngliche Ungleichung $\sum_j A_{ij}\delta_j \leq b_i$ ersetzt. Es bleibt allerdings noch zu zeigen, dass der zulässige Bereich des ursprünglichen Problems nicht verändert wurde und keine ganzzahlig zulässigen Punkte ausgeschlossen werden. Wenn wir (5.10.14) für den Fall $\delta_m = 0$ prüfen, erhalten wir wieder (5.10.12); in diesem Fall ist also alles in Ordnung. Der Fall $\delta_m = 1$ in (5.10.11) liefert ebenfalls wieder das ursprüngliche Problem

$$A_{im} + \sum_{j \neq m} A_{ij}\delta_j \leq b_i \quad .$$

Somit ist gezeigt, dass (5.10.14) eine zulässige Ungleichung ist. Wegen $S_i - A_{im} < b_i$ oder äquivalent dazu $S_i - b_i < A_{im}$ ist festzustellen, dass der Koeffizient von δ_m und der der rechten Seite von (5.10.14) verglichen mit der ursprünglichen Ungleichung tatsächlich verringert wurde. Zusammenfassend lässt sich die folgende Regel aufstellen: wenn $S_i - A_{im} < b_i$ ist, können für die Koeffizienten A_{im} und b_i neue Werte festgesetzt werden

$$\begin{aligned} A_{im} &\longleftarrow S_i - b_i \\ b_i &\longleftarrow S_i - A_{im} \end{aligned}$$

und die ursprüngliche Ungleichung ersetzt werden. Daraufhin wird ein neuer Wert von S_i bestimmt und der Prozess wiederholt sich bei allen Koeffizienten bis keine Verschärfung mehr erzielt werden kann.

Wir fahren nun fort mit dem Beispiel und der Ungleichung (5.10.10)

$$4\delta_1 + 3\delta_2 + 2\delta_3 \leq 8 \quad , \tag{5.10.15}$$

wobei sich zunächst $S = 9$ ergibt. Mit $A_{11} = 4$ als den größten Koeffizienten folgen $S - A_{11} = 5 < 8 = b_1$, somit die verbesserten Koeffizienten

$$\begin{aligned} A_{11} &\longleftarrow 1 = S_1 - b_1 \\ b_1 &\longleftarrow 5 = S_1 - A_{11} \end{aligned}$$

und schließlich die verschärfte Ungleichung

$$\delta_1 + 3\delta_2 + 2\delta_3 \leq 5 \quad .$$

Der nächste Koeffizient, der ersetzt wird, ist A_{12} und schließlich A_{13}, so dass schließlich die Ungleichung

$$\delta_1 + \delta_2 + \delta_3 \leq 2$$

folgt, die besagt, dass höchstens zwei der Binärvariablen ungleich Null sein können. Im Sinne des Rucksackproblems garantiert diese Ungleichung die Wahl von maximal zwei Objekten. Betrachten wir die ursprüngliche Ungleichung (5.10.15), so erscheint das Ergebnis plausibel, da wir in der Tat höchstens zwei Objekte hätten wählen können.

5.10.4 Erzeugung von Clique-Ungleichungen

Betrachtet seien wieder n Binärvariablen $\delta_j \in \{0, 1\}$ und die Ungleichung

$$\sum_{j=1}^{n} A_j \delta_j \leq b \quad . \tag{5.10.16}$$

Wie bereits in Abschnitt 5.10.3 sei wieder angenommen, dass alle Koeffizienten A_j positiv sind und (5.10.16) eine Rucksack-Ungleichung darstellt. Die folgenden Überlegungen beziehen sich auf eine Ungleichung, die jedoch Teil einer Menge ähnlicher Ungleichungen sein können und auf die dieselbe Methode angewendet wird. Wir nehmen auch an, dass die Koeffizienten A_j der betrachteten Ungleichung in nichtaufsteigender Reihenfolge, also $A_1 \geq A_2 \geq \ldots \geq A_n$ sortiert sind. Eine *Clique* ist definiert als eine Ungleichung

$$\sum_{j \in S} \delta_j \leq 1 \quad , \quad S \subset \{1, \ldots, n\} \quad ;$$

wir verwenden auch die Bezeichnung *Clique*-Ungleichung. Zur Erzeugung einer *Clique* werden jeweils zwei aufeinanderfolgende Koeffizienten A_j und A_{j+1} beginnend mit $j = 1$ untersucht. Ist $A_j + A_{j+1} > b$, dann kann höchstens eine der beiden Variablen δ_j und δ_{j+1} den Wert 1 annehmen, da andernfalls (5.10.16) verletzt wird. Der Index j wird solange erhöht, bis für irgendein j^* schließlich $A_{j^*} + A_{j^*+1} \leq b$ gilt. Danach ist die Indexmenge $S = \{1, \ldots, j^*\}$ eine *Clique*, da jeweils nur eine der Variablen δ_{j_1} oder δ_{j_2} eines Indexpaares j_1, $j_2 \in S$ von Null verschieden sein kann. Eine Modellformulierung profitiert von verletzten *Cliquen* (schärfere LP-Relaxierung), so dass sinnvollerweise nur verletzte *Cliquen* als Ungleichungen zu einem Modell hinzugefügt werden. Anhand der Ungleichung

$$1.8\delta_1 + 1.5\delta_2 + \delta_3 + \delta_4 \leq 2 \tag{5.10.17}$$

sei die Methode erläutert.

Von links nach rechts folgend ist das erste Variablenpaar, das (5.10.17) nicht verletzt, δ_3 und δ_4, da $1 + 1 \leq 2$ ist. Somit ist $j^* = 3$ und wir fügen die *Clique*-Ungleichung

$$\delta_1 + \delta_2 + \delta_3 \leq 1$$

zu unserem Modell hinzu. Da δ_3 und δ_4 in gleicher Weise in (5.10.17) auftreten, könnte auch

$$\delta_1 + \delta_2 + \delta_4 \leq 1$$

als *Clique* hinzufügt werden.

5.10.5 Erzeugung von Cover-Ungleichungen

Ähnlich wie *Clique*-Ungleichungen stehen auch *Cover*-Ungleichungen in engem Zusammenhang mit binären Rucksackproblemen. Betrachtet seien wieder n Binärvariablen $\delta_j \in \{0,1\}$ und die Ungleichung

$$\sum_{j=1}^{n} A_j \delta_j \leq b \quad ,$$

in der alle Koeffizienten A_j positiv und diesmal in nichtfallender Reihenfolge – also $A_1 \leq A_2 \leq \ldots \leq A_n$ – angeordnet sind. Die Idee der *Cover*-Ungleichungen besteht darin, Binärvariablen zu finden, deren Summe der assoziierten Koeffizienten die Kapazität des Rucksacks überschreitet [vergl. hierzu auch das *Set-Covering-Problem* in Abschnitt 7.2]. Daher definieren wir eine *Cover*-Ungleichung [engl.: *Cover*] als die Ungleichung

$$\sum_{j} \delta_j \leq K \quad .$$

Finden wir eine Teilmenge $\mathcal{S} \subset \{1,\ldots,n\}$ von Indizes, sodass $\sum_{j\in\mathcal{S}} A_j > b$ und für jede echte Teilemenge $\mathcal{S}' \subset \mathcal{S}$, $\sum_{j\in\mathcal{S}'} A_j \leq b$ gilt, dann ist

$$\sum_{j\in\mathcal{S}} \delta_j \leq |\mathcal{S}| - 1 \tag{5.10.18}$$

eine *Minimal-Cover*-Ungleichung (Crowder *et al.* 1983,[53]). Ausgehend von (5.10.18) werden wir eine *Lifted-Cover-Ungleichung* als Lösung des Rucksackproblems

$$M := \max \sum_{j\in\mathcal{S}} \delta_j$$

unter der Nebenbedingung

$$\sum_{j\in\mathcal{S}} A_j \delta_j \leq b - A_k \quad , \quad \forall k \notin \mathcal{S}$$

herleiten. Ist $M > 0$, so wird δ_k zu (5.10.18) hinzugefügt [die Begriffe *aufgenommen* oder *erhoben* entsprechen dem englischen Begriff *lifted*], so dass (5.10.18) nun die Form

$$\sum_{j\in\mathcal{S}} \delta_j + (|\mathcal{S}| - 1 - M)\,\delta_k \leq |\mathcal{S}| - 1$$

annimmt. Falls $\delta_k = 0$, so ergibt sich gerade wieder (5.10.18); im Falle $\delta_k = 1$, folgt $\sum_{j\in\mathcal{S}} \delta_j \leq M$, was gerade aber der Definition von M entspricht. Die Erzeugung von Cover-Ungleichungen wird entsprechend auf alle anderen Ungleichungen angewendet. Das Verfahren wird nun anhand der folgenden Ungleichung

$$9\delta_1 + 10\delta_2 + 10\delta_3 + 11\delta_4 \leq 23 \tag{5.10.19}$$

erläutert. Zunächst muss eine geeignete Menge von Indizes $\mathcal{S}$ bestimmt werden. Hier bieten sich die vier Kandidaten {1,2,3}, {2,3,4}, {1,2,4} und {1,3,4} an, die allesamt die Ungleichung (5.10.19) verletzen und daher als Cover in Frage kommen. Der Einfachheit wegen sei die erste Indexmenge $\mathcal{S} = \{1,2,3\}$ gewählt, die auf den Minimal-Cover

$$\delta_1 + \delta_2 + \delta_3 \leq 2 \tag{5.10.20}$$

führt. Interpretiert man (5.10.20) als Rucksack-Ungleichung, so lässt sich feststellen, dass von den ersten drei Objekten höchstens zwei ausgewählt werden können. Der einzige Index, der nicht in $\mathcal{S}$ ist, ist $i = 4$. Auf diese Weise versucht die Lift-Technik δ_4 zum Minimal-Cover (5.10.20) hinzuzufügen; hierzu muss zunächst das Rucksackproblem

$$M = \max\{\delta_1 + \delta_2 + \delta_3\}$$

unter den Nebenbedingungen

$$9\delta_1 + 10\delta_2 + 10\delta_3 \leq 23 - 11 = 12$$

gelöst werden. In diesem Beispiel kann man leicht das Maximum $M = 1$ ableiten, da nur eine Binärvariable von Null verschieden sein kann. Somit kann δ_4 der Minimal-Cover-Ungleichung mit dem Koeffizienten $2 - M$ hinzugefügt werden, so dass sich

$$\delta_1 + \delta_2 + \delta_3 + \delta_4 \leq 2 \tag{5.10.21}$$

ergibt. Zu prüfen bleibt, ob und bei welchen Wertzuweisungen der Binärvariablen δ_i (5.10.21) verletzt wird. In der Tat zeigt sich, dass $\delta_1 = \delta_2 = \delta_3 = \delta_4 = 1$ keine zulässige Lösung von (5.10.21) ist. Dies ist nicht überraschend, da (5.10.21) gerade sicherstellt, dass höchstens zwei Binärvariablen ungleich Null sein können.

5.11 Effiziente Lösung von LP-Problemen

Die Lösung von LP-Problemen erlaubt zwar einige Feineinstellungen, die das Lösungsverhalten beeinflussen, ist aber nicht sehr sensitiv gegenüber diesen Einstellungen, vorausgesetzt, dass die Systemmatrix gut skaliert ist [siehe Abschnitt 5.11.2]. Es ist jedoch nützlich von einem *Warmstart* Gebrauch zu machen, wann immer dies möglich ist.

5.11.1 Warmstarts

Eine existierende Lösung eines Problems, das sich von einem nun zu lösenden Problem nicht allzu sehr unterscheidet, kann meist als Startbasis genutzt werden [siehe Abschnitt A.1.1 für eine Diskussion des Konzepts der *Basis*]. In diesem Fall spricht man von einem *Warmstart*. Wenn dies nicht möglich ist, wird der Gebrauch einer *Crashbasis* empfohlen. *Crash*-Techniken werden in Maros & Mitra (1996,[202]) diskutiert. Die Idee besteht darin, möglichst früh bestimmten, viel versprechenden Variablen bereits einen Lösungswert zuzuordnen, statt sie sukzessive und sequentiell in einem *Pricing*-Verfahren als Basisvariablen zu bestimmen. Die Auswertung der *Crash-Strukturen* kann die Lösung der LP-Probleme dramatisch beschleunigen; siehe z. B. Gould & Reid (1989,[115]).

5.11.2 Effiziente Skalierung

Die Skalierung der Systemmatrix eines LP- oder MILP-Problems kann durch das Verhältnis zwischen dem betragsmäßig größten und kleinsten Koeffizienten beschrieben werden.

Eine Matrix ist gut skaliert, wenn dieses Verhältnis klein ist, z. B. einen Wert zwischen 1 und 10 hat. Mit Hilfe einer gut skalierten Matrix lassen sich einige numerische Probleme vermeiden, so z. B. im Revidierten Simplexverfahren die Division großer durch kleine Größen, wodurch wiederum sehr große Größen entstehen. Zwar bieten kommerzielle Lösungsalgorithmen eine Vielzahl von Skalierungsmethoden, aber es ist doch eher ratsam, diverse Probleme durch eine sinnvolle Wahl der Variableneinheiten zu vermeiden. So ist es sicherlich sinnvoll, Gehälter in der Größenordnung von 10000 bis 20000 in Tausender-Einheiten zu führen, so dass ein Gehalt z. B. als 15.30 dargestellt wird, und im gleichen Modell auftretende Längengrößen im μm-Bereich, z. B. 0.0015mm besser als 1.5 darzustellen. Anderfalls führt der unsinnige Quotient beider Größen 15300.0 und 0.0015 auf den hohen Wert 10200000. Besonders kritisch wirkt sich numerisch die Differenz großer Zahlen aus, da es hier zur Auslöschung von Ziffern kommen kann. Ideal wäre es, wenn sich die Absolutwerte der von Null verschiedenen Matrixkoeffizienten zwischen 0.01 und 1000 bewegen, und die optimale Entscheidungsvariable den Wert 1000 nicht überschreitet. Ein vernünftiger Gebrauch der Maßeinheiten ist in jedem Falle sinnvoll: Die Verwendung von Gramm und Tonnen zur Beschreibung ähnlicher Mengen kann zu erheblichen Problemen führen. Ähnlich unklug ist es, Zielfunktionen in den monetären Grundeinheiten £, SFr, Euro, Rubel oder $ zu messen; sinnvoller ist es, diese in Einheiten von Tausend oder Millionen zu messen.

Einige Skalierungs-Varianten sind in Fulkerson & Wolfe (1962,[97]), Curtis & Reid (1972,[55]) sowie Tomlin (1975,[273]) beschrieben. Das Ziel ist stets, alle auftretenden Größen möglichst nahe an 1 heranzubringen, ohne dabei an Genauigkeit zu verlieren. Häufig führt die Skalierung von Zeilen und Spalten mit Hilfe der maximalen Elementmethode zu nützlichen Ergebnissen. Nachfolgend seien einige Möglichkeiten der automatischen Skalierung zusammengestellt:

1. Reihe (teilt jede Reihe durch das Element dieser Reihe mit größtem Absolutwert);
2. Spalte (teilt jede Spalte durch das Element dieser Spalte mit größtem Absolutwert);
3. Skalierung nach Curtis & Reid (1972,[55]);
4. geometrisches Mittel (ersetzt „größten absoluten Wert" von 1 oder 2 durch das geometrische Mittel.

Abschließend sei gesagt, dass insgesamt der Einfluss der Skalierung auf die Lösungszeit sehr schlecht verstanden ist. Bei nichtlinearen Problemen ist der Einfluss der Skalierung noch bedeutsamer. Es ist aber nicht nur eine Frage der Numerik, sondern auch der Modellierung an sich. Enthält beispielsweise ein Modell im Umfeld der Isotopenbestimmung Proben, deren Massen in der Größenordnung einiger tausendstel Gramm liegen, und gleichzeitig atomare Masseneinheiten in der Größenordnung von 10^{-23} Gramm, so ist dieser Unterschied in den Größenordnungen durch keine Skalierung aus der Welt zu schaffen. Spielen in der Wochen- und Monatsplanung logistischer Probleme Aspekte auf Minutenebene ein Rolle, so ist jede gute Skalierung dahin – und auch prinzipiell nicht möglich, wenn die Abbildungstreue diese Details wirklich erfordert.

5.12 Effiziente Modellierung - Gute Modellierungspraxis

Die Rechenzeiten bei der Lösung von MILP-Problemen können durch Beachtung einiger weniger Regeln, die *gute* von *schlechter* Modellierung unterscheiden, erheblich beeinflusst

werden. Die angeführten Beispiele zeigen, dass gute Modellierung eine enge Verknüpfung und ein tiefes Verständnis von Modellbildung und Lösungsalgorithmen erfordert.

1. Struktur der Zielfunktion: Werden verschiedene Kostenbeiträge (Transport, Produktion, Lager) aggregierten Variablen c^T, c^P und c^L zugeordnet und treten dann kompakt in der Zielfunktion in der Form $c = c^T + c^P + c^D$ auf, so kann dies zu erhöhten Rechenzeiten führen. Im Pricing-Verfahren kann sich dies nämlich nachteilig auswirken, da nur wenige Koeffizienten in der Zielfunktion auftreten, und somit die Auswahl von Basisvariablen in einer frühen Phase des Simplexverfahrens eher willkürlich sein wird. Presolve- und/oder *Crash-Verfahren* kommerzieller Software substituieren diese Terme in der Zielfunktion meist jedoch automatisch und die oben angeführten Einwände werden ungültig.

2. Vermeidung von Gleichungen, deren rechte Seite Null ist: Wenn möglich, sollten Gleichungen der Form $\sum A_j x_j = 0$ vermieden werden. Die Minimum-Verhältnis-Regel kann in diesem Falle nämlich Probleme damit haben, geeignete Basisvariablen im Simplexverfahren auszuwählen, die eliminiert werden können.

3. Bilanzgleichungen lassen sich auf zwei verschiedene Arten darstellen. Bezeichne x_1, x_2 und x_3 bzw. y_1 und y_2 die Strömungsmengen, die in einen Knoten einfließen bzw. ausfließen. Benötigt wird nun eine Ungleichung, die die Flusserhaltung beschreibt und sicherstellt, dass nicht mehr aus dem Knoten abfließt als einfließt, d. h.

$$x_1 + x_2 + x_3 - y_1 - y_2 \geq 0 \quad . \tag{5.12.1}$$

   Eine schärfere Form von (5.12.1) ist natürlich die exakte Flusserhaltung

$$x_1 + x_2 + x_3 - y_1 - y_2 = 0 \quad , \tag{5.12.2}$$

   die sicherstellt, dass einfließende und abfließende Mengen einander gleichen. Selbst wenn (5.12.2) die korrekte Formulierung der Flusserhaltung ist, so gibt es einige Vorteile bei der Verwendung von (5.12.1). Diese Relaxierung ist unter den folgenden Voraussetzungen zulässig: Man nehme an, dass y_1 und y_2 die Menge von Produkten darstellen, welche produziert und verkauft werden sollen und mit positiven Koeffizienten in einer zu maximierenden Zielfunktion auftreten. x_1, x_2 und x_3 seien Mengen von Vorprodukten, die in einem Produktionsnetzwerk gekauft oder als Zwischenprodukte produziert wurden. Von der numerisch-algorithmischen Seite besteht der Vorteil für die Verwendung von (5.12.1) darin, dass ein Optimierungsproblem mit größerem zulässigem Bereich gelöst wird, wodurch auch eine bessere Analyse der Unzulässigkeiten bei unkorrekt formulierten Modellen möglich wird. Wenn das Modell korrekt formuliert ist, dann sollte die Lösung in Übereinstimmung mit der wirtschaftlichen Situation die Eigenschaft $y_1 + y_2 = x_1 + x_2 + x_3$ haben, da es sich lohnt, bis zur vorgegebenen Nachfrage soviel wie möglich zu produzieren. Wenn diese Gleichung nicht erfüllt ist, „zerstört" der Lösungsalgorithmus nützliches Produkt, was zumindest beim Modellbilder Misstrauen hervorrufen sollte. Daten und Modell sollten reanalysiert werden, da offensichtlich etwas wirtschaftlich nicht gut oder falsch verstanden oder nicht richtig abgebildet wurde; manchmal kann eine Verschiebung des Kommas in einem Kostenwert die Ursache sein.

4. Bei der Lösung ganzzahliger Probleme ist es gelegentlich von Vorteil, zusätzliche ganzzahlige Variablen einzuführen, weil diese erlauben, Prioritäten auf Variablen zu setzen und somit die Variablenauswahl während des B&B-Prozesses zu beeinflussen. Die Einführung weiterer Variablen erlaubt auch weitergehende Verzweigungsstrategien auf Nebenbedingungen. Angenommen, wir führten eine zusätzliche ganzzahlige Variable α ein, die mit einigen Binärvariablen δ_i wie folgt verknüpft ist:

$$\alpha = \sum_i \delta_i \quad .$$

Angenommen, die LP-Relaxierung liefert einen fraktionalen Wert von $\alpha = 4.3$. Eine Verzweigung auf α ist äquivalent zu einer Verzweigung auf den Ungleichungen

$$\sum_i \delta_i \leq 4 \quad \text{oder} \quad \sum_i \delta_i \geq 5 \quad ,$$

was zu einer erheblichen Verbesserung führen kann.

5. Der Gebrauch von Prioritäten auf Variablen in der Verzweigungswahl: Betrachtet sei der Fall einer Chemiefirma, die einen grossen chemischen Reaktor entweder in Korea oder Brasilien bauen möchte. Der Reaktor wird entweder ein Batchreaktor oder ein kontinuierlicher Reaktor sein. Hier bietet es sich an, die nachstehend definierten Binärvariablen δ_{st}

$$\delta_{st} = \begin{cases} 1 \text{ , wenn ein Reaktor des Typs } t \text{ am Standort } s \text{ gebaut wird} \\ 0 \text{ , andernfalls} \end{cases}$$

einzuführen. Fragen wir uns jedoch, welche die wichtigere Entscheidung ist – der Reaktortyp oder der Ort? Da Korea und Brasilien eher unähnliche Länder mit verschiedener industrieller Infrastruktur, Märkten und Qualitäten von Kunstfertigkeit sind, ist die Wahl des Landes die wichtigere Entscheidung. Somit wäre es, um den Verzweigungsprozess zu kontrollieren, ein großer Vorteil, die Binärvariable

$$\alpha_s = \begin{cases} 1 \text{ , wenn irgendein Reaktor am Standort } s \text{ gebaut wird} \\ 0 \text{ , andernfalls} \end{cases} \quad .$$

einzuführen. Dies erlaubt uns nämlich, das B&B-Verfahren sich so verhalten zu lassen, wie wir das vernünftigerweise erwarten würden: zuerst die wichtigen Entscheidungen zu treffen, d. h. bei einer anstehenden Variablenwahl zunächst α_s zu wählen. Die Variablen δ_{st} und α_s werden durch

$$\delta_{st} \leq \alpha_s \quad , \quad \forall \{st\}$$

und

$$\alpha_s \leq \sum_t \delta_{st} \quad , \quad \forall s$$

verknüpft. Zu beachten ist, dass die Entscheidung $\alpha_s = 0$ die Implikation $\delta_{st} = 0$ für alle t nach sich zieht. Die Entscheidung $\alpha_s = 1$ erfordert dagegen $\sum_t \delta_{st} \geq 1$.

6. Relaxiert man Gleichungen, die ganzzahlige und kontinuierliche Variablen enthalten, indem man sie durch Ungleichungen ersetzt, so kann das B&B-Verfahren erheblich verbessert werden. Dies lässt sich mit Hilfe des folgenden Beispiels aus der chemischen Industrie gut verdeutlichen. Einige Einheiten u können eine Menge von Produkten p in Batches der Größen B_{up} produzieren. Dies führt zu den Gleichungen

$$B_{up}\beta_{upt} = x_{upt} \quad , \quad \forall\{upt \mid B_{up} > 0\} \quad , \tag{5.12.3}$$

die die Anzahl der Batches β_{upt} (übliche Werte liegen zwischen 0 und 20) mit der Gesamtmenge x_{upt}, die in der Zeitperiode t produziert wird, verbinden. Die Produktionsvariablen unterliegen ferner Beschränkungen für Kapazität und Verfügbarkeit und erscheinen in den Produktionsrezepten sowie der Zielfunktion. Das B&B-Verfahren konnte innerhalb von 20000 Knoten keine ganzzahlige Lösung finden. Die folgenden Überlegungen helfen zu verstehen, warum dieses Verhalten auftritt. Die wirtschaftlich relevanten Größen im Modell sind die Nachfragen; typische Werte sind einige hundert Tonnen. Die Nachfragen werden durch entsprechende Werte der Produktionsvariablen x_{upt} befriedigt. Bei einer optimalen Lösung nehmen die Variablen x_{upt} gewöhnlich irgendwelche kontinuierlichen Werte an. Daher werden die β_{upt} in der LP-Relaxierung und in den meisten Knoten des B&B-Baumes fraktional sein. Wenn zwei Probleme erzeugt werden, z. B., indem die Schranken $\beta_{upt} \leq \lfloor \beta_{upt} \rfloor$ oder $\beta_{upt} \geq \lceil \beta_{upt} \rceil$ hinzugefügt werden, ändern sich die Werte $x_{u'p't}$ in anderen Produktionseinheiten und für andere Produkte und produzierte neue fraktionale Werte $\beta_{u'p't}$ in der Lösung des Unterproblems. Die Lösung für dieses Problem besteht darin, die Gleichungen (5.12.3) aufzugeben, sie durch die Ungleichungen

$$B_{up}\beta_{upt} \geq x_{upt} \quad , \quad \forall u, p, t \mid B_{up} > 0 \tag{5.12.4}$$

zu ersetzen und das B&B-Verfahren anzuweisen, zunächst den Knoten $\beta_{upt} \geq \lceil \beta_{upt} \rceil$ zu untersuchen. Nun ist es für das B&B-Verfahren wesentlich einfacher, ganzzahlige Lösungen zu finden. Liefert eine LP-Relaxierung eine Lösung mit fraktionalem β_{upt}, dann hat das Unterproblem die gleiche Lösung, aber mit der Batchgröße $\lceil \beta_{upt} \rceil$. Um willkürlich große Werte zu vermeiden, werden diese Variablen in der Zielfunktion schwach bestraft (typischer Wert einer Zielfunktion ist oben 10^8 Euro, der geeignete Strafterm sollte von der Größenordnung 10^3 Euro, also etwa um einen Faktor 10^{-5} kleiner sein). Bedenkt man, dass die Kapazitätsbeschränkungen für die Produktionsvariablen verwendet werden, so könnte man argumentieren, dass die Batchgrößen, die mehr abdecken als was produziert wurde, die Produktionskapazität verletzen könnten. Dies stimmt in der Tat, und deshalb wurde die Ungleichung

$$B_{up}\beta_{upt} - x_{upt} \leq \varepsilon B_{up} \quad , \quad \forall\{u, p, t \mid B_{up} > 0\} \quad , \tag{5.12.5}$$

hinzugefügt, wodurch die Verletzung auf wenige Prozent ($0.01 \leq \varepsilon \leq 0.1$) der Batchgröße reduziert wurde. Daraufhin produzierte das B&B-Verfahren gute Lösungen nach 200 oder 300 Knoten. Wenn wir die Lösung betrachten, erkennt man, dass nur vier oder fünf von einigen hundert aus den Originalgleichungen (5.12.3) schwach verletzt wurden. Außerdem stimmten die spezifischen Verletzungen mit der Unsicherheit in den spezifizierten Kapazitäten überein.

5.13 Verzweigungsstrategien im Branch&Bound-Verfahren

Beale (1977,[26]) liefert eine detaillierte Beschreibung möglicher Verzweigungsstrategien. Kommerzielle Softwarepakete beinhalten in der Regel mehrere Methoden zur Kontrolle der Verzweigungsstrategien: benutzerspezifische Prioritäten und Pseudo-Kosten, erzwungene Verzweigungsrichtungen und Kontrolle des Akzeptanzwertes zur Reduzierung des Verzweigungsbaumes ein. Ist zu erwarten, dass die Verzweigung bezüglich einer bestimmten Variablen eine große Verbesserung in der Zielfunktion bewirken kann, so wird diese Variable als wichtig angesehen. Es könnte ein Vorteil sein, relativ wichtigen Variablen eine hohe Priorität zu geben. Diese werden dann für die Verzweigung in einem frühen Stadium mit dem Ziel einer Reduzierung der Gesamtrechenzeit ausgewählt: diese Art der Verzweigung erfordert mehr Rechenaufwand als andere und sollte deshalb nur einige Male an der Spitze des B&B-Baumes gemacht werden und nicht in den tieferen Regionen des Baumes. Pseudo-Kosten sind Einheitsraten der Zielfunktionsverbesserung, die benutzt werden, um den Effekt abzuschätzen, der sich ergibt, wenn bestimmte Ganzzahligkeitsbedingungen erzwungen werden. Schließlich ist es möglich, die Verzweigungsrichtung vorzugeben, aber dies ist wahrscheinlich von begrenztem Nutzen, wenn eine kommerzielle Software gute Verzweigungsstrategien implementiert hat.

5.13.1 Zielfunktion und Wahl des Akzeptanzwertes

Durch eine geschickte Kontrolle des Akzeptanzwertes Z^C der Zielfunktion kann der Benutzer die B&B–Suche erheblich unterstützen. Wenn alle Variablen, die in der Zielfunktion auftreten, ganze Zahlen sind, ist es oft möglich den minimalen Abstand Δ möglicher aufeinderfolgender Zielfunktionswerte Z_i zu berechnen; mit der Kenntnis von Δ kann eine geeignete Minimalverbesserung $\alpha = k\Delta$, $k \in \mathbb{N}$, gewählt werden, die wiederum erlaubt, den Akzeptanzwert $Z^C := Z_i + \alpha$ zu berechnen. Bei Verwendung eines von Null verschiedenen Wertes für α werden in den aktuellen Knoten des B&B-Baumes nur Zielfunktionswerte $Z \geq Z^C = Z_i + \alpha$ akzeptiert; alle übrigen werden eliminiert. Damit kann erheblich Rechenzeit gespart werden, da einerseits in einem Band der Breite α sämtliche LP-Relaxierungen, die noch fraktionelle Lösungswerte enthalten, eliminiert werden, anderseits durch geeignete Wahl von k bei der B&B-Suche weitere ganzzahlige Punkte nur akzeptiert werden, wenn sie zu erheblichen Verbesserung der Zielfunktion führen.

5.13.2 Verzweigungsheuristiken im B&B-Verfahren

Bei der Lösung von MILP-Problemen kann der Rechenaufwand erheblich beeinflusst werden, indem problemspezifische Informationen in das B&B-Verfahren eingebracht werden. Dies ist, wie schon formal in Abschnitt 4.3.2 beschrieben, aber hier noch einmal unter anderem Blickwickel und ausführlicher auf spezielle Verzweigungsobjekte zugeschnitten vertieft wird, in zweierlei Weise möglich:

1. Wahl des Verzweigungsobjektes und der -richtung: hierbei wird ein aktuelles Objekt (binäre Variable, ganzzahlige Variable, speziell-geordnete Menge, partiell-ganzzahlige Variable, oder halbstetige Variable), auf das verzweigt werden soll, und die Verzweigungsrichtung für die weitere Verzweigung gewählt;

2. Knotenwahl: hier wird aus der aktuellen Knotenliste der nächste Knoten, der untersucht werden soll, gewählt, sowie entschieden, wie Knoten eliminiert werden können; siehe hierzu Abschnitt 4.3.2.1.

Bei der Wahl des Verzweigungsobjektes kann das B&B-Verfahren durch vom Benutzer vorgegebene Prioritäten gesteuert werden. Verzweigungsobjekte mit hoher Priorität sollten solche sein, die, wie im Beispiel in Abschnitt 5.12 gezeigt, eine wichtige Entscheidung repräsentieren und konsequenzenreich sind. Führen diese Entscheidungen zu Knoten, die für ihre Auswertung viel Rechenzeit erfordern, so ist es sinnvoll, diese nur einige wenige Male in der oberen Region des B&B-Baumes zu bearbeiten. Betreffs der Knotenwahl ist es häufig von Vorteil, mit einer Tiefensuche so schnell wie möglich eine ganzzahlige Lösung zu bestimmen und damit große Teile des B&B-Baumes zu eliminieren.

Es gibt, wie in Abschnitt 4.3.2.2 ausgeführt, mehrere Möglichkeiten, die Variablenauswahl zu steuern. Die einfachste besteht darin, mit Hilfe von benutzerdefinierten Prioritäten die Variablen auszuwählen, die wesentliche Entscheidungen repräsentieren. Die Verzweigungen auf wichtige Variablen erfordern meist einen hohen Rechenaufwand. Daher ist es vorzuziehen, wenn lediglich an der Spitze des Baumes dieser Rechenaufwand auftritt und nicht später in tieferen Regionen. Jedem Verzweigungsobjekt wird eine Prioritätswert zugeordnet. Je kleiner dieser ist, desto wahrscheinlicher wird die Entität für Verzweigungen ausgewählt. Eine Binärvariable, die z. B. repräsentiert, ob ein Projekt finanziert wird, erhält logischerweise eine hohe Priorität, während eine Variable dagegen, die bereits nähere Details eines Modells beschreibt, nur eine niedrigere Priorität erhält.

Die Knotenwahl wurde ausführlich in Abschnitt 4.3.2.1 beschrieben. Hier sei daran erinnert, dass es oft es eine gute Strategie sein kann, mit einer Tiefensuche bei der B&B-Suche zu beginnen und einen guten ganzzahlig zulässigen Punkt so schnell wie möglich zu finden. Damit können bereits große Teile des Baumes eliminiert und das Verfahren erheblich beschleunigt werden. Allerdings gibt es auch Probleme, bei denen die Breitensuche gute Resultate erzielt. Neben der Priorität kann in einem B&B-Verfahren auch die Richtung der Verzweigung gewählt werden, z. B. ob zuerst der Zweig $\delta \leq 0$ oder der Zweig $\delta \geq 1$ untersucht wird.

5.13.3 Verzweigungsregeln für strukturierte Variablenmengen

Sehr nützliche Verzweigungsstrategien stehen für strukturierte Variablenmengen (siehe Abschnitt 5.7) und die in Abschnitt 5.13.4 besprochenen halbstetigen Variablen zur Verfügung. Im vorherigen Abschnitt wurden SOS-1-Mengen als Mengen von Elementen eingeführt, von denen höchstens eines einen von Null verschiedenen Wert annehmen kann und deren Elemente hinsichtlich ihrer Indizierung eine Ordnung aufweisen. Zum Beispiel könnte die Selektion der Größe von Rohren in Abschnitt 5.7.1 durch eine SOS-1-Menge beschrieben werden. Anstelle einer individuellen Verzweigung auf einzelne Variablen findet in diesem Falle eher eine Verzweigung auf eine ganze Menge statt.

Im Zusammenhang mit SOS-1- und SOS-2-Mengen können besondere Verzweigungsstrategien im B&B-Verfahren verwendet werden. Als Ergebnis einer LP-Relaxierung können verschiedene Variablen einer SOS-1- oder SOS-2 Menge fraktionale Werte annehmen. Ziel dieser speziellen Verzweigungsstrategien ist es, von Null verschiedene Werte hinsichtlich der Indizes „weit entfernter" SOS-1- bzw. SOS-2-Variablen zu vermeiden, indem geeignete Indexseparierungen vorgenommen werden. Hat eine SOS-1- oder SOS-2-Menge die Elemente $\lambda_1, \lambda_2, ..., \lambda_n$ und enthält die Lösung der LP-Relaxierung mehr als

ein von Null verschiedenes Element (oder mehr als zwei im Fall von SOS-2-Mengen, oder zwei nicht benachbarter Elemente), dann wird ein Paar benachbarter Variablen λ_r, λ_{r+1} gewählt, um mit Hilfe einer Referenzbedingung

entweder	sind alle Variablen λ_i mit $X_i < R$ auf den Wert Null gesetzt
oder	alle Variablen λ_i mit $X_i \geq R$ sind Null gesetzt

eine Indexseparierung vorzunehmen und die Verzweigung in der SOS-1- bzw. SOS-2-Menge zu initiieren.

Mit Hilfe des in Abschnitt 5.7.1 beschriebenen Beispiels kann der numerische Vorteil, der sich bei Verwendung von SOS-1-Mengen ergibt, dargestellt werden. In diesem Beispiel waren n verschiedene Kapazitäten C_i nach aufsteigendem Index [siehe Abb. 5.2] geordnet, von denen die passende Kapazität mit Hilfe von n Binärvariablen δ_i

$$\delta_i := \begin{cases} 1 \text{ , wenn die Größe } C_i \text{ für die Verbindungsleitung ausgewählt ist} \\ 0 \text{ , andernfalls} \end{cases}$$

ausgewählt werden soll. Diese δ_i Variablen formen eine SOS-1-Menge und es ist daher nicht mehr erforderlich, diese als binäre Variable zu erklären. Die Konvexitätsbedingung

$$\sum_{i=1}^{n} \delta_i = 1$$

garantiert, dass genau eine Größe gewählt wird. Daraus folgt

$$c = \sum_{i=1}^{n} C_i \delta_i \tag{5.13.1}$$

für die aktuell gewählte Größe. Die Gleichung (5.13.1) wird als Referenzbedingung verwendet. Zur Veranschaulichung sei $n = 4$ und $C_i = i$ gewählt und angenommen, dass die LP-Relaxierung folgende Lösungswerte

$$(\delta_1, \delta_2, \delta_3, \delta_4) = (0.47, 0.0, 0.0, 0.53)$$

für die δ_i Variablen erzeugt, aus der sich mit Hilfe von (5.13.1) der Wert

$$c = 1 \cdot 0.47 + 4 \cdot 0.53 = 2.59$$

für die Röhrengröße ergibt. Unter ökonomischen Gesichtspunkten wäre also die Röhrengröße 2.59 die beste Variante, was bedeutet, dass wir wahrscheinlich Größe 2 oder 3 haben werden. Ein konventionelles B&B-Verfahren würde nun wohl entweder auf δ_1 oder δ_4 verzweigen, da diese nicht ganzzahlig sind. Nehmen wir für den Moment an, dass die Variablenwahl δ_1 wählen wird. Die Verzweigungsrichtung $\delta_1 \geq 1$ impliziert $c = 1$, eine von $c = 2.59$ weit „entfernte" Lösung. Die Verzweigungsrichtung $\delta_1 \leq 0$ wird wahrscheinlich fraktionale Werte für δ_2 oder δ_3 ergeben. Wenn wir auf δ_4 verzweigen, ergibt sich eine ähnliche Situation; viel ist nicht gewonnen. Nutzt man bei der Verzweigung die Eigenschaften der SOS-1-Mengen, so werden die Indizes mit Hilfe des aus (5.13.1) errechneten Wertes c separiert und daraus zwei Unterprobleme erzeugt. In diesem Beispiel mit $c = 2.59$ führt die Separation auf

$$C_2 \leq c \leq C_3$$

und die beiden Unterprobleme

$$\delta_1 + \delta_2 = 0 \quad \text{und} \quad \delta_3 + \delta_4 = 0 \quad .$$

Im ersten Fall ist es wahrscheinlich, dass sich bei der Lösung des nächsten LP-Problems $\delta_3 = 1$ ergibt, während im zweiten Fall eher $\delta_2 = 1$ folgt. Nun sind wir in der Lage zu verstehen, warum die Ordnungsrelation bei SOS-1-Mengen eine wichtige Rolle spielt: die Ordnung wird benutzt, um alle Indizes in zwei disjunkte Indexmengen zu teilen. Je ausgeprägter die Ordnungsrelation ist, desto besser funktioniert diese Teilung.

Die meisten kommerziellen Softwarepakete unterstützen SOS-1- und SOS-2-Mengen, so dass die oben beschriebene Verzweigung automatisch, d. h. ohne Eingreifen des Benutzers, erfolgt. Allerdings kann es sinnvoll sein, dieses Verfahren durch geeignete Prioritäten [siehe Abschnitt 5.13.2] auf SOS-1- oder SOS-2 Mengen oder auch auf einzelne Variablen zu unterstützen.

5.13.4 Verzweigung auf halbstetigen und partiell-ganzzahligen Variablen

Halbstetige [engl.: *semi-continuous*] Variable können nur den Wert Null oder jede reelle Zahl zwischen einer unteren und einer oberen Schranke annehmen, d. h.

$$\sigma = 0 \quad \vee \quad L \leq \sigma \leq U \quad . \tag{5.13.2}$$

In einem B&B-Verfahren sind Verzweigungsregeln, die direkt auf dieser Struktur operieren, wesentlich effizienter als Verzweigungen auf Binärvariablen, die ebenfalls zur Modellierung halbstetiger Variaben verwendet werden könnten. Mit Hilfe der Binärvariablen δ und einer kontinuierlichen Variablen x kann (5.13.2) äquivalent durch die Ungleichungen

$$L\delta \leq x \leq U\delta \tag{5.13.3}$$

beschrieben werden. Die Implikationstabelle

$$\begin{array}{rcc} \delta = 0 & \Rightarrow & x = 0 \\ \delta = 1 & \Rightarrow & L \leq x \leq U \\ x = 0 & \Rightarrow & \delta = 0 \\ L \leq x \leq U & \Rightarrow & \delta = 1 \end{array}$$

zeigt, dass (5.13.3) in der Tat eine zu (5.13.2) gleichwertige Formulierung darstellt. Untersuchen wir nun, wie sich diese Formulierung und die binäre Variable δ während der Lösung der LP-Relaxierung (hier gilt nur $0 \leq x \leq U$) verhalten. In den meisten Modellen wird δ in der Zielfunktion mit einigen Koeffizienten auftreten, die evtl. Fix- oder Aktivitätskosten darstellen, während x positiv ist und eine zu produzierende oder zu verkaufende Menge darstellt. Daher wird δ den kleinstmöglichen Wert $\delta = x/U$ annehmen. Die Konsequenz ist, dass für jede binäre Variable, die eine halbstetige Variable darstellt, im B&B-Baum verzweigt werden muss. In der ursprünglichen Bedingung (5.13.2) sieht der LP-Algorithmus nur die relaxierte Ungleichung $0 < \sigma < U$ und die B&B-Logik arbeitet wie folgt: in den beiden Fällen $\sigma \geq L$ und $\sigma = 0$ ist nichts zu tun. Nur im Falle $0 < \sigma < L$ ist eine Verzweigung [$\delta \leq 0$ und $\delta \geq 1$] nötig. So führen effiziente Implementierungen halbstetiger Variablen zu einer (wesentlich) kleineren Anzahl von Verzweigungen. In einigen Tests, bei denen über 2500 halbstetige Variable verwendet wurden, ergab der

Vergleich zwischen (5.13.3) und alternativ (5.13.2) eine um einen Faktor 6 reduzierten CPU-Zeit zur Lösung eines Produktionsplanungsproblems, wenn halbstetige Variablen verwendet wurden.

Eine Verallgemeinerung von halbstetigen Variablen führt zu partiell-ganzzahligen Variablen, von denen bereits kurz auf Seite 11 die Rede war und die z. B. im folgenden Kontext auftreten: Ein Motorenhersteller möchte seine Fahrzeuge seinen Verteilzentren in Europa zuteilen. Natürlich möchte er ganzzahlige Werte in seinen Büchern führen, aber gerundete Lösungswerte, z. B. 5192 statt 5192.3 Autos für Italien scheinen zulässig zu sein. Allerdings ist der Fall 6.4 Autos für Andorra schon schwieriger zu entscheiden: sind nun 6 oder 7 Fahrzeuge zuzuteilen? Daher wird der Hersteller vielleicht verlangen, dass Zuteilungszahlen unter 50 ganzzahlig sein müssen, während für alle Werte größer als 50 eine einfache Rundung von Hand als zulässig betrachtet wird. Vorsicht ist jedoch geboten bei solchen Rundungsverfahren, wenn die zu rundenden Variablen in Bilanzgleichungen auftreten. Die können nämlich in solchen Fällen leicht verletzt werden. Dies zeigt das folgende Beispiel $x_1 + x_2 = y$ mit der kontinuierlichen Lösung $x_1 = 61.4$, $x_2 = 87.3$ und $y = 148.7$. Die gerundeten Werte 61, 87 und 149 erfüllen nämlich nicht die Bilanzgleichung, da $61 + 87 = 148 \neq 149$.

6 Lineare Optimierung in der Praxis

Dieses Kapitel enthält einige Fallstudien zur Linearen Optimierung, die aus realen Problemen hervorgegangen sind und erfolgreich gelöst werden konnten: die Optimierung der Produktion eines chemischen Reaktors, ein augenscheinliches nichtlineares Mischungsproblem, das sich unter Ausnutzung bestimmter Monotonieeigenschaften in ein lineares Problem transformieren lässt, und ein Verschnittproblem aus der Papierindustrie. Sämtliche Probleme besitzen Erweiterungen im Kontext ganzzahliger Optimierung.

Schließlich werden *Ziel-Programmierung* [engl.: *Goal Programming*], eine spezielle Technik zur Lösung multikriterieller Optimierungsprobleme, behandelt und einige Grenzen der linearen Programmierung diskutiert.

6.1 Produktionsplanung für einen Chemiereaktor

Eine Chemiefirma betreibt einen Reaktor, der in $M = 4$ verschiedenen Moden betrieben werden kann; die Moden unterscheiden sich z. B. in der Reaktortemperatur oder dem Druck. Je nach Modus m produziert der Reaktor $P = 4$ verschiedene Produkte p mit bekannten Ausbeutekoeffizienten Y_{mp}, die in der nachstehenden Rezepturtabelle zusammengefasst sind:

$m\backslash p$	1	2	3	4
1	61.0	12.1	9.5	4.3
2	61.0	17.0	9.0	7.6
3	61.0	8.0	6.0	3.2
4	61.0	10.0	6.0	2.3

.

Die Einheiten in dieser Tabelle sind Tonnen/Tag; die Einträge sind in der folgenden Weise zu interpretieren: für $m = 1$ produziert die Anlage z. B. 9.5 Tonnen/Tag von Produkt 3 und 12.1 Tonnen/Tag von Produkt 2.

Für jedes der Produkte p existiert eine Nachfrage D_p in Einheiten von Tonnen/Jahr:

p	1	2	3	4
D_p	20000	5800	4500	3000

.

Die Nachfrage muss nicht vollständig erfüllt werden, aber es ist auch nicht möglich, mehr zu verkaufen als durch D_p spezifiziert ist. Für jeden Reaktormodus m gibt es einen Nettogewinn von P_m kEuro/Tag wie nachfolgend zusammengestellt:

m	1	2	3	4
P_m	258	160	191	208

.

J. Kallrath, *Gemischt-ganzzahlige Optimierung: Modellierung in der Praxis*,
DOI 10.1007/978-3-658-00690-7_6, © Springer Fachmedien Wiesbaden 2013

Dieser Nettogewinn beinhaltet bereits die im voraus bekannten spezifischen Kosten[1] für den Reaktorbetrieb im Modus m sowie die Verkaufspreise der Produkte. Für den Planer stellt sich nun in diesem *Produkt-Portfolio-Optimierungsproblem* die Frage: An wie vielen Tagen soll der Reaktor in den jeweiligen Moden betrieben werden? Wegen Wartungsarbeiten steht der Reaktor nur an 330 Tagen/Jahr zur Verfügung.

Zur Beantwortung dieser Frage führen wir die Variablen $d_m \geq 0$ ein, die spezifizieren, an wie vielen Tagen der Reaktor in Modus m betrieben werden soll. Vielleicht könnte man erwarten, dass d_m nur ganze Zahlen annehmen darf. Es soll hier aber als hinreichend genau angesehen werden, mit rationalen Zahlen zu arbeiten. Zudem führen wir die Hilfsvariable[2] x_p ein, die uns sagt, wie viele Tonnen pro Jahr von Produkt p produziert werden sollen. Somit haben wir insgesamt 8 Variablen

$$d_m \geq 0 \quad , \quad m \in \{1, 2, ..., M\} \quad , \quad x_p \geq 0 \quad , \quad p \in \mathcal{P} := \{1, 2, ..., P\} \quad .$$

Zunächst formulieren wir die zeitliche Verfügbarkeitsbeschränkung für den Reaktor

$$\sum_{m=1}^{M} d_m \leq 330 \quad .$$

Die gesamt produzierte Menge x_p ergibt sich aus den Variablen d_m zu

$$x_p = \sum_{m=1}^{M} Y_{mp} d_m \quad , \quad \forall p \in \mathcal{P} \quad . \tag{6.1.1}$$

Da die Nachfrage nicht überschritten werden darf, aber auch nicht exakt erfüllt werden muss, folgt:

$$x_p \leq D_p \quad , \quad \forall p \in \mathcal{P} \quad . \tag{6.1.2}$$

Schließlich haben wir die Zielfunktion

$$\max_{\{d_m, x_p\}} \quad z = \sum_{m=1}^{M} P_m d_m \quad .$$

Im Prinzip könnte man die Variablen x_p durch Kombination von (6.1.1) und (6.1.2) vollständig eliminieren. Alle Informationen lassen sich direkt von d_m ableiten, denn sie sind die eigentlichen Entscheidungsvariablen. Somit lassen sich (6.1.1) und (6.1.2) durch

$$\sum_{m=1}^{M} Y_{mp} d_m \leq D_p \quad , \quad \forall p \in \mathcal{P}$$

ersetzen. Allerdings ist es recht nützlich, die Variablen x_p zu belassen, denn sie teilen uns sofort mit, wieviel insgesamt produziert wurde.

[1] In unserem stark vereinfachten Modell beinhalten die spezifischen Kosten bereits die Kosten für Rohmaterial, Energie, Wasser oder sonstige Ressourcen und Maschinen.

[2] Der Begriff *Hilfsvariable* ist natürlich etwas relativ. Er soll hier lediglich als Abgrenzung gegenüber denjenigen Variablen verwendet werden, die direkt eine Antwort auf die ursprünglich gestellten Fragen liefern.

Was können wir von dieser Fallstudie lernen? Das Beispiel ist sehr einfach gebaut und der Leser könnte sich fragen, wozu ein solch einfaches Beispiel behandelt wird. Das Beispiel soll zeigen, dass es in der realen Welt Probleme gibt, die zwar von einem mathematischen Standpunkt aus einfach sind, aber dennoch signifikante finanzielle Auswirkungen haben können. Zudem ist zu bedenken, dass dieses Beispiel erst nach intensiven Diskussionen mit dem Kunden diese einfache Gestalt angenommen hat. Dabei war es sehr wichtig, die relevanten Aspekte des Modells von den eher nebensächlichen Details, die nur zu unnötiger Komplexität führen würden, zu trennen und herauszuarbeiten, und schließlich eine Lösung zu entwickeln, die die finanzielle Situation des Kunden verbessert.

6.2 Verschnittprobleme

Verschnittprobleme [engl.: *trimloss problems*] bestehen in verschiedenen Industrien darin, eine feste Anzahl durch Länge und Breite definierter Muster bestimmter Materialien von kontinuierlichen endlich langen Rollen oder endlich langen Paletten aus Papier, Textilstoff, Glas, Holz oder Metall bekannter Breite rechtwinklig zuzuschneiden, und dabei den entstehenden Verschnitt aus unverwendbarem und überflüssigem Material zu minimieren. Modelle zur Lösung von Verschnittproblemen wurden von Gilmore & Gomory (1961,[105]) und in der Folgezeit von zahlreichen Autoren, insbesondere Dyson & Gregory (1974,[76]) entwickelt. Anwendungen in der Papierindustrie sind z. B. bei Vajda (1961,[277]), Anwendungen in der Glasindustrie in Dyson & Gregory (1974,[76]) beschrieben. In Farley (1991,[81]) wird ein Modell für den Zuschnitt von Photopapier beschrieben; schließlich findet sich in Abschnitt 11.2 ein weiteres Beispiel, in dem nicht nur die Anzahl der Muster, sondern auch das Design der Muster durch die Optimierung bestimmt wird.

6.2.1 Beispiel: Ein Verschnittproblem in der Papierindustrie

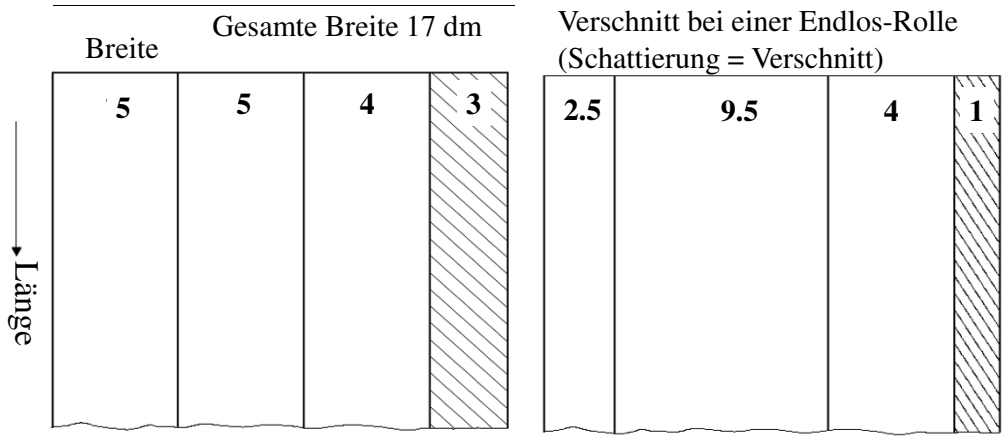
Gegeben sei eine Papierrolle unendlicher Länge und einer Breite von 20 Metern sowie durch $L \times B$ definierte Kundennachfrage von 30x5, 30x7 und 20x9 Metern. Hinsichtlich der Länge sind keine Unterbrechungen möglich, hinsichtlich der Breite sehr wohl.

Zur Modellierung dieses Problems muss zunächst entschieden werden, welche Kombinationen von der Rolle geschnitten werden können. Möglich wäre z. B. gleichzeitig zwei Lagen der Breite 7m und eine Lage der Breite 5m zu schneiden; der Rest der Breite 1m kann nicht weiter verwendet werden. Die Bestimmung der Menge aller Kombinationen ist ein Problem für sich und soll hier nicht beschrieben werden; allen zulässigen Kombinationen c ist gemeinsam, dass die einen Abfall [engl.: *waste*] oder Verschnitt W_m enthalten, der nicht weiter verwendet werden kann. Im vorliegenden Fall ergeben sich die sechs Muster [engl.: *pattern*] (Permutationen dieser Muster – also z. B. 7,5,7 – sind nicht aufgelistet):

m					W_m
1	5	5	5	5	0
2	5	5	7		3
3	5	5	9		1
4	5	7	7		1
5	7	9			4
6	9	9			2

.

Abbildung 6.1 Die Geometrie eines Verschnittproblems. Die Länge der Rollen ist hierbei als unendlich angenommen. Abgebildet sind zwei Muster für eine Masterrolle der Breite 17 dm.



Die auftretenden Breiten 5, 7 und 9 seien nachfolgend mit b ($b = 1, 2, 3$), die Muster mit m ($m = 1, 2, ..., 6$) indiziert. Zunächst seien die nichtnegativen, kontinuierlichen Variablen

$x_m \geq 0$: Länge der zu schneidenden Rolle bei Verwendung von Muster m
$s_b \geq 0$: Überschusslänge des Schnitts bei Verwendung der Breite b

eingeführt. Die Zielfunktion besteht in der Minimierung des gesamten Verschnitts

$$\min \quad (3x_2 + x_3 + x_4 + 4x_5 + 2x_6) + (5s_1 + 7s_2 + 9s_3) \quad . \tag{6.2.1}$$

Die Terme in der ersten Klammer beschreiben den direkten, durch die Kombination gegebenen, die Terme in der zweiten Klammer den aus der überschüssigen Länge resultierenden Verschnitt; die beiden ersten Kombinationen müssen mit einer Länge von 30m, die Kombination $c = 3$ mit einer Länge von 20m geschnitten werden.

1. Als erste Bedingung notieren wir: die Gesamtlänge der 5m breiten Schnitte (diese sind nur in den ersten vier Kombinationen möglich) muss mindestens 30m ergeben. Da einige Kombinationen mehrere Stücke einer bestimmten Länge ergeben und Überschusslängen anfallen können, führt dies auf

$$4x_1 + 2x_2 + 2x_3 + x_4 = 30 + s_1 \quad .$$

Analog führen die Bedingungen hinsichtlich der Gesamtlänge der 7m-Breiten (hier werden ebenfalls 30m gefordert) und der 9m-Breiten (hier werden 20m gefordert) auf

$$x_2 + 2x_4 + x_5 = 30 + s_2$$

und

$$x_3 + x_5 + 2x_6 = 20 + s_3 \quad .$$

Für das vorliegende, relativ kleine, Problem ist diese Modellformulierung geeignet; sie führt in echten Industrieanwendungen aber zu Schwierigkeiten, wenn die Anzahl der Kombinationen und damit der Variablen sehr groß wird.

Statt (6.2.1) kann alternativ auch die Gesamtschnittlänge

$$x_1 + x_2 + x_3 + x_4 + x_5 + x_6 \tag{6.2.2}$$

minimiert werden. Der Grund hierfür ist die Beobachtung, dass die Gesamtschnittlänge den nachgefragten Bedarf plus den Abfall abdecken muss. Da die nachgefragte Menge fest vorgegeben ist, ist die Minimierung der gesamten Schnittlänge aller Muster äquivalent zur Minimierung des Abfalls; im nächsten Abschnitt zeigen wir dies für eine ähnliche Situation auch algebraisch.

6.2.2 Beispiel: Ein ganzzahliges Verschnittproblem

Dieses Kapitel ist Fragestellungen gewidmet, die im Rahmen der LP behandelt werden können. Da das vorliegende Problem dem in Abschnitt 6.2.1 sehr ähnlich ist, wird es hier aufgeführt; zudem wird dadurch sichtbar, wie schnell ein LP-Problem zu einem MILP-Problem werden kann. Im vorliegenden Fall sollen 70 cm breite Papierrollen in den Breiten 12, 20 und 22 cm zugeschnitten werden; erlaubte Schnittmuster sind wieder bekannt. Dabei soll wöchentlich die folgende Nachfrage für Rollen befriedigt werden:

Breite	12 cm	20 cm	22 cm
Nachfrage	10	13	8

.

Wie soll die Ausgangsrolle zugeschnitten werden, wenn man den Verschnitt minimieren möchte? Im Modell seien die Indizes

$b \in \mathcal{B}$, $\mathcal{B} = \{1(12\text{ cm}), 2(20\text{ cm}), 3(22\text{ cm})\}$: Menge der Breiten
$m \in \mathcal{M}$, $\mathcal{M} = \{1, 2, ..., M\}$: Menge der Muster

verwendet. In das Modell fließen die folgenden Daten

D_b : Nachfrage für Rollen der Breite b
W_m : Verschnitt [in cm] der Muster m
N_{bm} : Anzahl der Rollen der Breite b, die sich aus einer einzelnen 70cm Rolle ergeben, wenn die Muster m verwendet wird
B_b : Breite [in cm] b

ein. Die Daten für N_{bm} sind (Muster, die aus Permutationen der aufgeführten Muster hervorgehen, sind hier nicht mehr separat aufgeführt)

	m	1	2	3	4	5	6	7	8	9	10	D_b
B_b	b											
12	1	5	4	4	2	2	2	0	0	0	0	10
20	2	0	1	0	2	1	0	3	2	1	0	13
22	3	0	0	1	0	1	2	0	1	2	3	8
W_m		10	2	0	6	4	2	10	8	6	4	

.

Die Muster – so sind sie konstruiert – erfüllen mit den ganzzahligen Variablen $\mu_m \in \{0, 1, 2, \ldots\}$, die beschreiben, wie viele Rollen nach Schnittmuster m zugeschnitten werden, die Bedingung

$$\sum_{b \in \mathcal{B}} N_{bm} \mu_m \leq B_{\mathrm{MR}} \quad , \quad \forall m \quad ,$$

wobei B_{MR} die Breite der Bestandsrollen bezeichnet; in diesem Fall 70 cm.

Im weiteren Verlauf bieten sich nun zwei Formulierungen an:

A) Können ungenutzte Breiten zurück ins Lager gehen (dies ist bei Standardgrößen, die häufig verwendet werden, meist der Fall), so ist der zu minimierende Verschnitt $\sum_{m \in \mathcal{M}} W_m \mu_m$, und zu erfüllen sind die Nachfragen

$$\sum_{m \in \mathcal{M}} N_{bm} \mu_m \geq D_b \quad , \quad \forall b \quad .$$

B) Können ungenutzte Breiten nicht weiterverwendet werden, so sind sie ebenfalls als Abfall zu betrachten, und die zu minimierende Zielfunktion kann wegen

$$\begin{aligned} \sum_{m=1}^{M} W_m \mu_m + \sum_{b=1}^{B} \left[\sum_{m=1}^{M} N_{bm} \mu_m - D_b \right] &= \sum_{m=1}^{M} \left(W_m + \sum_{b=1}^{B} N_{bm} \right) \mu_m - \sum_{b=1}^{B} D_b \\ &= \sum_{m=1}^{M} B_{\mathrm{MR}} \mu_m - D \\ &= B_{\mathrm{MR}} \sum_{m=1}^{M} \mu_m - D \end{aligned}$$

einfach – die Konstanten B_{MR} und $D := \sum_{b=1}^{B} D_b$ dürfen wir getrost außer Acht lassen – auf die Anzahl der Rollen

$$\sum_{m=1}^{M} \mu_m$$

reduziert werden. Dies hat den Vorteil, dass die Zielfunktion ganzzahlig ist und sich damit z. B. eine Minimalverbesserung von $\alpha = 1$ wählen lässt, der wie in Abschnitt 7.4.2 beschrieben, das B&B-Verfahren erheblich beschleunigen kann.

6.3 Ein scheinbar nichtlineares Mischungsproblem

In diesem Abschnitt wird eine Fallstudie diskutiert, die deutlich zeigt, in welcher Gestalt praktische Probleme auftreten können und wie wichtig die Modellierung ist. Zu Beginn des Projektes und ersten Gesprächen mit dem Kunden war nicht einmal klar, dass seine Fragestellung auf ein Optimierungsproblem führen würde, da er nach einem Expertensystem fragte, mit dessen Hilfe sich regelbasiert zulässige Mischungen erzeugen lassen. Das Projekt zeigte weiter, wie wichtig es für die Modellierung sein kann, in einem Projektteam mehrere Experten aus sehr unterschiedlichen Bereichen, hier der mathematischen Modellierung und dem physikalisch-chemischen Problemhintergrund, zu haben, die miteinander gut kommunizieren können.

Das Ziel des Projektes war das Design und die Produktion kostenminimaler Rezepturen für die Mischung von Flüssigkeiten, wobei diverse Produkteigenschaften, Nachfragebedingungen und einige logistische Randbedingungen (Zwangsverwendung von Produkten und Verfügbarkeitsrestriktionen) berücksichtigt werden mussten. Nachgefragt wurden typischerweise M Tonnen der gemischten Flüssigkeit, wobei sich die Produkteigenschaften Viskosität, Kochpunkt, und Alkoholkonzentration in den Schranken (η_-, η_+), (T_-^B, T_+^B) und (C_-^A, C_+^A) bewegen mussten. Außerdem waren die spezifischen Kosten K_i [Euro/Tonne] bekannt, die für die zu mischenden Komponenten i zu bezahlen waren, sowie die physikalischen und chemischen Eigenschaften der Komponenten.

6.3.1 Der Weg zum Modell

Angenommen, die Zusammensetzung der Mischung wäre bekannt: Wie könnten dann sämtliche übrigen Informationen abgeleitet werden bzw. was wäre dazu erforderlich? Wie kann also die Mischung charakterisiert werden? Hierzu bieten sich die dimensionslosen, relativen Massenanteile $x_i \geq 0$ der Komponenten i in der Mischung an. Die n Komponenten i müssen zunächst der Massenerhaltung

$$\sum_{i=1}^{n} x_i = 1 \tag{6.3.1}$$

genügen. Viele Phänomene der physikalischen Chemie lassen sich nun aber besser beschreiben, wenn man relativen Molmassenanteile w_i verwendet. Der Auftraggeber, selbst Chemiker, erklärt, wie die Molmassenanteile w_i mit den Massenanteilen x_i verknüpft sind:

$$w_i = \frac{x_i}{\mu_i} \Bigg/ \sum_{k=1}^{n} \frac{x_k}{\mu_k} \quad , \quad \forall i \quad , \tag{6.3.2}$$

wobei μ_i das Molekulargewicht der Komponente i bezeichnet, die der Normalisierung

$$\sum_{i=1}^{n} w_i = 1 \tag{6.3.3}$$

genügen. Dies kann durch Einsetzen von (6.3.2) in (6.3.3) unter Beachtung von (6.3.1) leicht überprüft werden. Betrachten wir, zumindest für den Moment, also $x_i \geq 0$ und $w_i \geq 0$ als unsere Entscheidungsvariablen. Um aus der Zusammensetzung auf alle übrigen Eigenschaften der Mischung schließen zu können, müssen wir in der Lage sein, die Stoffeigenschaften der Mischung aus den Eigenschaften der individuellen Komponenten abzuleiten. Auch hier weiß der freundliche Chemiker Rat. Die Viskosität berechnet sich gemäß der exponentiellen Relation

$$\eta = \prod_{i=1}^{n} \eta_i^{w_i} \quad , \tag{6.3.4}$$

während die Mischung des Alkohols linear in den Massenanteilen x_i ist, d. h.

$$C^A = \sum_{i=1}^{n} C_i^A x_i \quad . \tag{6.3.5}$$

Der Kochpunkt T^B der Mischung ist schwieriger zu berechnen, da er aus der impliziten Bedingung

$$P(T^B) = 1 \tag{6.3.6}$$

folgt, die besagt, dass der in der nicht mehr gebräuchlichen physikalischen Druckeinheit *physikalische Atmosphäre* (Abkürzung: *atm*) gemessene Dampfdruck im Kochpunkt genau den Wert Eins annimmt; 1 atm = 1013,25 hPa. Bei gegebener Temperatur T ist der Dampfdruck $P(T)$ eine Linearkombination der partiellen Dampfdrücke $P_i(T)$, d. h.

$$P(T) = \sum_{i=1}^{n} P_i(T) w_i \quad , \tag{6.3.7}$$

wobei $P_i(T)$ wiederum aus der Antoine-Gleichung [189]

$$P_i(T) = e^{A_i + B_i/(C_i+T)} \tag{6.3.8}$$

mit Materialkonstanten A_i, B_i und C_i berechnet werden kann.

Im Zusammenhang mit den logistischen Randbedingungen werden zudem die absoluten Mengen der beteiligten Komponenten benötigt. Diese ergeben sich aus

$$X_i = M x_i \quad , \quad \forall i \quad ,$$

wobei M die geforderte Menge (in Tonnen) der Mischung bezeichnet. Schließlich soll die Zielfunktion

$$Z = \sum_{i=1}^{n} (K_i M)\, x_i \tag{6.3.9}$$

minimiert werden, wobei K_i [Euro/Tonne] die spezifischen Kosten bezeichnet, die für die Komponenten bezahlt werden müssen.

6.3.2 Formulierung des Optimierungsproblems

Fassen wir die Ergebnisse aus dem vorherigen Abschnitt zusammen, so erhalten wir das lineare Optimierungsproblem

$$\min \quad Z = \sum_{i=1}^{n} (K_i M)\, x_i$$

unter den Nebenbedingungen

$$\sum_{i=1}^{n} x_i = 1 \quad , \quad \sum_{i=1}^{n} w_i = 1$$

und den Schranken

$$\begin{array}{ccccc} \eta_- & \leq & \eta & \leq & \eta_+ \\ C_-^A & \leq & C^A & \leq & C_+^A \\ T_-^B & \leq & T^B & \leq & T_+^B \end{array} \tag{6.3.10}$$

für Viskosität, Alkoholkonzentration und Kochpunkt der Mischung. Das System der Nebenbedingungen wird durch (6.3.1) und möglichen zusätzlichen Ungleichungen

$$\begin{aligned} x_i &\geq X^{MIN} \quad , \quad i \in \mathcal{J}_1 \\ x_i &\leq X^{MAX} \quad , \quad i \in \mathcal{J}_2 \end{aligned}$$

oder festen Wertzuweisungen für einige Komponenten

$$x_i = X_i^{FIX} \quad , \quad i \in \mathcal{J}_3$$

mit geeigneten Indexmengen $\mathcal{J}_1$, $\mathcal{J}_2$ und $\mathcal{J}_3$, die Teilmengen von $\{1, 2, ..., n\}$ sind, vervollständigt.

6.3.3 Analyse und Reformulierung des Modells

Die in (6.3.10) auftretenden Terme (6.3.4) und (6.3.7) lassen das Problem als ein nichtlineares beschränktes Optimierungsproblem erscheinen. Wie im Folgenden gezeigt, lässt sich das Problem aber als lineares Problem lösen.

Die Zielfunktion und die Alkoholkonzentration sind lineare Funktionen in x_i. Die Kombination von (6.3.5) und (6.3.10) ergibt

$$C_-^A \leq \sum_{i=1}^{n} C_i^A x_i \leq C_+^A \quad . \tag{6.3.11}$$

Als nächstes sei die Viskosität betrachtet. Hierbei sei daran erinnert, dass die Logarithmusfunktion *streng* monoton steigend ist, d. h.

$$x_1 < x_2 \Leftrightarrow \ln(x_1) < \ln(x_2) \quad . \tag{6.3.12}$$

Für unsere Zwecke genügt es aber, monoton wachsende Funktionen zu betrachten, also solche mit der Eigenschaft

$$x_1 < x_2 \Leftrightarrow f(x_1) \leq f(x_2) \quad . \tag{6.3.13}$$

Da die Logarithmusfunktion die Eigenschaft (6.3.13) besitzt, können die Viskositätsbedingung und (6.3.4) in eine lineare Ungleichungskette transformiert werden, indem beide Seiten von (6.3.4) und (6.3.10) logarithmiert werden und somit auf

$$\ln(\eta_-) \leq \sum_{i=1}^{n} \ln(\eta_i) w_i \leq \ln(\eta_+) \tag{6.3.14}$$

führen. Diese Transformation zeigt, warum die Monotonie wichtig ist. Unser Modell garantiert, dass (6.3.14) erfüllt ist, und mit (6.3.12) lässt sich dann wieder rückschließen, dass auch $\eta_- \leq \eta \leq \eta_+$ gilt.

Die Kochtemperatur T^B der Mischung soll innerhalb der Schranken T_-^B und T_+^B liegen. Wiederum hilft die Monotonie und etwas Physik. Die Schranken implizieren nämlich, dass der Dampfdruck bei der Temperatur T_-^B etwas kleiner, der bei T_+^B etwas größer als 1 bar ist, also $P(T_-^B) < 1$ und $P(T_+^B) > 1$. Die Modellierung der $>$ bzw. $<$ Ungleichung kann mit Hilfe eines kleinen Parameters $\varepsilon > 0$, z. B. $\varepsilon \approx 10^{-6}$ vorgenommen werden. Dann folgt

$$P(T_-^B) \leq 1 - \varepsilon < 1 < 1 + \varepsilon \leq P(T_+^B) \tag{6.3.15}$$

oder, mit (6.3.7)

$$\sum_{i=1}^{n} P_i(T_-^B) w_i \leq 1-\varepsilon \quad , \quad \sum_{i=1}^{n} P_i(T_+^B) w_i \geq 1+\varepsilon \quad .$$

Einzig bleibt noch, die Eliminierung der w_i und das Modell komplett in den Variablen x_i zu schreiben. Bezeichne $\circ$ eine der Relationen $\{\leq, \geq, =\}$. Eine Bedingung der Form

$$\sum_{i=1}^{n} F_i w_i \circ F^* \tag{6.3.16}$$

lässt sich unter Verwendung von (6.3.2) in die äquivalente Form

$$\sum_{i=1}^{n} \frac{F_i - F^*}{\mu_i} x_i \circ 0 \tag{6.3.17}$$

transformieren. Somit erscheint das Problem in der folgenden Form eines linearen Problems in den Variablen x_i

$$\min \quad Z = \sum_{i=1}^{n} (K_i T)\, x_i$$

unter den Nebenbedingungen

$$x_i \geq 0 \quad , \quad \forall i \quad ,$$

$$\sum_{i=1}^{n} x_i = 1 \quad ,$$

$$\sum_{i=1}^{n} \frac{\ln(\eta_i) - \ln(\eta_-)}{\mu_i} x_i \geq 0 \quad , \quad \sum_{i=1}^{n} \frac{\ln(\eta_i) - \ln(\eta_+)}{\mu_i} x_i \leq 0 \quad ,$$

$$\sum_{i=1}^{n} \frac{P_i(T_+^B) - (1+\varepsilon)}{\mu_i} x_i \geq 0 \quad , \quad \sum_{i=1}^{n} \frac{P_i(T_-^B) - (1-\varepsilon)}{\mu_i} x_i \leq 0 \quad ,$$

$$C_-^A \leq \sum_{i=1}^{n} C_i^A x_i \leq C_+^A \quad .$$

Schließlich kommen noch die logistischen Nebenbedingungen

$$Mx_i \geq X^{MIN} \quad , \quad i \in \mathcal{J}_1 \tag{6.3.18}$$

$$Mx_i \leq X^{MAX} \quad , \quad i \in \mathcal{J}_2$$

bzw. die Gleichungen für feste Beimischungen

$$Mx_i = X_i^{FIX} \quad , \quad i \in \mathcal{J}_3 \tag{6.3.19}$$

hinzu. Das Modell kann hinsichtlich der Numerik noch verbessert werden, indem man die Bedingungen (6.3.18)-(6.3.19) als Schranken formuliert, da das Simplexverfahren Schranken, wie in Anhang A.1.3 beschrieben, effizienter behandelt als lineare Nebenbedingungen. In diesem Fall ergeben sich obere und untere Schranken:

$$\begin{aligned} x_i &\geq X^{MIN}/M \quad , \quad i \in \mathcal{J}_1 \\ x_i &\leq X^{MAX}/M \quad , \quad i \in \mathcal{J}_2 \end{aligned}$$

bzw. für einige Komponenten

$$x_i = X_i^{FIX}/M \quad , \quad i \in \mathcal{J}_3 \quad .$$

Mit etwas Matrixalgebra können wir nun die maximale Anzahl der Komponenten in der Mischung berechnen. Bezeichnet $|\mathcal{J}_i|$ die Anzahl der Komponenten, die in der Indexmenge $\mathcal{J}_i$ enthalten sind, so ist die Anzahl n^P der Komponenten in der Mischung durch

$$n^P \leq m = 7 + |\mathcal{J}_1| + |\mathcal{J}_2| + |\mathcal{J}_3|$$

beschränkt, wobei m die Gesamtzahl der Nebenbedingungen im Modell benennt. Der Grund dafür ist, dass nicht mehr Basisvariablen als Nebenbedingungen im Modell sein können. Somit kann unserem Kunden bereits ohne einen einzigen Rechenlauf mitgeteilt werden, dass seine Mischungslösung höchstens n^P Komponenten beinhaltet.

6.4 Multikriterielle Optimierung und Goal Programming

Lineare Optimierung behandelt Problemstellungen mit einer eindeutig bestimmten Zielfunktion. In der Praxis trifft man jedoch häufig auf Fragestellungen, in denen widersprüchliche Ziele in einem Modell zu berücksichtigen sind. Eine Firma möchte z. B. einen Chemiereaktor so bauen, dass die Ausstoßraten maximal sind, gleichzeitig die Größe des Reaktors minimal wird und einige sicherheitsrelevante Parameter maximal werden. Diese Zielkriterien laufen einander entgegen; jede zulässige Lösung des Problems lässt sich hinsichtlich dieser Zielkriterien bewerten. Dieses Beispiel kann um weitere Beispiele ergänzt werden, in denen man neben der Deckungsbeitragsmaximierung z. B. die Gesamttransportmenge minimieren möchte, in denen eine Risikominimierung und eine Maximierung der Kapitalrendite konkurrieren, oder in denen die Liefertreue und die Sicherheitsbestandmenge nicht vereinbare Zielkriterien sind.

6.4.1 Multikriterielle Optimierung

Ein Zugang zur Lösung solch *multikriterieller Optimierungsprobleme*, die auch *Vektorminimierungsprobleme* genannt werden, besteht darin, alle Zielkriterien auf eine gemeinsam messbare Größe, z. B. Geld, abzubilden. Allerdings ist es nicht immer möglich, konkurrierende Ziele sinnvoll auf einer Skala abzubilden. Im Falle des Chemiereaktors ist es z. B. problematisch, die sicherheitsrelevanten Aspekte in Geld auszudrücken. Ein sinnvolles Konzept im Zusammenhang mit multikriterieller Optimierung ist die Betrachtung von *pareto-optimalen Lösungen*. Diese sind folgendermaßen charakterisiert: in einem multikriteriellen Optimierungsproblem mit n zu maximierenden Zielkriterien $Z_i(\mathbf{x})$, $i = 1, \ldots, n$ heißt eine zulässige Lösung $\mathbf{x}^*$ *pareto-optimale Lösung,* wenn es keine andere zulässige Lösung $\mathbf{x}$ gibt mit $Z_i(\mathbf{x}) \geq Z_i(\mathbf{x}^*)$ für alle i und $\max_i sign\{Z_i(\mathbf{x}) - Z_i(\mathbf{x}^*)\} = 1$, d. h. es gibt keine andere zulässige Lösung, die hinsichtlich aller Zielkriterien mindestens genauso gut ist, wie die Lösung $\mathbf{x}^*$ und nicht in wenigstens einem Zielkriterium besser ist. Ein spezieller Lösungszugang zu multikriteriellen Optimierungsproblemen besteht darin, zu verlangen, dass alle Zielkriterien bestimmte Zielmarken [engl.: *targets*] so gut wie möglich erreichen; evtl. einige exakt annehmen. So könnte in dem eingangs formulierten Problem gefordert sein, dass die Ausstoßraten so nahe wie möglich an 50 Tonnen/Stunde

heran reichen und der Reaktor möglichst in eine Fläche von 50 m mal 50 m hineinpasst. Diese Zielmarken werden in der angelsächsischen Literatur *targets* oder *goals* genannt. Die daraus zu konstruierende umfassende Zielfunktion trachtet danach, die Abweichung der Zielkriterien von ihren Zielmarken zu minimieren. Die aus diesem *Goal Programming* Ansatz bestimmten Lösungen sind pareto-optimal.

6.4.2 Ziel-Programmierung - Goal Programming

Goal Programming kann verstanden werden als eine Erweiterung der Linearen Optimierung, bei der bestimmte Ziele bzw. Schranken für einen speziellen Satz von Nebenbedingungen vorgegeben sind. Hierbei lassen sich zwei Varianten unterscheiden:

1. die *Archimedische* Variante; und
2. die *lexikographische* (in der angelsächsischen Literatur auch *pre-emptive*) Variante.

In der Archimedischen Variante werden Abweichungen von den Zielmarken mit Hilfe von Gewichten oder Straftermen berücksichtigt und somit gewichtete Summen von Zielfunktionen verwendet. Das folgende Beispiel mag dies verdeutlichen. Das erste Zielkriterium sei der Gewinn, der durch $2x+3y$ dargestellt werden kann, das zweite Zielkriterium sei die durch $y+8z$ beschriebene Kapitalrendite; zusätzlich zu den Variablen x, y und z mögen weitere Variablen und Nebenbedingungen in das Modell einfließen. Die Zielmarken seien G_1 (Wunschmarke für den Gewinn) und G_2 (Wunschmarke für die Kapitalrendite); diese Werte für das vorliegende Jahr könnten sich z. B. aus den Ergebnissen des Vorjahres mit bestimmten Prozentaufschlägen errechnen.

Die Abweichungen von den Zielmarken können mit Hilfe der nichtnegativen Variablen d_1^+, d_1^-, d_2^+, $d_2^- \geq 0$ in der folgenden Weise beschrieben werden:

$$\begin{array}{rrrrrcll} 2x & +3y & & +d_1^+ - d_1^- & & = & G_1 \quad , & \text{goal 1} \\ & y & +8z & & +d_2^+ - d_2^- & = & G_2 \quad , & \text{goal 2} \end{array} \quad .$$

Infolge des Auftretens der Hilfsvariablen d (entgegengesetze Vorzeichen in der Nebenbedingung, gleiche Vorzeichen in der Zielfunktion weiter unten) kann nur höchstens eine der beiden Variablen d_1^+ und d_1^- in der optimalen Lösung Basisvariable und somit größer Null werden; das gleiche Argument gilt für d_2^+ und d_2^-. Die globale Zielfunktion, die die Minimierung der Abweichungen von den Zielmarken beschreibt, nimmt somit die Gestalt

$$\min \quad d_1^+ + d_1^- + d_2^+ + d_2^-$$

an und führt auf ein gewöhnliches lineares Optimierungsproblem; die Berücksichtigung von mehr als zwei Zielkriterien ist offensichtlich.

Bei dieser Formulierung werden die beiden Zielkriterien als gleich wichtig angesehen, d. h. ein Überschuss im ersten Zielkriterium um den Wert 100 könnte einem Unterschuss im zweiten entsprechen; in einem Problem mit fünf Zielkriterien ist ein Überschuss von 40 in einem Kriterium gleichwertig einem Unterschuss von jeweils 10 in den anderen Kriterien. Selbst wenn die Zielkriterien auf kompatible Einheiten abgebildet wären, erscheint diese Situation nicht wünschenswert. Vermieden werden kann sie durch die beiden folgenden Vorgehensweisen:

(a) Einführung oberer Schranken auf die Werte der Variablen d_i, z. B.,

$$d_1^+ \leq 10 \quad , \quad d_1^- \leq 10 \quad , \quad d_2^+ \leq 5 \quad , \quad d_2^- \leq 5 \quad ,$$

wodurch die Abweichungen von den Zielmarken in vernünftigen Grenzen bleiben;

(b) Einführung von Gewichten in der Zielfunktion. So kann die Zielfunktion

$$d_1^+ + d_1^- + 10d_2^+ + 10d_2^-$$

z. B. so interpretiert werden, dass den Abweichungen in der Kapitalrendite eine 10 mal größere Bedeutung beigemessen wird als den Abweichungen im Gewinn.

In der *lexikographischen Variante* werden die Zielkriterien nach Bedeutung und Priorität geordnet, wobei – in der angegebenen Ordnung – das Zielkriterium i als unendlich wichtiger angesehen wird als das Zielkriterium $i+1$; diese Gewichtung kann durch prozentuale oder absolute Angaben, die quantifizieren, wieweit ein bestimmtes Zielkriterium von seinem spezifischen Optimum abweichen darf, relaxiert werden. Im Reaktorbeispiel könnte die Prioritätenliste so aussehen: minimiere die Größe des Reaktors, maximiere die Sicherheitsaspekte und maximiere die Ausstoßraten. Lexikographisches Goal Programming ist klar empfohlen, wenn im Modell inkompatible Zielkriterien, die in eine Hierachie gebracht werden können, zu betrachten sind, die sich nicht gut mit einer einheitlichen Skala messen lassen. Jedes Zielkriterium für sich muss allerdings noch auf seiner eigenen Skala quantifiziert werden. An einem ausführlichen Beispiel wird illustriert, wie die Zielmarken berechnet und multikriterielle Probleme in gewöhnliche LP-Probleme überführt werden. Hierzu sei das folgende lexikographische Problem

Name	Zielkriterium	Typ	A/P	Δ
goal 1 :	$5x + 2y - 20$	max	P	10
goal 2 :	$-3x + 15y - 48$	min	A	4
goal 3 :	$1.5x + 21y - 3.8$	max	P	20

mit zwei nichtnegativen Variablen $x \geq 0$ und $y \geq 0$ betrachtet, wobei in der dritten Spalte der Buchstabe A bzw. P absolute bzw. prozentuelle Abweichung erlaubt. Schließlich wird das Problem noch um eine Beschränkung $42x + 13y \leq 100$ ergänzt.

Die Grundidee besteht nun darin, die Zielkriterien hinsichtlich ihrer Prioritätenliste abzuarbeiten, d. h. das LP-Problem wird hinsichtlich des Kriteriums höchster Priorität maximiert. Mit Hilfe des so erhaltenen Wertes z_1^* wird dieses Zielkriteriums in eine Nebenbedingung mit Zielmarke $Z_1 = z_1^* - \frac{10}{100} z_1^*$ überführt, und dann wird hinsichtlich des zweiten Zielkriteriums minimiert usw., wobei bei einem zu maximierenden (minimierenden) Ziel das spezifizierte Δ nachfolgend negativ (positiv) berücksichtigt wird. Im Beispiel soll das erste Zielkriterium maximiert, das zweite minimiert und das dritte wieder maximiert werden; die Relaxierungen werden zweimal prozentual und einmal absolut vorgenommen. Die Lösung des Problems lautet

$$z_1^* = -4.615385 \quad \Rightarrow \quad 5x + 2y - 20 \geq -4.615385 - 0.1 \cdot (-4.615385) \quad . \tag{6.4.20}$$

Nun wird hinsichtlich des zweiten Zielkriteriums minimiert, wobei (6.4.20) als zusätzliche Nebenbedingung zum ursprünglichen Problem hinzugefügt wird. Es folgt:

$$z_2^* = 51.133603 \quad \Rightarrow \quad -3x + 15y - 48 \geq 51.133603 + 4 \quad . \tag{6.4.21}$$

Nunmehr wird auch das zweite Zielkriterium in die Nebenbedingung (6.4.21) mit Zielmarke umgewandelt (man beachte: absolute Abweichung +4 ist erlaubt) und hinsichtlich des dritten Zielkriteriums maximiert und ergibt schließlich $z_3^* = 141.943995$ und die Lösung $x = 0.238062$ und $y = 6.923186$. Der Vollständigkeit halber (und für die Einbeziehung evtl. weiterer Zielkriterien) könnte man auch noch das dritte Zielkriterium in eine Nebenbedingung umwandeln und erhielte so

$$1.5x + 21y - 3.8 \geq 141.943995 - 0.2 \cdot 141.943995 = 113.555196 \quad .$$

Damit eröffnet lexikographisches Goal Programming mit Zielfunktionen einen sinnvollen Weg zur Lösung multikriterieller Optimierungsprobleme. Zu bedenken ist aber, dass die Wahl der Reihenfolge die Lösung signifikant beeinflusst. In diesem Zusammenhang ist es wichtig, dass die Prozent- oder Absolutabweichung sehr behutsam gewählt und Skalierungseffekte berücksichtigt werden.

Neben der vorgestellten lexikographischen Goal Programming Variante mit Zielkriterien gibt es auch eine, die lexikographische Nebenbedingungen verwendet; die Zielkriterien bestehen hierbei in der Minimierung der Verletzung der Nebenbedingungen (dies kann z. B. Berücksichtigung finden bei der Modellierung von zu beachtenden Sicherheitsbeständen). Im idealen Fall sind alle Nebenbedingungen erfüllt. Andernfalls wird versucht, in der vorgegebenen Ordnung so viele Nebenbedingungen wie möglich zu erfüllen. Problematisch ist hierbei aber die Definition passender Gewichte. Die Prozent- oder Absolutabschläge beim lexikographischen Goal Programming mit Zielkriterien sind wesentlich einfacher und konsistenter zu interpretieren.

Für eine tiefere Betrachtung von *Goal Programming* sei auf die Bücher von Ignizio (1976,[145]) und Romero (1991,[251]) verwiesen, die eine Reihe von Variationen der ursprünglichen Ideen enthalten.

Eine zu Goal Programming sehr ähnliche Technik besteht in der Modellierung von *weichen Nebenbedingungen*; dies sind Nebenbedingungen, die verletzt werden dürfen, vorausgesetzt, dass dafür eine gewisse Strafe bezahlt wird. Die Idee hierbei ist, die rechte Seite einer Nebenbedingung durch eine passende Variable, die dann in der Zielfunktion auftreten wird, zu erweitern und so die Bedingung zu relaxieren. Seien z. B. x und y die Produktionsmengen zweier Produkte, für die die Kapazitätsbedingung $x + y \leq 100$ einzuhalten ist. Die Kapazität kann von 100 auf 110 Mengeneinheiten unter Zahlung von 20 Euro je Mengeneinheit erhöht werden. Dies führt auf die Modellierung

$$\begin{aligned} x + y - s &\leq 100 \\ s &\leq 10 \quad , \end{aligned}$$

wobei die zusätzliche Variable s in der zu minimierenden Zielfunktion $... + 20s$ auftritt.

6.4.3 Goal Programming und weiche Nebenbedingungen

In Mitra *et al.* (1995,[216]) wird der Gebrauch von Goal Programming zur Modellierung von *weichen* oder *relaxierten Nebenbedingungen* vorgestellt. In diesem Beispiel wird ein Scheduling-Problem für Kutterboote der Küstenwache gelöst. Hier wird eine vereinfachte Variante davon vorgestellt, wobei ein Teilziel dabei ist, eine Technik abzuleiten, die uns später helfen wird, die Ursache für die Unzulässigkeit eines Problems zu ergründen.

Für N^K Kutterboote soll aus N^S möglichen Fahrplänen ein Fahrplan gewählt werden, der N^R bekannten Beschränkungen genügen soll; diese sind hier nicht weiter spezifiziert, bekannt ist lediglich, ob ein Kutter, wenn er für einen bestimmten Fahrplan eingesetzt wird, eine Bestimmung verletzt. Es ist unwahrscheinlich, dass alle Beschränkungen eingehalten werden können; daher ist das Ziel, die Fahrpläne so mit Kutterbooten zu besetzen, dass eine vorgegebene Anzahl von Bestimmungen möglichst eingehalten wird, d. h. die Beschränkungen sollen als weiche Nebenbedingungen erfüllt werden.

Das Problem hängt von den folgenden Indizes ab:

$$\begin{array}{lcl} i \in \{1,2,...,N^R\} & : & \text{zu erfüllende Beschränkungen der Fahrpläne} \\ k \in \{1,2,...,N^K\} & : & \text{Kutterboote} \\ l \in \{1,2,...,N_k^S\} & : & \text{mögliche Fahrpläne für Kutter } k \\ t \in \{1,2,...,N^T\} & : & \text{Zeitperiode} \quad . \end{array}$$

Zur Verfügung stehen die Daten:

$$\begin{array}{lcl} A_{itkl} & : & = 1, \text{ wenn Kutter } k, \text{ falls für Fahrplan } l \text{ eingesetzt,} \\ & & \text{die Fahrplanbestimmung } i \text{ in Periode } t \text{ erfüllt, 0 sonst} \\ D_{it}^+ & : & \text{spezifische Strafkosten für die Überschreitung} \\ & & \text{einer Bestimmung von Fahrplan } i \text{ in Periode } t \\ D_{it}^- & : & \text{spezifische Strafkosten für die Unterschreitung} \\ & & \text{einer Bestimmung von Fahrplan } i \text{ in Periode } t \\ R_{it} & : & \text{Anzahl der einzuhaltenden Bestimmungen für Fahrplan } i \text{ in Periode } t \quad . \end{array}$$

Hierzu führen wir die kontinuierlichen (Hilfs-)Variablen

$$\begin{array}{lcl} d_{it}^+ \geq 0 & : & \text{Betrag der Überschreitung der Anzahl der Fahrplan-} \\ & & \text{bestimmungen für } i \text{ in Zeitperiode } t \\ d_{it}^- \geq 0 & : & \text{Betrag der Unterschreitung der Anzahl der Fahrplan-} \\ & & \text{bestimmungen für } i \text{ in Zeitperiode } t \quad , \end{array}$$

sowie die Binärvariablen, $\delta_{kl} \in \{0,1\}$,

$$\delta_{kl} := \begin{cases} 1 \text{ , wenn Kutter } k \text{ im Fahrplan } l \text{ eingesetzt wird} \\ 0 \text{ , sonst} \end{cases} \quad , \quad \forall k \quad , \quad \forall l$$

ein. Die bestmögliche Einhaltung aller Bestimmungen lässt sich mit Hilfe der Zielfunktion

$$\min \quad \sum_{i=1}^{N^R} \sum_{t=1}^{N^T} (D_{it}^- d_{it}^- + D_{it}^+ d_{it}^+)$$

formulieren. Hinzu treten die Gleichungen

$$\sum_{k=1}^{N^K} \sum_{l=1}^{N_k^S} A_{itkl} \delta_{kl} + d_{it}^- - d_{it}^+ = R_{it} \quad , \quad \forall i \quad , \quad \forall t \quad , \tag{6.4.22}$$

und

$$\sum_{l=1}^{N_k^S} \delta_{kl} = 1 \quad , \quad k \in \{1,2,...,N^K\} \quad ; \tag{6.4.23}$$

letztere garantiert, dass jeder Kutter einem Fahrplan zugeordnet wird. Die Gleichung (6.4.22) beschreibt die Erfordernis, dass jeder Plan die Anzahl der Bestimmungen R_{it} möglichst einhält. Mögliche Abweichungen werden durch die Hilfsvariablen d_{it}^- und d_{it}^+ aufgefangen; andernfalls versuchen sie einen Ausgleich zwischen den verschiedenen Bestimmungen zu leisten. Zu bemerken ist, dass in jedem Paar (d_{it}^-, d_{it}^+) nur eine Variable von Null verschieden sein kann; damit ist in jedem Falle eindeutig klar, dass eine Unter- oder Überschreitung der Bestimmungen vorliegt und es lässt sich ablesen, welche der Gleichungen

$$\sum_{k=1}^{N^K} \sum_{l=1}^{N_k^S} A_{itkl}\delta_{kl} = R_{it} \quad , \quad \forall i \quad , \quad \forall t \quad , \tag{6.4.24}$$

unzulässig sind. Die Differenz $d_{it}^- - d_{it}^+$ misst die Verletzung der Gleichung (6.4.24); die Erfüllung von (6.4.24) ist äquivalent zu $d_{it}^- = d_{it}^+ = 0$.

6.5 Grenzen Linearer Programmierung

Im Folgenden sollen kurz einige Grenzen der linearen Optimierung diskutiert werden.

6.5.1 Zielfunktion

Die Linearität der Zielfunktion mag für manche Kostenstrukturen möglicherweise nicht adäquat sein; einige spezielle Kostentypen können mit Hilfe linearer gemischt-ganzzahliger Modellstrukturen abgebildet werden, was in Abschnitt 11.4 demonstriert wird. Auch lassen sich manche konvexe nichtlineare Zielfunktionen mit Hilfe geeignet konstruierter konvexer Unterschätzer linear approximieren, z. B. Produkte mehrerer kontinuierlicher Variablen. Als Möglichkeit, Probleme mit mehreren Zielkriterien zu formulieren, wurde in Abschnitt 6.4.2 das Goal Programming vorgestellt.

6.5.2 Linearität in den Nebenbedingungen

In den Nebenbedingungen können nichtlineare Bedingungen auftreten, die sich nur schwer linear approximieren lassen, von den Nichtlinearitäten infolge diskreter Variablen sei hier abgesehen, da sie im Rahmen der gemischt-ganzzahligen linearen Optimierung behandelt werden. Problematisch sind nichtlineare Gleichungen, da diese insbesondere nicht mit SOS-2-Mengen modelliert werden können, weil dies erhebliche Fragen nach der Zulässigkeit aufwirft. SOS-2-Mengen bieten sich jedoch bei der Approximation konvexer nichtlinearer Ungleichungen an.

6.5.3 Harte und weiche Nebenbedingungen

Dieser Abschnitt gilt nicht nur für LP-Probleme, sondern für alle beschränkten Optimierungsprobleme. Optimierungsmodelle basieren auf der Annahme, dass alle Nebenbedingungen im Optimum *strikt* erfüllt sind und als Folge davon auch, dass alle Nebenbedingungen in einem bestimmten Sinn gleich wichtig sind. Dies muss nicht für jede Situation eine sinnvolle Annahme[3] sein, da einige *weiche Nebenbedingungen* [engl.: *soft constraints*], z. B. Fertigstellungstermine, als weniger streng anzusehen sind als die Einhaltung von Tankkapazitäten, die exakt erfüllt werden müssen, und daher auch *harte Nebenbedingungen* [engl.: *hard constraints*] genannt werden. Weiche Nebenbedingungen können mit Hilfe von *relaxierenden Variablen*, die die Verletzung von Nebenbedingungen messen und mittels spezifischer Kosten quantifizieren, etwas gelockert werden.

[3] Natürlich gibt es bestimmte Nebenbedingungen, die strikt erfüllt sein müssen. Hierzu zählen Massenbilanzen und Nebenbedingungen, die sich z. B. aus physikalischen Gesetzen ableiten.

6.5.4 Konsistente und verfügbare Daten

In mathematische Optimierungsmodelle fließen Daten ein, die die Qualität der Lösung und das Vertrauen, das man ihr entgegen bringt, erheblich beeinflussen. Die Qualität der Daten kann dabei sehr variieren. In einem grossen Produktionsplanungsmodell kann es z. B. sehr schwierig sein, die genauen Kosten für einen Produktionsprozess, der sich über 20 Stufen erstreckt, zu erhalten oder zu definieren. Bei der Erhebung von Daten sollte deshalb versucht werden, ein Maß für die Genauigkeit der Daten mitzubestimmen. Damit lässt sich dann eine *Sensitivitätsanalyse* durchführen; die es erlaubt, in der Post-Optimalen Analyse weitere Informationen abzuleiten.

6.6 Post-Optimale Analyse

Die Post-Optimale Analyse beginnt, wenn der Lösungsalgorithmus terminiert ist und sich mit einer Meldung der Art das „*Problem ist unzulässig*" [engl.: *problem is infeasible*], „*Problem ist unbeschränkt*" [engl.: *problem is unbounded*] oder „*optimale Lösung gefunden*" [engl.: *optimal solution found*] zurückmeldet. In der Regel ist es einfach, heraus zufinden, warum ein Problem unbeschränkt ist; meist wurden einige Nebenbedingungen oder Schranken vergessen. Die Ursachen für Unzulässigkeiten zu identifizieren, ist dagegen schon viel schwieriger und wird in Abschnitt 6.6.1 diskutiert. Wurde eine optimale Lösung gefunden, so stellt sich die Frage „*Wie stabil ist die Lösung*?" Hierbei ist der Begriff der Stabilität verknüpft mit der Auswirkung kleiner Schwankungen in den Eingangsdaten auf das Resultat. Im Umfeld der LP wird diese Frage mit der Hilfe der *Parametrischen Optimierung* und dem *Ranging-Verfahren* untersucht. Diese Art der Sensitivitätsanalyse wird von Wallace (2000,[287]) kritisch hinterfragt und diskutiert.

6.6.1 Untersuchung von Unzulässigkeiten

Das Problem ist unzulässig - was kann man tun? Die Diagnose von Unzulässigkeiten ist eines der schwierigsten Probleme in der Modellierung und Optimierung überhaupt. Besonders in großen Problemen kann dies sehr schwierig sein und oft ist es nötig, kleine Teilaspekte zu betrachten. Gerade von Neulingen oder Personen, die im Hinblick auf die Optimierung unerfahren sind, ist manchmal die Frage zu vernehmen „*Welche Nebenbedingung ist unzulässig*?" und sind sehr verwundert zu erfahren, dass dies eine sinnlose Frage ist. Dies wird aus der folgenden Überlegung deutlich, bei der wir ein kleines Geschäft betrachten, das zwei verschiedene Sorten von Schrauben produziert und verkauft. Betrachtet seien z. B. die die Anzahl der Sorte 1 und 2 bezeichnenden nichtnegativen Variablen β und σ in den Ungleichungen

$$\begin{array}{rcrcl} \beta & + & \sigma & \geq & 6 \\ 2\beta & + & 3\sigma & \leq & 7 \end{array} \quad , \tag{6.6.1}$$

von denen die erste Ungleichung die zu erfüllende Nachfrage, die zweite Ungleichung dagegen die verfügbare Arbeitskraft beschreibt. Keine der beiden Ungleichungen ist für sich genommen unzulässig oder falsch; aber zusammen sind sie nicht erfüllbar. Unzulässigkeiten können eine Reihe verschiedener Ursachen haben:

1. Verwechslung der Ungleichungsrichtung in einer Ungleichung; hier hilft es manchmal, das Modell mit einer Kollegin gemeinsam durchzugehen.
2. Falsche oder verwechselte Daten, z. B. weil in einer Zahl ein Komma falsch gesetzt wurde; hier hilft es, die Daten graphisch zu untersuchen.
3. Das Problem ist tatsächlich unzulässig, z. B. weil mehr Produktion gefordert wurde als es die Kapazität zulässt; die Erklärung ist zwar einleuchtend, aber in der Regel ist dieser Fall schwierig zu untersuchen und wird nachstehend diskutiert.

Ursachen für Unzulässigkeiten sind schwer aufzuspüren; die Betrachtung aller Relationen ist hilfreich und nötig, aber es sind eben mehrere Relationen, die sich bei der gegenwärtigen Datenlage widersprechen. Hilfreich können die Diskussionen in Greenberg (1993c,[119]) sein. Es ist zwar kein Wundermittel, aber dennoch hilft in vielen Situationen die *Methode der Unabhängigen Mengen von Nebenbedingungen* [engl.: *Independent Infeasible Sets; IIS*]. Ein IIS ist eine Menge von Nebenbedingungen mit der Eigenschaft, dass, wenn man auch nur eine Nebenbedingung aus der Menge entfernt, die übrigen konsistent erfüllt werden können. In diesem Sinne ist ein IIS die kleinste Menge widersprüchlicher Nebenbedingungen. Hat sie nur wenige Nebenbedingungen, so kann unter Umständen die Unzulässigkeit aufgespürt werden; andernfalls ist dies sehr schwierig.

Hat man einen Verdacht, welche Nebenbedingungen problematisch sein könnten, so kann es hilfreich sein, diese im Sinne weicher Nebenbedingungen zu relaxieren. Angenommen, in einem Produktionsplanungsmodell darf am Depotstandort d ein prozentualer Mindestanteil D^F_{dpt} der Nachfrage D_{dpt} für Produkt p, der durch den Verkauf von Depot in Zeitperiode t gedeckt werden soll, nicht unterschritten werden, also

$$s^L_{dpt} \geq D^F_{dpt} D_{dpt}, \quad \forall \left\{ dpt \,\middle|\, \exists D_{dpt} \wedge \exists D^F_{dpt} \right\} \quad .$$

Dies kann je nach Lager- und Produktionssituation – hier nicht weiter im Detail beschrieben – leicht zu einer Unzulässigkeit führen. Die nachfolgende Ungleichung

$$s^{LR}_{dpt} + s^L_{dpt} \geq D^F_{dpt} D_{dpt}, \quad \forall \left\{ dpt \,\middle|\, \exists D_{dpt} \wedge \exists D^F_{dpt} \right\} \tag{6.6.2}$$

für die Verkaufsvariable s^L_{dpt} enthält daher zusätzlich die relaxierenden Variablen $s^{LR}_{dpt} \geq 0$; diese treten zudem in der Zielfunktion

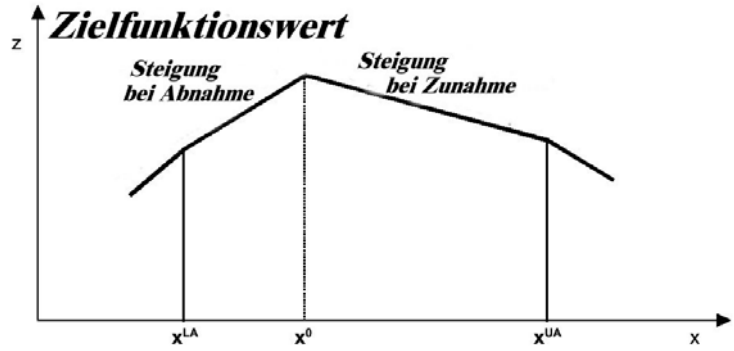
$$c^{SLB} := \sum_{d \in \mathcal{D}} \sum_{p \in \mathcal{P}} \sum_{t=1}^{N^T} 3 S^P_{dpt} s^{LR}_{dpt}$$

als zusätzlicher Kostenterm mit einem passend gewählten Strafkoeffizienten auf, der drei mal so groß wie der Verkaufspreis ist. Treten die relaxierenden Variablen s^{LR}_{dpt} in der optimalen Lösungen mit von Null verschiedenen Werten auf, so deutet dies darauf hin, dass die Ungleichung (6.6.2) ein wahrscheinlicher Kandidat für die Unzulässigkeit ist. Dies kann im nächsten Schritt überprüft werden, indem man $D^F_{dpt} = 0$ setzt. Ist der Verdacht richtig, so wird das Problem wahrscheinlich zulässig sein, wenn man einmal annimmt, dass dies die einzige Ursache für eine Unzulässigkeit ist.

6.6.2 Sensitivitätsanalyse und Ranging bei LP-Problemen

Wurde eine optimale Lösung eines Optimierungsproblems gefunden, so stellt sich die Frage „*Wie stabil ist die Lösung*?" Hierbei ist der Begriff der Stabilität verknüpft mit der

Abbildung 6.2 Sensitivitätsanalyse: Zielfunktion gegen optimalen Wert einer Variablen. Eine besondere Rolle spielen die Steigungen bei Ab- oder Zunahme des Wertes der optimalen Variablen.



Auswirkung kleiner Schwankungen in den Eingangsdaten auf das Resultat. Bei kleineren LP-Problemen ist das Ranging-Verfahren sehr nützlich; es wird meist auf die Zielfunktion und die rechte Seite der Nebenbedingungen angewendet und wird nachfolgend anhand eines Maximierungsproblems erläutert. Betrachten wir mit Hilfe von Abbildung 6.2 eine Variable x, die mit Koeffizienten P in der Zielfunktion auftritt und untersuchen, wie sich der Zielfunktionswert ändert, wenn wir x vom optimalen Wert x^* wegbewegen.

Bei Ab- oder Zunahme von x wird der Zielfunktionswert zunächst mit einer konstanten Steigung abnehmen; ab einem bestimmten Wert wird die Abstiegsgerade allerdings noch steiler. Sowohl die Steigung als auch der Punkt, wenn sich die Steigung ändert, lassen sich mit den in der LP zur Verfügung stehenden Konzepten berechnen; im allgemeinen werden die Steigungen bei Ab- oder Zunahme von x, die auch *Einheitskosten bei Ab- oder Zunahme* genannt werden, nicht gleich sein. Die Werte x^{LA} und x^{UA}, bei denen sich diese Steigung ändert, heißen *obere* und *untere Aktivität*. In einem industriellen Umfeld mag es gute Gründe dafür geben, x ein wenig von seinem optimalen Wert wegzubewegen, z. B. dass der Wert für x_* zwar noch gestern, aber aus einem Grund, der nicht im Modell abgebildet wurde, heute so nicht realisierbar ist; deshalb möchte man mehr über die Stabilität der optimalen Lösungen wissen. Die Betrachtung der *Einheitskosten bei Ab- oder Zunahme* verschafft uns sogleich Klarheit darüber, ob mit kleinen oder großen Abweichungen vom Gesamtprofit zu rechnen ist. Eine andere Quelle der Unsicherheit kann der Zielfunktionswert P, der spezifische mit x assoziierte Gewinn sein. Abbildung 6.3 illustriert, wie der optimale Wert x_* von P abhängt. Innerhalb der Grenzen P^L und P^U variiert x überhaupt nicht, ändert sich an diesen Grenzen aber unstetig und nimmt die Werte A und B an, die auch *untere* und *oberer Profit* genannt werden und gerade wieder den Werten x^{LA} und x^{UA} entsprechen.

Dieses unstetige Verhalten der optimalen Lösungen mag auf den ersten Blick überraschend erscheinen, haben wir es doch mit einem kontinuierlichen Optimierungsproblem zu tun und würden daher ein eher stetiges Verhalten der Lösung hinsichtlich der Eingangsdaten erwarten. Abbildung 6.4 enthält ein Problem mit zwei Variablen x und y und soll helfen, dieses Phänomen besser zu verstehen. Die zu maximierende Zielfunktion sei durch $Z = Px + Qy$ beschrieben; ihr Gradient ist $\nabla Z = (Q, P)^{\mathrm{T}}$ und die Kontur-

Abbildung 6.3 Sensitivitätsanalyse: Optimaler Variablenwert x gegen Zielfunktionskoeffizient P.

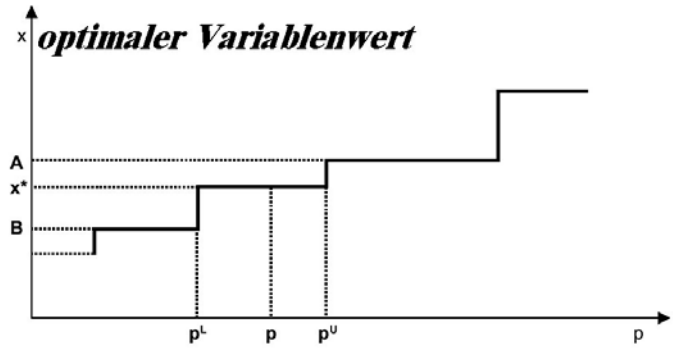
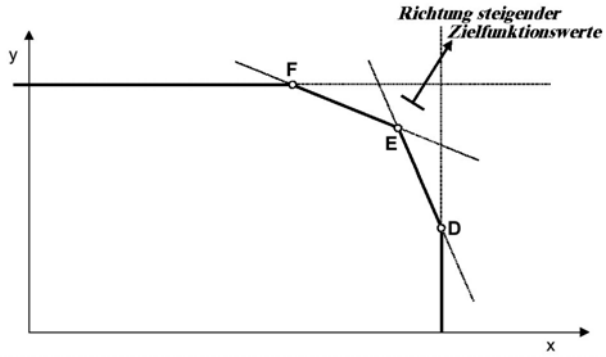


Abbildung 6.4 Sensitivitätsanalyse: Unstetiges Verhalten der optimalen Variablenwerte. Bei kleinen Änderungen der Koeffizienten der Zielfunktion können die optimalen Variablenwerte von einer Ecke zu einer anderen Ecke springen.



linien sind durch die Geraden $y = \frac{Z}{Q} - \frac{P}{Q}x$ gegeben. Für festes Q, z. B. $Q = 1$ wird bei Zunahme von P der optimale Lösungspunkt E zum Eckpunkt D bzw. bei Abnahme von P zum Eckpunkt F springen; dies geschieht, wenn die Zielfunktionskontourgerade steiler, d. h. stärker fallend, als die durch die Eckpunkte E und D führenden Gerade bzw. weniger steil als die durch die Eckpunkte E und F führenden Gerade wird. Eine kleine Änderung in P kann daher zu großen Änderung des optimalen Wertes für x führen. Ist die Zielfunktionskontourlinie für einen bestimmten Wert P_{ED} bzw. P_{EF} parallel zu einer der durch E und D bzw. E und F führenden Geraden, so könnten im Prinzip auch alle Zwischenwerte für x auf diesen Geradenstücken angenommen werden; insbesondere können sowohl die zu E als auch D bzw. sowohl E also auch F gehörenden Werte von x angenommen werden.

6.6.3 Parametrische Optimierung

Um zu sehen, wie sich die optimalen Lösungswerte bei Änderungen der Zielfunktion oder der rechten Seite der Nebenbedingungen verhalten, kann das Optimierungsproblem

$$\max_{\mathbf{x}\geq 0} \left\{ \mathbf{c}^{\mathrm{T}}\mathbf{x} \mid \mathsf{A}\mathbf{x} \leq \mathbf{b} + \theta\boldsymbol{\beta} \right\}$$

im Sinne der *Parametrischen Optimierung* untersucht werden, wobei θ zwischen 0 und 1 variiert und den Änderungsvektor der rechten Seite bezeichnet.

In ähnlicher Weise kann es nützlich sein, die Lösung des Maximierungsproblems

$$\max_{\mathbf{x}\geq 0} \left\{ (\mathbf{c} + \theta\boldsymbol{\delta})^{\mathrm{T}}\mathbf{x} \mid \mathsf{A}\mathbf{x} \leq \mathbf{b} \right\} \quad ,$$

in der angelsächsischen Literatur „*parametrics on the cost rows*" genannt, zu untersuchen, wobei θ wieder zwischen 0 und 1 liegt und der Vektor $\boldsymbol{\delta}$ die Änderung in der Zielfunktion beschreibt. Sollen die rechte Seite der Nebenbedingungen und die Zielfunktion simultan variiert werden, so wird

$$\max_{\mathbf{x}\geq 0} \left\{ (\mathbf{c} + \theta\boldsymbol{\delta})^{\mathrm{T}}\mathbf{x} \mid \mathsf{A}\mathbf{x} \leq \mathbf{b} + \theta\boldsymbol{\beta} \right\} \quad ,$$

betrachtet; man spricht von „*parametrics on the rim*".

Schwieriger, und meines Wissen nach nicht in Software implementiert, ist der Fall, bei dem sämtliche Koeffizienten der Nebenbedingungen zu variieren wären, d. h.

$$\max_{\mathbf{x}\geq 0} \left\{ \mathbf{c}^{\mathrm{T}}\mathbf{x} \mid (\mathsf{A} + \theta\alpha)\mathbf{x} \leq \mathbf{b} + \theta\boldsymbol{\beta} \right\} \quad .$$

Während das Ranging-Verfahren und auch die Parametrische Optimierung recht einsichtig und für kleine Probleme hilfreich sind, haben diese Techniken für große Probleme kaum Bedeutung, da große LP-Probleme fast immer entartet sind, d. h. es gibt viele alternative Lösungen mit gleichem Zielfunktionswert. Geringfügige Änderungen in der rechten Seite der Nebenbedingungen führen dann zwar zu unterschiedlichen Werten der Entscheidungsvariablen, aber die Zielfunktionswerte sind sehr ähnlich. Daher stimmen die unteren und oberen Aktivitäten x^{LA} und x^{UA} mit dem optimalen Wert für x überein und es wurde nicht wirklich etwas gewonnen. Bei der Anwendung der Parametrischen Optimierung auf große Probleme ist es wahrscheinlich, dass man bei Änderung von θ eine große Anzahl verschiedener Lösungen erhält.

6.6.4 Sensitivitätsanalyse in ganzzahligen Optimierungsproblemen

Abschließend folgen einige Bemerkungen zur Sensitivitätsanalyse in MIP-Problemen. Das Konzept der Sensitivitätsanalyse als Mittel der Untersuchung der Stabilität einer optimalen Lösung bei kleinen Änderungen der Eingangsdaten ist eng verknüpft mit dem Konzept der Stetigkeit. In stetigen Problemen erwartet man, dass kleine Änderungen der Eingangsdaten auch kleine Effekte in den Ausgangsdaten haben. Aber selbst in kontinuierlichen linearen Optimierungsproblemen müssen wir damit rechnen, dass der optimale Wert einer Variablen x von dem spezifischen Profitkoeffizienten P in unstetiger Weise abhängt; vergl. hierzu S. 195. In MIP-Problemen ist diese Eigenschaft noch wesentlich stärker ausgeprägt. Hierzu sei das Maximierungsproblem

$$\max_{\delta_1,\delta_2,\delta_3} Z \quad , \quad Z := 5\delta_1 + 100\delta_2 + 10\delta_3$$

unter der Nebenbedingung (hier eine Kapazitäts- oder Verfügbarkeitsbedingung)

$$20\delta_1 + 100\delta_2 + 40\delta_3 \leq C \quad , \quad \delta_i \in \{0, 1\}$$

mit der Kapazität $C = 100$ betrachtet. Die optimale Lösung ist $\delta_1 = \delta_3 = 0$, $\delta_2 = 1$ und $Z = 100$. Nimmt C auch nur minimal ab, z. B. $C = 100 - \varepsilon$, wobei ε eine nichtnegative Zahl ist, so ändert sich die Lösung zu $\delta_1 = \delta_3 = 1$, $\delta_2 = 0$ und $Z = 15$. In LP-Problemen kann die Zielfunktion derartiges Sprungverhalten bei Wechsel von Basisvariablen wie auf Seite 195 zwar auch haben, ist aber in der Regel eher stetig; bei MIP-Problemen ist die Zielfunktion aber meist unstetig.

Daher ist für MIP-Probleme eine sinnvolle Sensitivitätsanalyse nahezu unmöglich; hinzu kommen noch die von Wallace (2000,[287]) geäußerten Bedenken. Abschließend sei bemerkt, dass Williams (1995,[292]) versucht, mit Hilfe einer Verknüpfung von Sensitivitätsanalyse und Dualität einen sinnvollen Ansatz für MILP-Probleme zu erarbeiten, aber die Möglichkeiten scheinen begrenzt.

7 Gemischt-ganzzahlige lineare Optimierung in der Praxis

Dieses Kapitel enthält Fallstudien zunehmender Komplexität zur gemischt-ganzzahligen linearen Optimierung und beginnt mit einem Standortplanungsproblem. Die nächste Gruppe von Fallstudien betrachtet ein Vertragsallokationsproblem, ein Metallverschnittproblem und ein Projektplanungsproblem, aus dem ein allgemeiner Projekt-Ressourcen-Planner konzipiert wird. Schließlich wird ein Routenplanungsproblem formuliert.

7.1 Modellieren will gelernt sein - Aufbau von Erfahrung

Viele praktische Probleme, die auf gemischt-ganzzahlige Modelle führen, bedürfen einer sorgfältigen Modellierung; im Gegensatz zu LP-Problemen hängt die erforderliche Rechenzeit erheblich von einer guten Modellformulierung und von problemspezifischen Verzweigungsstrategien ab. Eine gute Modellierung ist dadurch charakterisiert, dass die LP-Relaxierung der eigentlichen MILP-Formulierung so nahe wie möglich kommt, d. h. dass der zulässige Bereich der LP-Relaxierung der konvexen Hülle des MILP-Problems so nahe wie möglich kommt. Der Prozess der guten Modellierung profitiert sehr von der Erfahrung, der Kenntnis davon, wie wir bestimmte Probleme effizient gelöst haben, was ein gutes von einem schlechten Modell unterscheidet, wie die in Kapitel 5 beschriebenen Techniken nutzbringend eingesetzt werden können.

7.2 Ein Standortplanungsproblem

In einem Bezirk gibt es sechs größere Städte, für die der lokale Feuerwehrdienst rationalisiert, d. h. die Anzahl der Feuerwehrhäuser minimiert werden soll – der Unzeitgeist des Kostencontrollings hat auch in den öffentlichen Behörden Einzug gehalten. Insbesondere sollen dabei die Anzahl und die räumliche Anordnung der Feuerwehrhäuser (jede Stadt hatte eine Station) so bestimmt werden, dass jede Stadt innerhalb von 20 Minuten von einem Feuerwehrhaus erreicht werden kann. Bekannt sind die Fahrzeiten von einer Stadt s_1 zu einer anderen Stadt s_2, nachfolgend in Minuten spezifiziert:

$s_1 \backslash s_2$	1	2	3	4	5	6
1	0	15	25	35	35	25
2	15	0	30	40	25	15
3	25	30	0	20	30	25
4	35	40	20	0	20	30
5	35	25	35	20	0	19
6	25	15	25	30	19	0

.

J. Kallrath, *Gemischt-ganzzahlige Optimierung: Modellierung in der Praxis*,
DOI 10.1007/978-3-658-00690-7_7, © Springer Fachmedien Wiesbaden 2013

Zu entscheiden ist, welche Feuerwehrhäuser weiterhin betrieben werden sollen, sodass die erwünschte Abdeckung gewährleistet und gleichzeitig die Anzahl der betriebenen Feuerwehrhäuser minimal wird. Hier kann eine ganzzahlig lineare Modellformulierung helfen, die auf den Binärvariablen $\delta_s \in \{0,1\}$

$$\delta_s := \begin{cases} 1 \text{ , wenn das Feuerwehrhaus in Stadt } s \text{ verwendet wird} \\ 0 \text{ , sonst} \end{cases}$$

aufbaut. Die Zielfunktion, hier die zu minimierende Anzahl der zu betreibenden Feuerwehrhäuser, kann in diesen Variablen sehr einfach beschrieben werden und lautet:

$$\min \quad \delta_1 + \delta_2 + \delta_3 + \delta_4 + \delta_5 + \delta_6 \quad . \tag{7.2.1}$$

In der Tabelle der Fahrzeiten interessieren besonders die Verknüpfungen von Städten, in denen die Fahrzeiten 20 Minuten nicht überschreiten. Beginnt man die Betrachtung mit Stadt 1, so folgt, dass in Stadt 1 selbst oder in Stadt 2 (oder in beiden) ein Feuerwehrhaus existieren muss, da die Stadt 1 sonst nicht in 20 Minuten von einem Feuerwehrfahrzeug erreicht werden kann; im angelsächsischen Sprachraum heißt dies *to cover*[1], und daher nennt man die zugehörige Ungleichung, die diese Überdeckung (Erreichbarkeit innerhalb von 20 Minuten) sicherstellt, auch *Cover-Ungleichung* oder kurz *Cover*

$$\textit{Cover} \text{ Stadt 1:} \qquad \delta_1 + \delta_2 \geq 1 \quad . \tag{7.2.2}$$

Das vorliegende Problem gehört daher zur Klasse der *Set-Covering*-Probleme, wobei ein „Cover" definiert ist durch die Bedingung, dass die Fahrzeit nicht größer als 20 Minuten sein darf. Führt man die für Stadt 1 durchgeführten Überlegungen für die anderen Städte durch, so ergibt sich die „Covering"- oder Überdeckungstabelle zu

$$C_{ss_f} := \begin{cases} 1 \text{ , wenn Stadt } s \text{ von Stadt } s_f \text{ aus innerhalb von 20 Minuten erreichbar ist} \\ 0 \text{ , sonst} \end{cases}$$

bzw. ausführlich

$s \backslash s_f$	1	2	3	4	5	6
1	1	1	0	0	0	0
2	1	1	0	0	0	1
3	0	0	1	1	0	0
4	0	0	1	1	1	0
5	0	0	0	1	1	1
6	0	1	0	0	1	1

.

Daraus ergeben sich die Ungleichungen

$$\begin{array}{lrrrrrrcl} \text{Cover Stadt 1:} & \delta_1 & +\delta_2 & & & & & \geq & 1 \\ \text{Cover Stadt 2:} & \delta_1 & +\delta_2 & & & & +\delta_6 & \geq & 1 \\ \text{Cover Stadt 3:} & & & \delta_3 & +\delta_4 & & & \geq & 1 \\ \text{Cover Stadt 4:} & & & \delta_3 & +\delta_4 & +\delta_5 & & \geq & 1 \\ \text{Cover Stadt 5:} & & & & \delta_4 & +\delta_5 & +\delta_6 & \geq & 1 \\ \text{Cover Stadt 6:} & & \delta_2 & & & +\delta_5 & +\delta_6 & \geq & 1 \end{array} \tag{7.2.3}$$

[1] Ein passender deutscher Begriff wäre vielleicht *überdecken*, *Überdeckung* bzw. *Überdeckungsungleichung*, aber besonders attraktiv sind diese Begriffe nicht und daher verwenden wir meist den englischen Ausdruck *Cover*.

Die Ungleichungen sind simultan z. B. für $\delta_2 = \delta_4 = 1$ und $\delta_1 = \delta_3 = \delta_5 = \delta_6 = 0$ erfüllt, d. h. es genügt, nur zwei Feuerwehrhäuser zu betreiben. Diese Lösung ist weiterhin auch optimal, d. h. es gibt keine Lösung mit nur einem Feuerwehrhaus. Weiterhin ist sie eindeutig, d. h. es gibt keine andere Lösung mit nur zwei Feuerwehrhäusern.

Lesern, die mehr über Standortplanung wissen möchten, seien Hamacher (1995,[126]), Nickel (1995,[226]) und Klose (2001,[181]) empfohlen. Eine interessante Anwendung der Standortplanung in Verbindung mit Netzwerkdesignproblemen sind Umschlagpunkt-Lokalisations-Probleme im Flugverkehr, wie sie z. B. von Mayer (2001,[205]) untersucht werden. Hierbei werden die Standorte der Umschlagpunkte [engl.: *hubs*] bestimmt und die Quell- und Zielpunkte den Umschlagpunkten zugeordnet.

7.3 Optimierung im Verkehr – Einsatzplanung für Busfahrer

Die lokalen Verkehrsbetriebe[2] möchten der verkehrsbedingt zeitlich variierenden Nachfrage nach Busfahrern kostenminimierend nachkommen und den Einsatz der Busfahrer entsprechend planen. Die Busfahrer arbeiten in 8-Stundenschichten mit einer einstündigen Essenspause und sollen den Schichten zugeordnet werden. Der Tag wird in Stunden unterteilt. Die Busfahrer sind in insgesamt $S = 12$ mögliche Schichten $s = 1, \ldots, S$ eingeteilt und fahren während der Zeit von 6^{00} bis 24^{00}, überdecken also einen 18-Stundentag. Die Schichten starten zu verschiedenen Zeiten; die erste um 6^{00}, die letzte um 17^{00}, allgemein beginnt die Schicht s um $s+5$; wir schreiben von nun an bei Uhrzeiten die beiden hochgestellten Nullen nicht mehr.

Bevor das Problem mathematisch formuliert werden soll, ist es sinnvoll, zu überprüfen, ob alle Einzelheiten korrekt verstanden wurden: die Mitarbeiter, hier Busfahrer, sollen Schichten zugeteilt werden. Aus der Zuteilung zu einer Schicht leitet sich die zeitliche Information zum Einsatz eines Busfahrers ab. Zu jeder Tageszeit ist bekannt, wieviel Busfahrer benötigt werden; die Anzahl der erforderlichen Busfahrer darf nicht unterschritten werden. Zu den Tageszeiten, zu denen mehr Busse fahren sollen, wird erwartet, dass mehr Busfahrer einer Schicht zugeordnet werden, d. h. das Problem wird von der Nachfrage getrieben. Die Kosten sollen minimiert werden, wobei die Schichten am frühen Morgen und in der Nacht besonders teuer sind. Die folgende Tabelle fasst die Kostendaten zusammen:

Schicht	1	2	3	4-8	9	10	11	12
Zeit	6-13	7-14	8-15	9-20	14-21	15-22	16-23	17-24
Kosten	100	90	80	50	60	70	80	100

.

Zu erwarten sind also Lösungen, die weniger Busfahrer den teueren Schichten zuordnen und mehr Fahrer in den Mittagszeiten einteilen. Diese Vorüberlegungen helfen sehr bei der Strukturierung und Modellierung des Problems, später auch bei der Validierung. Weichen die Ergebnisse nämlich von unseren Erwartungen stark ab, so deutet dies auf einen Mangel im Verständnis des Problems oder auf Modellierungsfehler hin.

[2] Diese Fallstudie stammt aus England aus den frühen 1990er-Jahren. Damit sind Abweichungen zum aktuellen Recht in Deutschland leicht möglich. Außerdem werden viele Aspekte, die in der Praxis relevant sind, nicht oder nur vereinfacht berücksichtigt. Lesenswert ist hier der Beitrag von Kliewer et al. (2009,[179]).

Sei t der Index der Arbeitsstunden, also $t = 1, 2, ..., T$; in unserem 18-Stundentag haben wir also $T = 18$. Das Zeitintervall von 6 Uhr bis 7 Uhr wird als erste Arbeitsstunde bezeichnet und durch $t = 1$ indiziert. Mit Hilfe dieses Zeitindex können wir festhalten, dass die Schicht s startet, wenn $t = s$; wegen $S = 12$ startet die späteste Schicht also im Zeitintervall $t = 12$, d. h. $s + 5 = t + 5 = 17$ Uhr. Die weiteren Daten sind

C_s : die Kosten der Schicht s
R_t : die benötigte Anzahl von Busfahrern im Zeitintervall t, und .
W_h : die Arbeitsstruktur innerhalb einer Schicht

Hierbei ist wieder zu beachten, dass in der Tabelle R_t der Zeitindex t dem absoluten Zeitintervall $[t+5, t+6]$ entspricht. Der Index h in der Indikationstabelle W_h läuft von 1 bis 8 und gibt an, ob in der Arbeitsstunde h einer Schicht gearbeitet wird ($W_h = 1$) oder eine Pause vorliegt ($W_h = 0$). Eine Schicht mit folgender Arbeitsstruktur [3 Stunden Arbeit, 1 Stunde Pause, 4 Stunden Arbeit] wird daher durch

h	1	2	3	4	5	6	7	8
W_h	1	1	1	0	1	1	1	1

beschrieben. Alternativ hierzu könnte auch die Indikationstabelle B_{st} mit folgender Bedeutung verwendet werden:

$$B_{st} := \begin{cases} 1 \text{ , wenn die Schicht } s \text{ die das Zeitintervall } t \text{ als Arbeitszeit enthält} \\ 0 \text{ , sonst} \end{cases} .$$

Im Zeitintervall t bzw. $[t+5, t+6]$ ist die Schicht s in ihrer $(t-s+1)$-ten Arbeitsstunde. Daher kann B_{st} aus W_h wie folgt gewonnen werden:

$$B_{st} = \begin{cases} 0 & \text{, wenn } t < s \\ W_{t-s+1} & \text{, wenn } s \le t \le s+7 \\ 0 & \text{, wenn } t > s+7 \end{cases} \quad , \quad \forall\{st\} \quad .$$

Zu bestimmen sind in unserem Modell die Anzahlen α_s der Busfahrer, die einer Schicht s zugeteilt werden. Daher führen wir die ganzzahligen Variablen $\alpha_s \in \{0, 1, \ldots, 8\}$ in unser Modell ein. Betrachten wir die Werte in Nachfragetabelle R_t

Zeit	6	7	8-10	10-12	12-14	14-16	16-17	17-19	19-20	20-23	23-24
t	1	2	3,4	5,6	7,8	9,10	11	12,13	14	15,16,17	18
R_t	3	9	10	6	7	6	8	10	8	3	2

für die Busfahrer, so wird ersichtlich, dass keiner Schicht mehr als acht Busfahrer zugeordnet werden müssen, da zu den spezifizierten Zeitintervallen mit hoher Nachfrage mindestens zwei Schichten im Einsatz sind. Insbesondere wird ersichtlich, dass man drei Busfahrer in der ersten Schicht (diese beginnt um 6 Uhr) und zwei Busfahrer in den Schichten 11 und 12 (diese enden um 23 Uhr bzw. 24) zuordnen muss. Damit hat man aber schon mindestens zwei Busfahrer, die während der Hauptzeiten (8-10) und (17-19) tätig sind. Daher brauchen nie mehr als acht Busfahrer einer Schicht zugeordnet werden.

Die Zielfunktion ergibt sich einfach zu

$$\min \quad \sum_{s=1}^{S} C_s \alpha_s$$

und unterliegt der Bedingung, dass die Anzahl der eingesetzten und zur Zeit t aktiven Busfahrer die der benötigten nicht unterschreitet, also

$$\sum_{s=1}^{S} B_{st}\alpha_s \geq R_t \quad , \quad \forall t \quad ,$$

bzw. äquivalent hierzu

$$\sum_{s=\max\{1,t-7\}}^{\min\{t,T+1-7\}} W_{t-s+1}\alpha_s \geq R_t \quad , \quad \forall t \quad .$$

Als Lösung erhält man bei diesem Problem

s	1	2	3	4	7	8	11	12
α_s	3	6	1	3	5	2	2	1

.

Wenig überraschend ist die Beobachtung, dass in der sehr frühen und den sehr späten Schichten nur wenige Busfahrer eingesetzt werden. Beachtenswert dagegen, dass man mit nur 8 Schichten auskommt.

7.4 Drei instruktive praktische Probleme

Die drei folgenden Fallstudien erschienen ursprünglich in Ashford & Daniel (1992,[21]) unter dem Titel „Some Lessons in Solving Practical Integer Programs" und mit freundlicher Genehmigung des Herausgebers des *Journal of the Operational Research Society*, Macmillan (Stockton Press, UK) in Kallrath & Wilson (1997,[167]). Aus didaktischen Gründen werden einige Aspekte hier etwas verändert dargestellt. In den Fallstudien werden verschiedene Möglichkeiten aufgezeigt, wie eine vorhandene Modellformulierung verbessert werden kann. Hierzu zählen insbesondere die Verschärfung oberer Schranken, die Verwendung spezieller Strukturen wie semi-kontinuierliche Variablen und SOS-1-Mengen und der dazu passenden Verzweigungsregeln.

7.4.1 Optimierung in der Energiewirtschaft – Vertragsallokation

Ein öffentlicher Energieträger, der für sechs Kraftwerke in sechs Regionen verantwortlich ist, muss zehn Städte mit Strom versorgen und Stromerzeugungsverträge für seine Regionen so billig wie möglich abschließen; jeder Stadt entspricht ein abzuschließender Vertrag. Jede Region, bzw. das darin befindliche Kraftwerk, hat eine begrenzte Kapazität. Bekannt sind die spezifischen Stromerzeugungskosten jeder Region für jeden möglichen Vertrag. Wird ein Vertrag mit einer Region abgeschlossen, so muss er mindestens ein bestimmtes Stromvolumen umfassen. Zur Gewährung der Ausfallsicherheit muss jede Stadt von mindestens zwei Regionen versorgt werden.

Die Regionen seien nachfolgend mit dem Index r, die Städte mit c bezeichnet. Als Entscheidungsvariablen führen wir zunächst die kontinuierlichen Variablen $x_{rc} \geq 0$ ein, die beschreiben, wieviel die Region r für die Stadt c produziert. In das Problem fließen die folgenden Daten ein:

A_r : die in r erzeugbare Strommenge,
C_{rc} : die spezifischen Kosten, wenn Region r für Stadt c produziert,
L_r : die minimale Strommenge, die von Region r abgenommen werden muss, falls von Region r Strom bezogen wird und
R_c : die für Stadt c benötigte Strommenge.

Die zu minimierende Zielfunktion repräsentiert die Kosten

$$\sum_r \sum_c C_{rc} x_{rc} \quad .$$

Zu erfüllen sind die folgenden Ungleichungen (Erfüllung der Nachfrage)

$$\sum_r x_{rc} \geq R_c \quad , \quad \forall c \quad ,$$

die Kapazitätsbeschränkung für jede Region,

$$\sum_c x_{rc} \leq A_r \quad , \quad \forall r \quad ,$$

sowie die Nichtnegativitätsbedingungen

$$x_{rc} \geq 0 \quad , \quad \forall \{rc\} \quad .$$

Strukturell handelt es sich bei diesem Problem um ein einfaches Transportproblem. Zu beachten sind aber noch die Bestimmungen für die Ausfallsicherheit und die Mindestabnahmemenge. Hierzu werden die Binärvariablen δ_{rc} benötigt; $\delta_{rc} = 1$ soll äquivalent zu $x_{rc} > 0$ sein, was bedeutet, dass $x_{rc} = 0$ zu $\delta_{rc} = 0$ führt. Um dies zu erreichen, wird zunächst die Ungleichung

$$x_{rc} \leq M \delta_{rc} \quad , \quad \forall \{rc\} \tag{7.4.1}$$

mit einer passend grossen Zahl M verwendet. Die Wahl von M sollte sich sinnvollerweise an den verfügbaren Kapazitäten, die zwischen 10 und 60 liegen, orientieren; 100 wäre ein zulässiger Wert, aber der kleinere Wert 60 wäre, wie auf Seite 161 begründet, besser. Der ursprüngliche Probleminhaber verwendete die Ungleichung

$$x_{rc} - M \delta_{rc} \geq L_r - M \quad , \quad \forall \{rc\} \quad , \tag{7.4.2}$$

um die Mindestabnahme sicher zustellen und

$$\sum_r \delta_{rc} \geq 2 \quad , \quad \forall c \quad , \tag{7.4.3}$$

um die Ausfallsicherheit zu modellieren.

Obwohl das vorliegende Problem mit 10 Städten und 6 Regionen noch recht überschaubar ist, war die B&B-Suche zunächst nach 5000 Knoten nicht beendet.

Was kann man tun? Wie in Abschnitt 5.10.1.2 argumentiert, ist es sinnvoll, die obere Schranke M so klein wie möglich zu wählen. Hier bietet sich die Wahl $M = \min\{A_r, R_c\}$ an. Daher nimmt (7.4.1) die Gestalt

$$x_{rc} \leq \min\{A_r, R_c\} \delta_{rc} \quad , \quad \forall \{rc\} \tag{7.4.4}$$

an. Wesentlich wichtiger ist noch, dass (7.4.2) besser ohne die obere Schranke M formuliert werden kann:

$$x_{rc} \geq L_r \delta_{rc} \quad , \quad \forall \{rc\} \quad . \tag{7.4.5}$$

Mit dieser Formulierung liefert das B&B-Verfahren innerhalb von 8 Knoten die optimale Lösung. In Verbindung mit (7.4.4) zeigt sich, dass (7.4.3) die Ursache für diese enorme Beschleunigung ist. Ohne diese Ungleichung hat das B&B-Verfahren wesentlich mehr Freiheiten und obwohl die optimale Lösung bereits nach 24 Knoten vorliegt, bedarf es weiterer 1417 Knoten, um die Optimalität dieser Lösung zu beweisen. Ohne (7.4.3) hätte man statt der Binärvariablen δ_{rc} auch semi-kontinuierliche Variablen σ_{rc}, $\sigma_{rc} = 0$ oder $L_r \leq \sigma_{rc} \leq L_r$ zur Abbildung der Ungleichungen (7.4.4) und (7.4.5) verwenden können. Bei Verwendung dieser Variablen terminiert das B&B-Verfahren bereits nach 132 Knoten. Zusammenfassend lässt sich bei dieser Fallstudie festhalten, dass die Wahl der oberen Schranke und die spezielle Verzweigungstechnik das B&B-Verfahren erheblich beschleunigen können.

7.4.2 Optimale Produktion von Barren in der Metallindustrie

In einem Schmelzofen fester Größe wird Metall geschmolzen und dann nach vorgegebenen Programmen bzw. Gießkombinationen in Barren verschiedener Größe gegossen. Jede Kombination erzeugt dabei eine bekannte Anzahl von Barren bestimmter Größe bzw. bestimmten Gewichts. Durch eine geeignete Anzahl von Programmen, d. h. von Gießkombinationen, soll dabei die Nachfrage nach bestimmten Barrentypen erfüllt und gleichzeitig der gesamte Abfall minimiert werden. Jede Kombination verursacht eine bestimmte Abfallmenge; unerwünschte Barren sollen ebenfalls als Abfall betrachtet werden.

Bezeichne der Index i den durch sein Gewicht charakterisierten Barrentyp und c die programmierbare Schmelzkombination, so lauten die Daten des Problems:

M_i	:	Gewicht des Barren vom Typ i
N_{ic}	:	Anzahl der Barren vom Typ i aus Schmelzkombination c
R_i	:	die nachgefragte Anzahl von Barren vom Typ i
W_c	:	Abfall (in Gewichtseinheiten) von Schmelzkombination c .

Dieses Problem ähnelt sehr den in Abschnitt 6.2.2 (Fall B) diskutierten Verschnittproblemen, was bei der Formulierung berücksichtigt werden sollte. Bezeichnet die ganzzahlige Variable α_c die Anzahl von Kesseln, die gemäß der Fließkombination c gegossen werden sollen, so lässt sich damit das Problem wie folgt formulieren: die Erfüllung der Nachfrage

$$\sum_c N_{ic} \alpha_c \geq R_i \quad , \quad \forall i \quad ,$$

wobei der Gesamtabfall

$$Z := \sum_c W_c \alpha_c + \sum_i \sum_c M_i \{N_{ic} \alpha_c - R_i\} \tag{7.4.6}$$

minimiert werden soll: der erste Term ist der mit jeder Kombination assoziierte Abfall, der zweite der mit unerwünschten Kombinationen verbundene. Durch eine geeignete Umformung ergibt sich daraus

$$Z = \sum_c \left\{ W_c + \sum_i M_i N_{ic} \right\} \alpha_c - \sum_i M_i R_i \quad .$$

Der Term $\sum_i M_i R_i$ ist konstant und kann daher in der Minimierung vernachlässigt werden. Der Koeffizient im ersten Term bezeichnet gerade das Gesamtgewicht des Kessels, hängt somit gar nicht von c ab und kann deshalb auch in der weiteren Betrachtung entfallen. Die Zielfunktion vereinfacht sich daher zu

$$\min \quad \sum_c \alpha_c \quad . \tag{7.4.7}$$

Der Versuch, eine Probleminstanz mit 18 Barrentypen und 1345 möglichen Schmelzkombinationen, also 1345 ganzzahligen Variablen, und der ursprünglichen Zielfunktion (7.4.6) zu lösen, führte nach 8000 Knoten im B&B-Verfahren zu keiner Lösung. Nutzt man jedoch die Struktur der Zielfunktion und verwendet die Form (7.4.7) und bemerkt, dass, sobald eine Lösung gefunden wurde, die nächste um mindestens 1 besser sein muss, somit eine Minimalverbesserung $\alpha = -0.999$ verwendet werden kann, so benötigt das B&B-Verfahren nur 695 Knoten, um die optimale Lösung zu bestimmen und deren Optimalität zu beweisen.

Bei diesem Beispiel, das ähnlich wie ein Verschnittproblem formuliert wurde, lässt sich zusammenfassend festhalten, dass die Ausnutzung der Struktur der Zielfunktion und die Wahl einer geeigneten Minimalverbesserung ein explosives Wachstum des B&B-Baumes verhindert.

7.4.3 Projekt-Portfolio-Optimierung und Projektplanung

Über einen Zeitraum von T Monaten soll eine Anzahl von N^P Projekten p zeitlich geplant werden; jedes Projekt kann höchstens einmal durchgeführt und darf dabei nicht unterbrochen werden, allerdings muss nicht jedes Projekt gestartet werden; in diesem Sinne handelt es sich hier um Projekt-Portfolio-Optimierung. Jedes Projekt benötigt während seiner Laufzeit eine zeitlich variierende Anzahl von Mitarbeitern und produziert nach seiner Fertigstellung monatlich bis zum Ende des Planungszeitraumes einen bestimmten Erlös R_p. So benötigt das Projekt 1 z. B. im ersten Monat 3 Mitarbeiter, 4 im zweiten und 2 im letzten Monat. Projekte können auf Monatsbasis beginnen und enden, sämtliche Daten liegen auf Monatsbasis vor; daher ist eine zeitliche Auflösung in Monaten sinnvoll. In jedem Monat steht nur eine begrenzte, vom jeweiligen Monat abhängige Anzahl von Mitarbeitern zur Verfügung und die Firma möchte die Projekte so planen, d. h. beginnen zu lassen, dass die verfügbare Personalkapazität nicht überschritten und der Erlös maximal wird. Diese Erlösstruktur liegt beispielsweise dann vor, wenn staatliche Fördermittel bis zum Ablauf eines Förderdatums für vorher fertiggestellte Energieeinsparmaßnahmen gewährleistet werden.

In der Modellformulierung werden die folgenden Indizes

$m \in \{1, ..., M_p\}$: die Menge der relativen Monate für Projekt p
$p \in \{1, ..., N^P\}$: die Menge der Projekte
$t \in \{1, ..., T\}$: die Menge der Kalender-Zeitscheiben (Monate)

verwendet. Eine Probleminstanz wird durch die folgenden Problemdaten charakterisiert:

A_t	:	das im Monat t für alle Projekte zur Verfügung stehende Personal
D_p	:	die Dauer des Projektes p in Monaten
P_{pm}	:	Anzahl der von Projekt p im m-ten Laufzeitmonat benötigten Mitarbeiter
R_p	:	der monatliche Erlös von Projekt p nach Fertigstellung

Die Entscheidungen werden durch die Binärvariablen δ_{pt},

$$\delta_{pt} := \begin{cases} 1 \text{ , wenn Projekt } p \text{ im Monat } t \text{ startet} \\ 0 \text{ , sonst} \end{cases} \qquad \forall p \quad , \quad t = 1, ..., T - D_p + 1$$

charakterisiert. Die ganzzahligen Hilfsvariablen $s_p = 0, 1, 2, \ldots$ repräsentieren den Beginn des Projektes p und sind mit den Variablen δ_{pt} gemäß

$$s_p = \sum_{t=1}^{T-D_p+1} t\delta_{pt} \quad , \quad \forall p$$

verknüpft; sie nehmen daher auch automatisch ganzzahlige Werte annehmen, d. h. sie brauchen nicht als ganzzahlige Variablen deklariert zu werden.

Wenden wir uns zunächst der Zielfunktion zu. Der Erlös ergibt sich aus dem monatlichen Erlös und der verbleibenden Zeit nach Projektende. Beginnt ein Projekt im Monat t, so endet es im Monat $t + D_p - 1$, folglich erhalten wir den Erlös R_p für $T - D_p - t + 1$ Monate. Um den Gesamterlös zu berechnen, müssen alle Zeitperioden berücksichtigt werden, in denen ein Projekt starten kann und vor Ablauf des Planungshorizontes beendet wird. Damit ein Projekt rechtzeitig endet, darf es nicht später als im Monat $T - D_p$ enden. Daher ist der zu maximierende Gesamterlös durch

$$\max \quad \sum_{p=1}^{N^P} \sum_{t=1}^{T-D_p+1} E_{pt}\delta_{pt} \quad , \quad E_{pt} := (T - D_p - t + 1)R_p \tag{7.4.8}$$

gegeben. Es brauchen nicht alle Projekte durchgeführt, aber ein Projekt kann nur einmal gestartet werden, d. h.

$$\sum_{t=1}^{T} \delta_{pt} \leq 1 \quad , \quad \forall p \quad . \tag{7.4.9}$$

Die verfügbare Personalmenge darf nicht überschritten werden, d. h.

$$\sum_{p=1}^{N^P} \sum_{u=\max\{1,t-D_p+1\}}^{t} P_{p,t-u+1}\delta_{pu} \leq A_t \quad , \quad \forall t \quad .$$

Hierbei ist zu beachten, dass im Kalendermonat t das Projekt p genau $P_{p,t-u+1}$ Mitarbeiter benötigt, wenn das Projekt im Monat u begonnen wurde; es benötigt kein Personal im Monat t, wenn es mehr als $t - D_p$ Monate zuvor begann.

Der Start eines Projekts ergibt sich, wie schon erwähnt, hier aber aus Gründen der Vollständigkeit in der Liste der Nebenbedingungen eingereiht, aus den Binärvariablen zu

$$s_p = \sum_{t=1}^{T-D_p+1} t\delta_{pt} \quad , \quad \forall p \quad . \tag{7.4.10}$$

Das ursprüngliche Problem umfasste einen Zeitraum von 50 Monaten und 17 Projekten, enthielt somit 615 Binärvariablen und war nach 742 Knoten im B&B-Verfahren gelöst. Alternativ ist auch eine Formulierung mit SOS-1-Mengen denkbar. Hierzu wird eine weitere Binärvariable $\delta_{p,T+1}$ eingeführt, die den Wert Eins annimmt, wenn das Projekt p überhaupt nicht begonnen wird. Dann bildet die Menge

$$S_p = \{\delta_{p1}, \delta_{p2}, ..., \delta_{p,T-d_j+1}, \delta_{p,T+1}\}$$

eine SOS-1-Menge, die natürlicherweise nach Monaten geordnet ist. Als Referenzbedingung bietet sich der Ausdruck

$$\sum_{p=1}^{N^P} \sum_{t=1}^{T-D_p+1} t\delta_{pt} + (T+1)\delta_{p,T+1}$$

an. Statt (7.4.9) wird noch der Term $\delta_{p,T+1}$ hinzugefügt, sodass sich die Konvexitätsbedingung

$$\sum_{t=1}^{T+1} \delta_{pt} = 1 \quad , \quad \forall p$$

ergibt. Die Variablen δ_{pt} brauchen nun natürlich nicht als Binärvariablen deklariert zu werden. Nun ist die Optimalität bereits nach 187 Knoten im B&B-Verfahren entschieden. Es ist interessant zu sehen, wie sich das B&B-Verfahren in Abhängigkeit von der Problemgröße verhält. Bei einem Planungshorizont von 60 Monaten sind bereits 785 Binärvariablen im Modell. Bei Verwendung der Binärvariablen werden 8152 Knoten, bei Verwendung der SOS-1-Menge nur 1398 Knoten benötigt, um die Optimalität der Lösung zu beweisen. Obwohl die Ordnung in der SOS-1-Menge nur schwach ausgeprägt ist, bewirkt sie doch eine erhebliche Verbesserung des B&B-Verfahrens.

7.5 Ein Projekt-Ressourcen-Planer

Das im Abschnitt 7.4.3 beschriebene Modell und Problem lässt sich zu einem allgemeinen *Projekt-Ressourcen-Planer* (PRP) erweitern; dieser kann auch zur Projekt-Portfolio-Optimierung eingesetzt werden und verwendet die folgenden Modellobjekte:

1. *Standorte s*: An diesen Orten können die Projekte durchgeführt werden.
2. *Projekte p*: Jedes potentielle Projekt im Portfolio hat die Attribute Kosten, Erlös, Ressourcenprofil als Funktion der Zeit, Dauer und mögliche Orte, wo es durchgeführt werden kann. Ein Projekt kann entweder an allen Standorten durchgeführt werden oder nur an bestimmten Standorten.
3. *Ressourcen p*: Für jeden Einsatzort werden mögliche lokale Ressourcen und Ressourcenrestriktionen berücksichtigt; einige Ressourcen werden als bewegliche oder globale Ressourcen betrachtet, z. B. kann *ein* Projektleiter sich um mehrere, an verschiedenen Orten laufende Projekte, kümmern.
4. *Zeit p*: Hier muss unterschieden werden zwischen der absoluten Zeit (Kalenderzeit), und der relativen oder internen Zeit (Projektzeit) innerhalb eines laufenden Projektes.

Jedes Portfolio ist zunächst charakterisiert durch die

1. die Anzahl der Projekte, die gleichzeitig betrachtet werden;
2. den Planungszeitraum, z. B. 5 Jahre; und
3. die zeitliche Auflösung.

Jedes Projekt wird charakterisiert durch:

1. Die *Mindestzeit*, die zur Fertigstellung des Projektes benötigt wird (wenn es ökonomisch sinnvoll ist, darf ein Projekt auch länger dauern; diese mögliche Verlängerung soll hier aber nicht weiter betrachtet werden).
2. *Zeitabhängige Projektprofile*, die beschreiben, welche und wieviele Mitarbeiter mit welchem Qualifikationsprofil zur Durchführung des Projektes benötigt werden (Beispiel: in Phase 1 werden zwei gute Chemiker, in Phase 2 dagegen drei geniale Ingenieure benötigt; in jeder Phase wird ein halber Projektleiter als erforderlich betrachtet, d. h. die Hälfte der Zeit wird er mit diesem Projekt zu tun haben; in der anderen Hälfte der Zeit kann er sich um andere Dinge kümmern.).
3. *Fixkosten*, die einmal beim Projektbeginn zu bezahlen sind, um das Projekt überhaupt zu starten.
4. *Wiederkehrende Kosten* je Zeitperiode während der Projektlaufzeit.
5. *Einmal fällig werdender Erlös* (vergl. das Beispiel in Abschnitt 7.4.3) nach Fertigstellung des Projektes.
6. *Wiederkehrende Erlöse* je Zeitperiode nach Fertigstellung des Projektes.

Jeder Standort s, an dem ein solches Projekt durchgeführt werden kann, ist wiederum charakterisiert durch

1. Die *zeitabhängige Verfügbarkeit des Personals* (Anzahl der Chemiker, Ingenieure, Techniker, etc. je Zeitperiode) oder anderer Ressourcen.
2. *Ober- und Untergrenzen des verfügbaren Personals*, das für ein bestimmtes Projekt eingesetzt werden darf (man kann nicht beliebig viele Chemiker einsetzen, da die Laborplätze in ihrer Anzahl beschränkt sind und einige Chemiker möchte man für die tägliche Arbeit reservieren; zu wenige Chemiker darf man aus Gründen der Aufsicht nicht einsetzen);
3. Eine *Ausschlussliste an Standorten*, an denen das Projekt nicht durchgeführt werden kann (weil z. B. eine bestimme Ausrüstung nicht verfügbar ist).
4. Die *zeitspezifischen Kosten*, z. B. \$/Zeitperiode, die für eine Personalressource bestimmer Qualifikation zu entrichten sind (damit lassen sich z. B. Vollzeitangestellte, aber auch freie Mitarbeiter beschreiben), wobei die Kosten sowohl bezüglich der internen Projektzeitskala als auch der Kalenderzeit variieren dürfen.
5. Eine Obergrenze für die *maximale Anzahl der Projekte*, die an diesem Standort durchgeführt werden können.

Global wird schließlich noch dasjenige Personal beschrieben, das als bewegliche Ressource angesehen wird (Projektleiter zählen dazu; Laborchemiker dagegen nicht). Ein PRP sollte typischerweise die folgenden Zielfunktionen unterstützen:

1. Maximierung des periodenbezogenen Erlöses ähnlich wie in (7.4.8),

$$\max \quad \sum_{s=1}^{N^S} \sum_{p=1}^{N^P} \sum_{t=1}^{T-D_p+1} E^C_{spt} \delta_{spt} \quad , \quad E^C_{spt} := (T - D_p - t + 1) R_p \quad , \qquad (7.5.11)$$

   d. h. ein periodenbezogener Erlös nach Wahl und Fertigstellung eines Projektes;

2. Maximierung des Erlöses gemäß (7.4.8) zuzüglich eines einmaligen Erlöses E^S_{spt} beim Start des Projektes, d. h.

$$\max \quad \sum_{s=1}^{N^S} \sum_{p=1}^{N^P} \sum_{t=1}^{T-D_p+1} \left(E^S_{spt} + E^C_{spt} \right) \delta_{spt} \quad . \qquad (7.5.12)$$

3. Maximierung des Nettogewinns (7.5.12) abzüglich variabler Kosten und Fixkosten, um das Projekt zu starten, d. h.

$$\max \quad \sum_{s=1}^{N^S} \sum_{p=1}^{N^P} \sum_{t=1}^{T-D_p+1} \left(E^S_{spt} - C_{spt} + E^C_{spt} \right) \delta_{spt} \quad ,$$

   wobei die variablen Kosten bereits mit in C_{spt} einbezogen sind, da man diese bei Projektstart deterministisch für alle nachfolgenden Perioden kennt und berechnen kann.

4. Minimierung der Kosten bei Durchführung aller Projekte

$$\min \sum_{p=1}^{N^P} \sum_{t=1}^{T-D_p+1} C_{spt} \delta_{spt} \quad ;$$

   hierbei kann es erforderlich sein, dass man zusätzliche Ressourcen zur Verfügung stellen muss, da das Problem unzulässig werden kann, und anstelle von (7.4.9) tritt

$$\sum_{s=1}^{N^S} \sum_{t=1}^{T} \delta_{spt} = 1 \quad , \quad \forall p \quad .$$

5. Maximierung der Gesamtzahl aller ausgeführten Projekte, wobei die zur Verfügung stehenden Ressourcengrenzen zu berücksichtigen sind, d. h.

$$\max \quad \sum_{s=1}^{N^S} \sum_{p=1}^{N^P} \sum_{t=1}^{T-D_p+1} \delta_{spt} \quad .$$

6. Maximierung der Ressourcenausnutzung, d. h.

$$\max \quad \sum_{s=1}^{N^S} \sum_{t=1}^{T} r^A_{st} \quad , \quad r^A_{sk} := \sum_{p=1}^{N^P} \sum_{k=1}^{t} P_{sp,t+1-k} \delta_{spk} \quad .$$

Der PRP liefert ein Projektportfolio mit zeitlicher Feinstruktur, die eine gewählte Zielfunktion strikt optimal werden lässt und zeigt standortbezogen oder global die Ressourcenauslastung bzw. welche Ressourcen unbenutzt sind. Mit einem guten PRP lassen sich zudem Stundenpläne und Arbeitspläne generieren, Urlaubspläne erstellen sowie die optimale Anzahl von Mitarbeitern zur Erfüllung einer bestimmten Menge von Aufträgen berechnen. Zur Lösung kleinerer und mittlerer Probleme können MILP-Ansätze erfolgreich sein; für größere Probleme, oder solche, mit sehr engen Ressourcenrestriktionen sind CP oder die in [46] und [225] beschriebenen Verfahren besser geeignet.

7.6 Routenplanung mit Zeitfenstern

In diesem Abschnitt soll exemplarisch ein kleines Routenplanungsproblem[3] betrachtet werden, wie es in der Praxis z. B. beim Behindertentransport auftritt. Hierbei werden von einer Koordinierungszentrale Anfragen für den Transport von Personen gesammelt. Jede Anfrage ist charakterisiert durch ein Abholzeitfenster und ein Zeitfenster für das Eintreffen am Zielort. Die Zentrale koordiniert und plant die Fahrzeuge einer Fahrzeugflotte, sodass die Anfragen möglichst ohne große Abweichungen von den Wunschzeiten erfüllt werden. Diese Aufgabenstellung kann auf viele ähnliche Situationen übertragen werden.

Bei der vorgestellten Problemformulierung greifen wir die von Desaulniers *et al.* (2001,[69]) präsentierte gemischt-ganzzahlige nichtlineare Formulierung für ein allgemeines Routenplanungsproblem mit Abholung und Zeitfenstern [engl.: *vehicle routing problem with pick-up and delivery and time windows*] auf, erweitern diese aber und erlauben den wiederholten Einsatz eines Fahrzeug, d. h. ein Fahrzeug kann mehrere Touren (Verlassen und Rückkehr zum Depot) fahren, und transformieren das Model derartig, dass schließlich eine gemischt-ganzzahlige lineare Formulierung erreicht wird.

In der Modellformulierung werden die folgenden Indizes

$$
\begin{array}{ll}
a \in \{1, ..., N^A\} & : \text{die Menge der Aufträge} \\
i \in \{1, ..., N^I\} & : \text{(meist) die Menge der Abholknoten} \\
j \in \{1, ..., N^J\} & : \text{(meist) die Menge der Ziel- bzw. Lieferknoten} \\
k \in \{1, ..., N^K\} & : \text{der } k\text{-te Abhol- bzw. Lieferknoten} \\
n \in \{1, ..., N^N\} & : \text{die Menge aller Knoten; } N^N = 2N^A + 2N^V N^T \\
t \in \{1, ..., N^T\} & : \text{die Menge der Touren für ein Fahrzeug} \\
v \in \{1, ..., N^V\} & : \text{die Menge der Fahrzeuge}
\end{array}
$$

verwendet. Die Menge der Aufträge wird in der folgenden Weise auf die Knoten eines Graphen abgebildet: a erzeugt die beiden Knoten i und $N + i$; in diesem Sinne können Auftrag a und Knoten i auch als synonym angesehen werden, was bei der Indizierung im folgenden auch ausgenutzt wird. Verschiedenen Knoten i kann aber dieselbe Abholposition zugeordnet werden, d. h. die Knoten korrespondieren nicht unmittelbar mit örtlichen

[3] Die vorgestellte MILP-Formulierung lässt sich wie vorgestellt sicher nicht auf größere industrielle Probleme der Transportlogistik übertragen. Zum einen nicht wegen der fehlenden Abbildungstreue, da viele praxisrelevante Aspekte nicht berücksichtigt wurden, zum anderen nicht, da ein einfacher MILP-Ansatz numerisch schnell an seine Grenzen stößt. Hierzu seien auch die Artikel von Bunte & Kliewer (2009,[47]) und Kliewer *et al.* (2012,[180]) empfohlen.

Lokalisationen. Die Abholknoten sind in der Knotenmenge $\mathcal{P} := \{1, \ldots, N\}$ zusammengefasst, die Zielknoten in der Menge $\mathcal{D} := \{N+1, \ldots, 2N\}$. Die Menge $\mathcal{N} := \mathcal{P} \cup \mathcal{D}$ vereinigt beiden Knotenmengen. Eine Probleminstanz wird durch die folgenden Daten

A_n	:	früheste Anfahrtszeit für Knoten n (Zeitfenster)
B_n	:	späteste Anfahrtszeit für Knoten n (Zeitfenster)
C_{vt}	:	die Personenkapazität von Fahrzeug v in Tour t
C^D_{ijvt}	:	die Fahrkosten von Fahrzeug vt vom Knoten i zum Knoten j
$C^{PA}_{ivt}, C^{PB}_{ivt}$	:	Strafkosten für Nichteinhaltung der Zeitfensterbedingungen
T_{ijvt}	:	die Fahrzeit von Fahrzeug vt vom Knoten i zum Knoten j
D_i	:	Personenbedarf, der bei Anfahrt von Knoten i bzw. Auftrag a anfällt
L_i, L_{N+i}	:	Lastwechsel bei Anfahrt von Knoten i; $L_i := D_i$, $L_{N+i} := -D_i$
S_n	:	vorgegebene Aufenthaltszeit am Knoten n, z. B. Einladezeit
T^P_a	:	die gewünschte Abholzeit für Auftrag a
T^D_a	:	die gewünschte Ankunftszeit für Auftrag a

charakterisiert. Weiterhin ist es nützlich, die folgenden Mengen einzuführen:

$\mathcal{A}_{vt}$	:	die Menge aller von Fahrzeug vt fahrbaren Wege von i nach j
$\mathcal{D}$	:	die Menge aller Zielknoten
$\mathcal{D}_{vt}$		die Menge aller Zielknoten für Fahrzeug vt
$\mathcal{N}$	:	die Menge aller Abhol- und Zielknoten, $\mathcal{N} := \mathcal{P} \cup \mathcal{D}$
$\mathcal{N}_{vt}$	:	die Menge aller von vt anfahrbaren Knoten; siehe $\mathcal{N}$
$\mathcal{P}$	:	die Menge aller Abholknoten
$\mathcal{P}_{vt}$		die Menge aller Abholknoten für Tour t des Fahrzeugs v
$\mathcal{V}_{vt}$	:	die um Abfahrt- & Zieldepot erweiterte Knotenmenge; $\mathcal{V}_{vt} := \mathcal{N}_{vt} \cup \{s_v, d_v\}$

Damit lässt sich nun der Graph $\mathsf{G}_{vt} := (\mathcal{V}_{vt}, \mathcal{A}_{vt})$ mit Knoten $\mathcal{V}_{vt}$ und Kanten $\mathcal{A}_{vt}$ definieren. Die Entscheidungen werden durch die binären Flussvariablen δ_{ijvt},

$$\delta_{ijvt} := \begin{cases} 1 \text{ , wenn Fahrzeug } vt \text{ von Knoten } i \text{ nach Knoten } j \text{ fährt} \\ 0 \text{ , sonst} \end{cases} \qquad \forall \{ijvt | (ij) \in \mathcal{A}_{vt}\},$$

den binären Hilfsvariablen

$$\alpha_{ivt} := \begin{cases} 1 \text{ , wenn Fahrzeug } vt \text{ Auftrag } i \text{ bedient} \\ 0 \text{ , sonst} \end{cases} \qquad \forall \{ivt | (i) \in \mathcal{A}_{vt}\}$$

und den nichtnegativen kontinuierlichen Variablen

p^V_{nvt}	:	Personenzahl im Fahrzeug vt bei Abfahrt im Knoten n
t^A_{nvt}	:	Ankunftszeit von Fahrzeug v auf seiner Tour t im Knoten n
$r^A_{ivt}, r^B_{jvt},$	:	Variablen zur Relaxierung der Zeitfensterbedingungen

charakterisiert. Aus praktischen Gründen (re-scheduling, on-line scheduling) kann es sinnvoll sein, den Fahrzeugen bereits bestimmten Teilmengen $\mathcal{A}_{vt}$ des Netzwerkes zu zuteilen, wodurch sich viele Binärvariablen einsparen lassen.

Die zu minimierende Zielfunktionen

$$\sum_{v=1}^{N^V} \sum_{t=1}^{N^T} \sum_{(ij) \in \mathcal{A}vt} C^D_{ijvt} \delta_{ijvt} + \sum_{v=1}^{N^V} \sum_{t=1}^{N^T} \sum_{i \in \mathcal{P}_{vt}} C^{PA}_{ivt} r^A_{ivt} + \sum_{v=1}^{N^V} \sum_{t=1}^{N^T} \sum_{j \in \mathcal{D}_{vt}} C^{PB}_{jvt} r^B_{ivt} \quad .$$

enthält als Erstes die Kosten entlang der Kanten des Graphen dar, z. B. weglängenabhängige Fahrtkosten. Der zweite und dritte Term sind mit der Relaxierung der Zeitfensterbedingung (7.6.14) verbundene Strafterme.

Zu erfüllen sind folgenden Nebenbedingungen. Jeder Abholknoten kann genau einmal verlassen werden

$$\sum_{v=1}^{N^V}\sum_{t=1}^{N^T}\sum_{j\in\mathcal{N}_{vt}\cup d_v}\delta_{ijvt}=1 \quad , \quad \forall i\in\mathcal{P} \quad ,$$

d. h. zu jedem Liefer- bzw. Zielknoten korrespondiert ein entsprechenden Abholknoten,

$$\sum_{j\in\mathcal{N}vt}\delta_{ijvt}-\sum_{j\in\mathcal{N}vt}\delta_{j,N+i,vt}=0 \quad , \quad \forall v\in\mathcal{V} \quad , \quad \forall t\in\mathcal{T} \quad , \quad \forall i\in\mathcal{P}_{vt} \quad .$$

Jedes Fahrzeug muss sein Depot verlassen (notfalls direkt zum Zieldepot fahren)

$$\sum_{j\in\mathcal{N}_{vt}\cup d_v}\delta_{s_vjvt}=1 \quad , \quad \forall v\in\mathcal{V} \quad , \quad \forall t\in\mathcal{T} \quad ,$$

zu jedem Ziel- oder Zwischenzielknoten muss es einen Abfahrt- oder Zwischenabfahrtsknoten geben,

$$\sum_{i\in\mathcal{N}_{vt}\cup s_v}\delta_{ijvt}-\sum_{i\in\mathcal{N}_{vt}\cup d_v}\delta_{jivt}=0 \quad , \quad \forall v\in\mathcal{V} \quad , \quad \forall t\in\mathcal{T} \quad , \quad \forall j\in\mathcal{N}_{vt} \quad ,$$

jedes Fahrzeug muss sein Ziel-Depot wieder anfahren,

$$\sum_{i\in\mathcal{N}_{vt}\cup s_v}\delta_{id_kvt}=1 \quad , \quad \forall v\in\mathcal{F} \quad , \quad \forall t\in\mathcal{T} \quad ,$$

Verträglichkeit zwischen Fahrt-Route und Zeitplan, d. h. Zeitbilanz,

$$\delta_{ijvt}\left[t^A_{ivt}+S_i+T_{ijvt}-t^A_{jvt}\right]=0 \quad , \quad \forall\{ijvt|(ij)\in\mathcal{A}_{vt}\} \quad , \tag{7.6.13}$$

die, wie in Abschnitt 6.4.3 beschrieben, relaxierten Zeitfensterbedingungen,

$$A_i-r^A_{ivt}\le t^A_{ivt}\le B_i+r^B_{ivt} \quad , \quad \forall\{ivt|i\in\mathcal{V}_{vt}\} \quad , \tag{7.6.14}$$

jeder Zielknoten wird später als der entsprechende Abholknoten angefahren,

$$t^A_{ivt}+T_{i,N+i,vt}\le t_{N+i,vt} \quad , \quad \forall\{ivt|i\in\mathcal{P}_{vt}\} \quad , \tag{7.6.15}$$

Bilanzierung der Last,

$$\delta_{ijvt}\left[p^V_{ivt}+L_j-p^V_{jvt}\right]=0 \quad , \quad \forall\{ijvt|(ij)\in\mathcal{A}_{vt}\} \quad ,$$

bzw. in der linearen Form

$$p^V_{ivt}+L_j\le p^V_{jvt}+M_{ijvt}-M_{ijvt}\delta_{ijvt} \quad , \quad \forall\{ijvt|(ij)\in\mathcal{A}_{vt}\} \quad ,$$

Kapazitätsbeschränkung der Fahrzeuge in den Abfahrtsknoten,

$$L_i\le p^V_{ivt}\le C_{vt} \quad , \quad \forall\{ivt|i\in\mathcal{P}_{vt}\} \quad ,$$

Kapazitätsbeschränkung der Fahrzeuge in den Zielknoten,

$$0 \leq p^V_{N+i,vt} \leq C_{vt} - L_i \quad , \quad \forall\{ivt|N+i \in \mathcal{D}_{vt}\} \quad ,$$

Definition der Anfangslast,

$$p^V_{s_v vt} = 0 \quad , \quad \forall vt \quad ,$$

sowie die Nichtnegativitäts- und Ganzzahligkeitsbedingung

$$\begin{aligned} 0 &\leq \delta_{ijvt} \leq 1 \quad , \quad \delta_{ijvt} \in \{0,1\} \quad , \quad \forall\{ijvt|(ij) \in \mathcal{A}_{vt}\} \\ 0 &\leq \alpha_{ivt} \leq 1 \quad , \quad \alpha_{ivt} \in \{0,1\} \quad , \quad \forall\{ivt|(ij) \in \mathcal{A}_{vt}\} \quad . \end{aligned}$$

Bei der von Desaulniers *et al.* (2001,[69]) vorgestellten Formulierung kommt jedes Fahrzeug nur einmal zum Einsatz. Um mehrere Touren (Reise vom Ausgangs- s_v zum Zieldepot d_v, welches in unserer Beschreibung räumlich identisch mit dem Ausgangsdepot sein soll) zu ermöglichen, sei angenommen, dass in dem betrachteten Zeitintervall von einem Fahrzeug N^T Touren möglich sind; räumlich können Ausgangs- und Zieldepots identisch sein, dies muss aber nicht so sein. Der Modellierungsansatz beruht auf der Überlegung, dass die Abfahrtszeit $a_{s_v,vt}$ aus dem Ausgangsdepot s_v für eine nachfolgende Tour t nicht früher sein darf als die Rückkehrzeit $t^A_{d_v,v,t-1}$ der vorausgehenden Tour $t-1$ desselben Fahrzeugs und gegebenenfalls noch eine Pausenzeit T^B_{vt} zu berücksichtigen ist. Erreicht Fahrzeug v während seiner Tour t einen beliebigen Knotenpunkt n, so sind aufeinanderfolgende Touren eines Fahrzeuges durch

$$t^A_{nvt} \geq t^A_{d_v,v,t-1} + T_{s_v nvt} + T^B_{vt-1} \quad , \quad \forall v \quad , \quad t = 2, \ldots, N^T$$

miteinander verknüpft. Für die Depotknoten s_v und d_v werden keine Zeitfensterbedingungen berücksichtigt.

Bei der Beziehung (7.6.15) ist zu beachten, dass eine Verbindung vom Knoten i zum Knoten $N+i$ nicht unbedingt existieren muss, also auch nicht die Fahrtzeitinformation T_{ijvt}. Existiert die Verbindung, so kann der Zielknoten auch auf Umwegen erreicht werden; die Zeit T_{ijvt} ist dann wegen der Dreiecksungleichung meist eine untere Schranke – dies sollte aber im Einzelfall überprüft werden.

Die nichtlineare Bedingung (7.6.13) impliziert „wenn $\delta_{ijvt} = 1$, dann $t^A_{ivt} + S_i + T_{ijvt} - t^A_{jvt} = 0$" und lässt sich in die folgende lineare Beziehung transformieren

$$t^A_{ivt} + S_i + T_{ijvt} \leq t^A_{jvt} + (1 - \delta_{ijvt})\, M_{ijvt} \quad ,$$

wobei M_{ijvt} eine hinreichend groß gewählte Zahl ist; eine gute Wahl für M_{ijvt} ist

$$M_{ijvt} := \max\{B_i + S_i + T_{ijvt} - A_j, 0\} \quad .$$

Für $\delta_{ijvt} = 1$ führt $t^A_{ivt} + S_i + T_{ijvt} \leq t^A_{jf}$ wegen (7.6.15) auf $t^A_{ivt} + S_i + T_{ijvt} = t^A_{jvt}$. Andernfalls gilt wegen der Dreiecksungleichung in der Ebene $t^A_{ivt} + S_i + T_{ijvt} < t^A_{jvt}$, falls $\delta_{ijvt} = 0$ und das Fahrzeug auf einem anderen Weg von Knoten i zum Knoten j fährt. Es kann aber auch sein, dass das Fahrzeug zunächst den Knoten j und dann erst den Knoten i besucht; in diesem Fall gilt $t^A_{ivt} > t^A_{jvt}$.

Bei der Zeitfensterbedingung (7.6.14) ist zu beachten, dass das Fahrzeug v in seiner Tour t tatsächlich den Auftrag a bedient bzw. Knoten i anfährt; wenn dies nicht so ist, so darf (7.6.14) nicht verwendet werden (es ergäben sich sonst unsinnige Beiträge in der Zielfunktion) und wird in folgender Weise durch Relaxierung ausgeschaltet:

$$A_i\alpha_{ivt} - r^A_{ivt} \le t^A_{ivt} \le B_i + r^B_{ivt} + M^R - M^R\alpha_{ivt} \quad , \quad \forall\{ivt|i \in \mathcal{V}_{vt}\} \quad ,$$

mit einer hinreichend groß gewählten Zahl M^R; ein sicherer Wert bei Tagesplanung ist $M^R = 24$. Da die Variablen r^A_{ivt} und r^B_{jvt} in der Zielfunktion bestraft werden, ist es zwar nicht notwendig, aber zur Verbesserung der Modellgüte sinnvoll, die Variablen zu beschränken und mit der Binärvariablen α_{ivt} gemäß

$$r^A_{ivt} \le R^A\alpha_{ivt} \quad , \quad \forall i \in \mathcal{P}_{vt} \quad , \quad \forall v \in \mathcal{V} \quad , \quad \forall t \in \mathcal{T} \quad ,$$

und

$$r^B_{jvt} \le R^B\alpha_{jvt} \quad , \quad \forall j \in \mathcal{D}_{vt} \quad , \quad \forall v \in \mathcal{V} \quad , \quad \forall t \in \mathcal{T} \quad ,$$

zu verknüpfen, wobei R^A und R^B die maximal erlaubten Zeiten für zu frühe oder zu späte Ankunft am Knoten bedeuten; hier kann z. B. 0.5 Stunden ein sinnvoller Wert sein. Die Variable α_{ivt} ist nützlich, um das Branch&Bound-Verfahren mit Hilfe von Prioritäten und Richtungen zu unterstützen, wie vorstehend beschrieben, die Zeitfensterbedingungen zu relaxieren, aber auch, um fest vorgegebene Zuordnungen von Fahrzeugen zu Aufträgen, z. B. bei re-scheduling, in das Modell zu integrieren. Es gilt

$$\alpha_{ivt} = \sum_{n \in \mathcal{N}_{vt}} \delta_{nivt} \quad , \quad \forall i \in \mathcal{P}_{vt} \quad , \quad \forall v \in \mathcal{V} \quad , \quad \forall t \in \mathcal{T} \quad ,$$

und jedem Auftrag kann natürlich nur eine Fahrzeug-Tour-Kombination vt zugeordnet werden, d. h.

$$\sum_{v=1}^{N^V} \sum_{t=1}^{N^T} \alpha_{ivt} = 1 \quad , \quad \forall i \in \mathcal{P}_{vt} \quad .$$

Die vorgestellte MILP-Formulierung ist in der Lage, kleinere Probleminstanzen zu behandeln. Aber selbst Probleme mit nur 5 Aufträgen, von denen 4 die Abholzeit 10^{50} Uhr und einer die Abholzeit 10^{43} Uhr haben, benötigen aber schon etwa 330000 Knoten bevor die optimale Lösung bestimmt ist, dies allerdings in wenigen Minuten auf einem gängigen PC. Es lassen sich aber bei günstigeren Daten auch Instanzen mit bis zu 20 Aufträgen optimal lösen. Da die Routenplanungsprobleme zu den $\mathcal{NP}$-vollständigen Problemen gehören [254], ist nicht verwunderlich, dass zu ihrer Lösung exakte Verfahren und Heuristiken verknüpft werden. Die zur Lösung asymmetrischer Rundreiseprobleme mit Zeitfensterbedingungen von Ascheuer *et al.* (2001,[19]) entwickelten und beschriebenen Ansätze sind hier auch gut verwendbar.

8 Polylithische Modellierungs- und Lösungsansätze in der Praxis

Dieses Kapitel handelt von polylithischen Modellierungs- und Lösungsansätzen. Solche Ansätze erlauben es, die Menge der lösbaren Praxisprobleme sowohl in ihrer Qualität (Struktur) und Größe (Anzahl der Variablen und Constraints) erheblich zu erweitern. Anhand von Problemen aus der Papierindustrie, die mit Hilfe polylithischer Modellierungs- und Lösungsansätze gelöst wurden, werden diese Ansätze illustriert. Im Einzelnen werden die Rollenminimierung mit Hilfe eines Spaltenerzeugungsverfahrens, gleichzeitige Minimierung von Verschnitt und Anzahl verwendeter Muster, sowie die Formatproduktion behandelt.

8.1 Polylithische Modellierungs- und Lösungsansätze

Viele praktische gemischt-ganzzahlige Optimierungsprobleme lassen sich nicht leicht lösen; wie in Abschnitt 1.8 erläutert, wollen wir diese als *schwierig* bezeichnen.

Anstatt diese schwierigen Probleme direkt als monolithische Probleme anzugehen, kann man sie äquivalent auch als Sequenz von Modellen lösen, was auf polylithische Modellierungs- und Lösungsansätze führt. Dazu gehören zum Beispiel:

1. Spaltenerzeugung [engl.: *column generation*] wie in Abschnitt 8.2.3 erläutert und angewendet,
2. *Branch-and-Price*, eine Erweiterung von Spaltenerzeugung auf gemischt-ganzzahlige Probleme [*cf.* Abschnitt 4.3.4 oder Barnhart *et al.* (1998,[23])],
3. Benders' Dekomposition – siehe Abschnitt 8.1.3.2.1,
4. die Auswertung von Hilfsproblemen zur Ableitung sicherer Schranken für das Originalproblem, welches wiederum dazu führt, dass sich das Originalproblem leichter lösen lässt,
5. Hybridverfahren, in denen konstruktive Heuristiken gemeinsam mit exakten Optimierungsalgorithmen verwendet werden (siehe Abschnitt 8.3), oder
6. lexikographisches Goal-Programming zur Lösung multikriterieller Optimierungsprobleme (siehe Abschnitt 6.4).

8.1.1 Idee und Grundlagen Polylithischer Lösungsansätze

Basierend auf dem griechischen Begriff *monolithos* (monolithisch; ein Stein, der nur aus einem Block besteht) führte Kallrath (2009,[160]; 2011,[162]) den entsprechenden Begriff *polylithisch* für Modellierungs- und Lösungsansätze ein, in denen gemischt-ganzzahlige oder nichtkonvexe nichtlineare Optimierungsprobleme mit Hilfe maßgeschneiderter Methoden gelöst werden, bei denen mehrere verknüpfte Modelle oder Algorithmen zum Einsatz kommen.

J. Kallrath, *Gemischt-ganzzahlige Optimierung: Modellierung in der Praxis*,
DOI 10.1007/978-3-658-00690-7_8, © Springer Fachmedien Wiesbaden 2013

8.1.1.1 Monolithische Modelle und Lösungsansätze

Ein *monolithisches Modell* besteht einfach nur aus *einem* Modell mit Daten, Variablen und Constraints und einem Aufruf eines Lösungsalgorithmus zur Lösung von LP-, MILP-, NLP-, oder MINLP-Problemen – für die Probleme, die wir bisher in diesem Buch vorgestellt haben, ist diese Verwendung eines *general purpose solvers* der Standard. Dadurch bleibt die Struktur des Modells und seiner Lösung relativ einfach und übersichtlich. Bei Verwendung einer algebraischen Modellierungssprache werden die möglichen Schranken auf die Variablen nur einmal gesetzt. Die Modellausgabe und Ergebnisse des Solvers fließen direkt in einen ansprechend aussehenden Ausgabebericht.

8.1.1.2 Polylithische Modelle und Lösungsansätze (PMSAs)

Im Gegensatz zu den monolithischen Verfahren bestehen *polylithische Modelle* aus einer Menge von Modellen, die hinsichtlich ihrer Ein- und Ausgaben miteinander verknüpft sind, d. h. Modell $m+1$ kann Ergebnisse der ersten m Modelle verwenden. Dies kann ausgenutzt werden, um bestimmte Variablen zu initialisieren oder Variablen zu beschränken. Beispiele polylithischer Lösungsansätze sind Dekompositionsverfahren, wie z. B. das Spaltenerzeugungsverfahren von Gilmore & Gomory (1961,[106]) oder Branch&Price [*cf.* Barnhart *et al.* (1998,[23])], oder Hybrid-Verfahren [*cf.* Pochet &Wolsey (2006,[233])], in denen konstruktive Heuristiken und/oder lokale Such- und Verbesserungsverfahren mit exakten MIP-Algorithmen verknüpft sind, um zulässige Punkte oder gute untere und obere Schranken zu berechnen. In natürlicher Weise erhält man damit maßgeschneiderte Algorithmen [engl.: *tailor-made algorithms*]. Dies verlangt jedoch bei der Programmierung sehr viel Sorgfalt und Aufmerksamkeit hinsichtlich der folgenden Punkte:

1. Schranken und Startwerte der Variablen müssen sehr sorgfältig verfolgt und aktualisiert werden.
2. Die Ergebnisse von diskreten Variablen sind in der Regel nicht ganzzahlig, sondern weichen minimal davon ab. Da diese Ergebniswerte jedoch weiter verwendet werden, ist es oft sinnvoll und nötig, diese zu runden. Allerdings kann es hierdurch zu Unzulässigkeiten kommen. Zur Vorsicht wird geraten.
3. Variablen und Nebenbedingungen in verschiedenen Modellen sollten als lokale Objekte deklariert werden; andernfalls sollte man durch die Namenswahl Missverständnisse vermeiden.
4. Der Wartungsaufwand polylithischer Verfahren ist wesentlich höher als bei monolithischen; die Erweiterung zusätzlicher Funktionalitäten in Modell und Lösungsansatz kann aufwendig sein.

PMSAs haben ein breites Anwendungsfeld; sie erweitern die Menge der Praxisprobleme, die wir noch in vernünftiger Zeit lösen können, erheblich. Wir finden Sie in problemspezifischen Preprocessing, in allgemeinen mathematischen Algorithmen und in strukturierten primalen[1] Heuristiken und Hybrid-Verfahren. Maßgeschneiderte polylithische

[1] Der Begriff *primal* bezieht sich hier auf das primale Problem, d. h. auf das Optimierungsproblem in seiner Ausgangsform. Primale Heuristiken und Verfahren liefern lediglich primal zulässige Punkte. Erst durch die Hinzunahme dualer Informationen kann ein Optimalitätsnachweis erbracht werden.

Lösungsansätze mit Tausenden oder Millionen von Lösungsaufrufen stellen eine besondere Herausforderung für algebraische Modellierungssprachen dar. Warm- oder Hotstarts, mit deren Hilfe vermieden werden kann, dass das Modell wieder und wieder übersetzt und kompiliert werden muss, werden zu einer notwendigen Voraussetzung.

PMSAs sind empfohlen, wenn das Problem nicht direkt gelöst werden kann – sie stellen ein mächtiges, aber letztes Mittel dar. Sie können auch als eine Art Brückentechnologie angesehen werden, die ersetzt werden kann, wenn im Bereich der Lösungsalgorithmen signifikante Verbesserungen erzielt werden.

8.1.2 Problemspezifisches Preprocessing

In Abschnitt 5.10 wurden bereits verschiedene *Preprocessing*-Methoden vorgestellt und darauf hingewiesen, dass kommerzielle Softwareanbieter diese Methoden in ihren Solvern verwenden, häufig aber nicht genau verraten, was und wie implementiert wurde. Neben diesem Transparenzdefizit ist ein weiteres Argument für die Entwicklung eigener Preprozessingtechniken und der Ausnutzung problemspezifischer Strukturen und Eigenschaften, dass der Probleminhaber in der Regel mehr von seinem Problem weiß als die Software automatisch identifizieren kann.

8.1.2.1 Dynamische Reduzierung von Big-M-Koeffizienten

In Kapitel 5 wurden an verschiedenen Stellen Big-M-Koeffizienten (nachfolgend in Kurzform: *Big-Ms*) verwendet. Die Bedeutung von minimalen Big-Ms bei der Aktivierung oder Deaktivierung von Ungleichungen wie

$$x \leq X + M(1 - \delta)$$

mit einer kontinuierlichen Variablen x und einer Binärvariablen δ, die iterative Reduzierung von M in einer Folge von Modellen und ein illustratives Produktionsplanungsbeispiel, das komplett in der Modellierungssprache `GAMS` umgesetzt wurde, wird in Kallrath (2009,[160]) diskutiert. Die Anzahl der Knoten in einen Branch&Bound Algorithmus kann dadurch signifikant verringert werden. In Allgemeinen ist diese Methode effektiv, wenn die im kommerziellen Solver eingebauten Presolving-Verfahren nicht zu einer Modellverschärfung führen.

8.1.2.2 Verschärfte Schranken auf ganzzahlige Variablen

Mit Hilfsproblemen und PMSAs können verschärfte obere Schranken auf ganzzahlige Variablen berechnet werden. Ein Beispiel dafür ist die Intervallverkleinerung der Anzahl von Batches in Lin *et al.* (2005,[195], Tabelle 4), in dem ein Reaktorportfolioproblem gelöst wird. Die Zielfunktion des Hilfsproblems bestand darin, die Anzahl der Batches zu maximieren. Im Kontext ganzzahliger Variablen ist dies sehr empfohlen, da es die untere Schranke des ursprünglichen Minimierungsproblem erheblich verbessern kann. Dies wird anschaulich klar, wenn man sich den Unterschied in der oberen Schranke einer ganzzahligen Variablen $\alpha \leq 3$ anstatt $\alpha \leq 3.4$ klarmacht.

8.1.2.3 Datenzulässigkeitstest

In komplexen Anwendungen ist es nicht leicht, die Konsistenz der Eingangsdaten sicherzustellen oder zu prüfen, da sich Unzulässigkeiten meist erst dann zeigen, wenn mehrere Constraints gleichzeitig erfüllt werden müssen. Automatische Zulässigkeitsprüfungen sind daher in Produktionsplanung und Scheduling unerlässlich; dies gilt sowohl für *Supply Network Planning* als auch in der Energiewirtschaft. In der Prozessindustrie reichen solche Tests zum Beispiel in Schedulingmodellen von einfachen Tests zur Identifizierung inkonsistenter Batchgrößen über die Lösung von Hilfsmodellen zur Überprüfung der Konsistenz von State-Task-Netzwerken und Topologie, Kompatibilität der Produktionsanlagen, Lager und Nachfrage bis hin zu erweiterten Prozeduren wie Einzel-Auftrags-Analysen, in denen das Schedulingproblem für jeden einzelnen Auftrag exakt gelöst wird, wodurch unvermeidbare Unterproduktion oder Verspätungen identifiziert werden.

Insbesondere sollten unvermeidbare Unterproduktion oder Verspätungen rigoros mit Hilfe der in Abschnitt 6.4.2 dargestellten Mehrzieltechniken behandelt werden anstatt nur die Unterproduktion oder Verspätungen zu bestrafen. Wie von Kallrath & Maindl (2006,[163], pp. 344) diskutiert, führen Strafterme mit Koeffizienten, die nicht ökonomisch interpretiert werden können, zu numerischen Problemen und erschweren die Interpretation der Ganzzahligkeitslücke.

8.1.3 Mathematische Algorithmen

Hier betrachten wir allgemeine mathematische Algorithmen wie beispielsweise Dekompositionsverfahren, Branch&Bound, Branch&Cut, dynamische Programming (DP) oder Goalprogramming (GP) für multikriterielle Optimierungsprobleme.

8.1.3.1 Branch&Bound und Branch&Cut

Die meisten kommerziellen Solver verwenden Branch&Bound (B&B) um MIP oder nichtkonvexe NLP or MINLP Probleme zu lösen; dabei wird gewöhnlich auf Variablen verzweigt. Für semi-kontinuierliche Variablen (SCVs) oder special-ordered sets existieren spezielle Verzweigungsstrategien. Da nicht alle Solver diese Strukturen unterstützen, hat Kalvelagen (2003,[168]) die Verzweigungsmethode für SCVs in `GAMS` kodiert, um ein MINLP-Problem mit zusätzlichen SCVs zu lösen. Problemspezifische Verzweigungen auf anderen Strukturen wie z. B. Constraints findet man kaum in kommerziellen Solvern – ein weiteres Gebiet für PMSAs.

Automatisches Hinzufügen von Ungleichungen finden wir in Branch&Cut (B&C)-Verfahren, Äußere Approximation oder in GP wie in Abschnitt 6.4.2 diskutiert; *cf.* Karuppiah & Grossmann (2008,[173]) für globale Optimierungstechniken von nichtkonvexen MINLPs mit Strukturen, die sich für Dekomposition eignen, oder Rebennack *et al.* (2011,[247]) für ein detailliertes Tutorial über B&C-Verfahren zur Lösung des Stable-Set-Problems (SSP) sowie Rebennack *et al.* (2011,[246]) für Fortschritte auf dem Gebiet der Graphenverkleinerungsverfahren innerhalb von B&C-Verfahren für das SSP. Manchmal sind weitere problemspezifische Cuts verfügbar, die sowohl statisch oder auch dynamisch zum Problem hinzugefügt werden können. Der dynamische Fall ist natürlich der interessantere und ist durch verschiedene Beispiele in der `GAMS` Modell-Bibliothek illustriert. So werden z. B. die Tour-Eliminierungs Constraints dynamisch zum Kernmodell des

Handlungsreisenden-Problems (TSP) hinzugefügt. Zwar funktioniert diese Technik nur für kleine TSPs, aber sie zeigt, wie sich dynamisches Hinzufügen von Ungleichungen auf andere Probleme übertragen lässt.

8.1.3.2 Dekompositionsverfahren

Dekompositionsverfahren zerlegen das Problem in kleinere Probleme, die in Sequenzen oder Kombinationen gelöst werden können. Es gibt standardisierte Techniken wie Dantzig-Wolfe Dekomposition (ein spezielles Spaltenerzeugungsverfahren) zur Lösung von LP-Problemen mit speziellen Strukturen, oder Benders' Dekomposition [*cf.* Benders (1962,[34]) oder Floudas (1995,[88], Chap. 6)], aber auch strukturausnutzende, maßgeschneiderte Verfahren.

8.1.3.2.1 Benders' Dekomposition (BD) Dieses Verfahren wurde von Benders (1962,[34]) entwickelt, um MILP-Probleme in eine Folge von Masterprobleme (MP) und Unterprobleme zu zerlegen. Die MPs sind IP Probleme mit einer kontinuierlichen Variablen während es sich bei den Unterproblemen um LPs handelt. Das MP gibt eine ganzzahlige Lösung an das Unterproblem, welches entweder die Optimalität dieser Lösung im Hinblick auf das ursprüngliche Problem nachweist oder andernfalls eine Zulässigkeits- oder Optimalitätsungleichung an das MP zurück gibt. BD wurde auch auf sehr große LPs – bevorzugt in der Welt der stochastischen Optimierung, wo BD auch als L-shaped-Methode (Birge & Louveaux 2000,[37]) bekannt ist – und NLPs als sogenannte verallgemeinerte BD angewendet. Wendet man die BD auch auf die Unterprobleme an, so spricht man von einer *Nested BD*.

8.1.3.2.2 Spaltenenumerierung und Spaltenerzeugung Der Begriff *Spalten* [engl.: *column*] leitet sich aus dem Sprachumfeld der Linearen Programmierung ab. Dort steht seit den 1950er Jahren *columns* für die Variablen und *rows* für die Constraints. Im Simplexverfahren werden Nicht-Basis-Variablen als Kandidaten für mögliche Basis-Variablen im Pricing-Schritt untersucht. In dynamischen Varianten des Simplex-Verfahrens werden diese Nicht-Basis-Variablen überhaupt erst dynamisch erzeugt.

Im Zusammenhang von Spaltenenumerierung (CE, [engl.: *column enumeration*]) und Spaltenerzeugung (CG, [engl.: *column generation*]) hat der Begriff *Spalte* eine breitere Bedeutung und steht für beliebige Objekte, die in einem Modell auftreten. In Verschnittproblemen steht *Spalte* meist für Schnittmuster, in Netzwerk-Fluss-Problemen für zulässige Wege durch ein Netzwerk und in Vehicle-Routing-Problemen für Teilmengen von Aufträgen, die einem Fahrzeug oder einer zulässigen Tour zugeordnet werden können, wobei die zulässigen Touren wie z. B. bei Desrochers *et al.* (1992,[70]) mit Hilfe eines Kürzesten-Wege-Problems erzeugt werden.

Die grundlegende Idee von CG ist die Zerlegung eines Optimierungsproblems $\mathcal{P}$ in ein Masterproblem $\mathcal{P}_{\mathrm{M}}$ und ein Unterproblem $\mathcal{P}_{\mathrm{U}}$. Dabei hat die Zerlegung meist eine natürliche Interpretation. Nichtlineare Probleme $\mathcal{P}$ können dabei wie im Falle der Verschnittminimierung komplett auf die Lösung linearer Probleme zurückgeführt werden. Kritisch ist vor allem eine Zerlegung zu finden, die es erlaubt, beide Probleme $\mathcal{P}_{\mathrm{M}}$ und $\mathcal{P}_{\mathrm{U}}$ in kurzer Zeit zu lösen. Das berühmteste Beispiel ist wohl das in Abschnitt 8.2.3 beschriebene Spaltenerzeugungsverfahren von Gilmore & Gomory (1961,[106]; 1963,[107]) zur Berechnung der minimalen Anzahl von Rollen zur Erfüllung der Nachfrage kleinerer

Auftragsrollen gegebener Breite. Die monolithische Version dieses Problems führt auf ein MINLP-Problem mit einer sehr großen Anzahl von Variablen.

In einfachen Fällen ist es möglich, alle Spalten *a priori* zu erzeugen (CE). Falls dies nicht möglich ist, werden die Spalten dynamisch zum Problem hinzu gefügt; gute Überblicksaufsätze zur dynamischen Spaltenerzeugung sind zum Beispiel Barnhart *et al.* (1998,[23]) oder Lübbecke & Desrosiers (2005,[197]).

CE kann als eine spezielle Variante von CG angesehen werden, die anwendbar ist, wenn nur eine geringe Anzahl von Spalten existiert oder die Betrachtung einer kleinen Anzahl von Spalten ausreicht. In praktischen Verschnittproblemen ist dies zum Beispiel der Fall, da meist nur Muster mit einem kleinen Streifenverschnitt akzeptiert werden; dies schließt die meisten Muster bereits aus. CE führt in natürlicher Weise zu einem Auswahlproblem der vorhandenen Spalten; dieses Auswahlproblem wird auch *Partitionierungsmodell* genannt. Schrage (2006,[262], Sect. 11.7) gibt illustrative Beispiele aus den Anwendungsgebieten Zerlegung, Matching, Überdeckung, Partitionierung und Packprobleme. Ein allgemeiner Rahmen für die Zerlegung einer gegebenen Menge von Objekten in Teilmengen und nachfolgender Zuordnung zu Bestandsobjekten wurde von Rebennack *et al.* (2009,[243]) entwickelt und auf ein Verschnittproblem aus der Metallindustrie angewendet. Trotz der Limitierung hinsichtlich der Anzahl der Spalten bietet CE einige wichtige Vorteile. Hierzu zählen die einfache Anwendbarkeit auf MIP-Probleme, die fehlende Notwendigkeit ein Pricing-Problem zur Erzeugung neuer Spalten lösen zu müssen und der geringe Implementierungsaufwand.

8.1.3.2.3 Column Generation und Branch&Price Allgemein betrachtet können wir CG verwenden um wohl-strukturierte MILP-Probleme mit mehreren Hunderttausenden oder Millionen von Variablen, d. h. Spalten, zu lösen. Diese Probleme führen auf sehr große LP-Probleme, wenn man die Ganzzahligkeitsbedingungen relaxiert. Wenn das LP-Problem derartig viele Variablen (columns) enthält, dass es nicht mit einem direkt LP-Solver (revised simplex, interior point method) gelöst werden kann, beginnt man im sogenannten *Master Problem* (MP) mit einer kleinen Teilmenge von Variablen und erhält so das *eingeschränkte MP* (RMP). Nach Lösung des RMP werden in einem Pricing Problem (PP) neue Variablen identifiziert. Dieser Schritt entspricht der Identifizierung von Nicht-Basis-Variablen, die in die Basis des Simplex-Verfahrens aufgenommen werden und führte zu der Bezeichnung *column generation*. Das RMP wird nun mit der neuen Anzahl von Variablen gelöst. Das Verfahren terminiert, wenn mit Hilfe des Pricing-Problems keine neuen Variablen mehr identifiziert werden können. Die einfachste Variante des CG-Verfahrens finden wir in der Dantzig-Wolfe-Dekomposition (Dantzig & Wolfe 1960,[60]).

Gilmore & Gomory (1961) waren die ersten, die die Idee der CG auf ein IP-Problem angewendet haben: Das Verschnittproblem in der Papierindustrie. In diesem Fall handelt es sich auch beim PP um ein IP-Problem; das PP ist ein Rucksackproblem – hier werden die neuen *Spalten* erzeugt. Dieses Verschnittproblem ist einzigartig in dem Sinne, dass die durch das RMP erzeugten Spalten – hier Schnittmuster – hinreichend sind, um auch die optimale ganzzahlige Lösung des Problem zu erzeugen. Dies ist im Allgemeinen nicht so. Enthalten beide Probleme, $\mathcal{P}_{\mathrm{M}}$ und $\mathcal{P}_{\mathrm{U}}$, ganzzahlige Variablen, so ist die CG in ein B&B-Verfahren eingebettet: Dies nennt man daher *Branch&Price* (B&P); *cf.* Barnhart *et al.* (1998,[23]) und Savelsbergh (2001,[257]). In Kürze: B&P ist IP mit CG. Während des Verzweigungsprozesses werden neue Spalten erzeugt – eine andere Begründung für den Begriff *Branch&Price*. B&P (häufig auch mit Branch&Cut verknüpft)

sind maßgeschneiderte Implementierungen mit Zerlegungsstrukturen. Trotz des erheblichen Implementierungsaufwandes gehört B&P zu den leistungsvollsten Techniken zur Lösung von MIP-Problemen mit CG. Eine Liste erfolgreicher Anwendungen in verschiedenen Gebieten findet sich bei Kallrath (2008,[159]).

8.1.3.3 Lagrange-Relaxierung

Lagrange-Relaxierung (LR) – *cf.* Geoffrion (1974,[100]), Fisher (1985,[86]), Martin (1999, [204]) – ist weniger ein mathematisches Verfahren zur Bestimmung zulässiger Punkte, sondern stellt ein wichtiges Verfahren zur Verbesserung der unteren Schranke eines Minimierungsproblems dar, wobei duale Informationen verwendet werden. Das in Abschnitt 4.3.2 verwendete Branch&Bound-Verfahren zur Lösung von MILP-Problemen basiert zunächst auf einer Relaxierung des LP-Problems; die Relaxierung resultiert dabei aus einer Vernachlässigung der Ganzzahligkeitsbedingung der diskreten Variablen. Bei kombinatorischen Problemen wie z. B. beim Problem des Handlungsreisenden ist diese Bereichs-Relaxierung jedoch nur sehr schwach. Die LR schneidet da wesentlich besser ab, wie wir später anhand des Verallgemeinerten Zuordnungsproblems [engl.: *generalized assignment problem*] demonstrieren werden. Sie ist daher besonders im Umfeld diskreter Optimierung eine weit eingesetzte Technik. Die Grundidee ist hierbei, problematische Restriktionen zu vernachlässigen und diese stattdessen multipliziert mit einem zu bestimmenden Lagrange-Multiplikator in die Zielfunktion aufzunehmen. Falls – für fixierte Lagrangemultiplikatoren – das resultierende Optimierungsproblem leichter zu lösen ist, besteht gute Hoffnung, dass die untere Schranke effizient verbessert werden kann, indem mit Hilfe eines Subgradientenverfahrens die Lagrangemultiplikatoren iterativ bestimmt werden. Betrachten wir hierzu das (gemischt)-ganzzahlige lineare Programm P:

$$\begin{aligned} \min_{\mathbf{x}} z &= \mathbf{c}^{\mathrm{T}}\mathbf{x} \\ \mathsf{A}\mathbf{x} &\geq \mathbf{b} \\ \mathsf{B}\mathbf{x} &\geq \mathbf{d} \\ \mathbf{x} &\geq 0 \quad , \quad x_j \in \mathbb{N}_0 \quad \forall j \in J \end{aligned}$$

Zunächst ist stets zu entscheiden, welche Constraint dualisiert werden soll. Falls wir annehmen, dass sich das Problem

$$\begin{aligned} \min_{\mathbf{x}} z &= \mathbf{c}^{\mathrm{T}}\mathbf{x} \\ \mathsf{B}\mathbf{x} &\geq \mathbf{d} \\ \mathbf{x} &\geq 0 \quad , \quad x_j \in \mathbb{N}_0 \quad \forall j \in J \end{aligned}$$

relativ einfach lösen lässt, ist es eine gute Idee, $\mathsf{A}\mathbf{x} \geq \mathbf{b}$ zu relaxieren; das wollen wir im Folgenden nun vorbereiten. Mit der hinsichtlich der Ungleichung $\mathsf{B}\mathbf{x} \geq \mathbf{d}$ ganzzahlig zulässigen Menge

$$\Gamma := \{\mathbf{x} | \mathsf{B}\mathbf{x} \geq \mathbf{d}, x_j \in \mathbb{N}_0 \quad \forall j \in J\}$$

aller Punkte $\mathbf{x}$ können wir P auch schreiben als

$$\begin{aligned} \min_{\mathbf{x}} z &= \mathbf{c}^{\mathrm{T}}\mathbf{x} \\ \mathsf{A}\mathbf{x} &\geq \mathbf{b} \quad , \quad \mathbf{x} \in \Gamma \quad . \end{aligned}$$

Falls P eine optimale Lösung besitzt

$$\begin{aligned} \min_{\mathbf{x}} z &= \mathbf{c}^{\mathrm{T}}\mathbf{x} \\ \mathsf{A}\mathbf{x} &\geq \mathbf{b} \quad , \quad \mathbf{x} \in \Gamma \quad , \end{aligned}$$

und conv(Γ) die Relaxierung von Γ bzw. der konvexen Hülle von Γ bezeichnet, die man erhält, wenn man die diskrete Punktmenge Γ durch deren konvexe Hülle ersetzt, so gilt mit dem durch diese Lagrange-Dualisierung gewonnene Problem D

$$\begin{aligned} L(\mathbf{u}) &= \min\left\{\mathbf{c}^{\mathrm{T}}\mathbf{x} - \mathbf{u}^{\mathrm{T}}\left(\mathsf{A}\mathbf{x} - \mathbf{b}\right) | \mathbf{x} \in \Gamma\right\} \\ &= \min\left\{\mathbf{u}^{\mathrm{T}}\mathbf{b} + \mathbf{c}^{\mathrm{T}}\mathbf{x} - \mathbf{u}^{\mathrm{T}}\mathsf{A}\mathbf{x} | \mathbf{x} \in \Gamma\right\} \\ &= \min\left\{\mathbf{b}^{\mathrm{T}}\mathbf{u} + \left(\mathbf{c} - \mathsf{A}^{\mathrm{T}}\mathbf{u}\right)^{\mathrm{T}}\mathbf{x} | \mathbf{x} \in \Gamma\right\} \end{aligned}$$

die Gleichung

$$\min\{\mathbf{c}^{\mathrm{T}}\mathbf{x} | \mathsf{A}\mathbf{x} \geq \mathbf{b}, \mathbf{x} \in \mathrm{conv}(\Gamma)\} = \max\{L(\mathbf{u}) | \mathbf{u} \geq 0\} = \max D \quad ,$$

d. h. die optimale Lösung des relaxierten Problems lässt sich aus der optimalen Lösung des zu P dualen Lagrangeproblems gewinnen. Bezeichnet $\bar{\Gamma}$ die LP-Relaxierung von Γ, d. h. wir vernachlässigen die Ganzzahligkeitsbedingung von x, und relax(P) die LP-Relaxierung von P, so gilt wegen der Relaxierungseigenschaften

$$\Gamma \subseteq \mathrm{conv}(\Gamma) \subseteq \bar{\Gamma}$$

die Ungleichungskette

$$\min[\mathrm{relax}(P)] \leq \min[\mathrm{conv}(P)] = \max D \leq \min P \quad ,$$

d. h. für jedes nichtnegative $\mathbf{u}$ gibt uns die Lagrange-Relaxierung eine untere Schranke von P, was sich auch direkt aus der Struktur von $L(\mathbf{u})$ erschließt, denn für jeden zulässigen Punkt $\mathbf{x}$ und nichtnegativen $\mathbf{u}$ wird mit $\mathbf{u}^{\mathrm{T}}\left(\mathsf{A}\mathbf{x} - \mathbf{b}\right)$ ein positiver Term abgezogen. In diesem Sinne ist das zu lösende Problem D eine Relaxierung von P. Im Falle $\min[\mathrm{relax}(P)] = \min P$ stimmen LP- und Lagrange-Relaxierung überein; in anderen Fällen ist die LR meist stärker als die LP-Relaxierung. Bei $L(\mathbf{u})$ handelt es sich um eine stückweise lineare und konkave Funktion; sie ist somit nach $\mathbf{u}$ differenzierbar bis auf eine endliche Menge diskreter Punkte $\bar{\mathbf{u}}$. Es ist die fehlende Differenzierbarkeit an endlich vielen Punkten, die auf das Konzept des *Subgradienten* führt. Ein reeller Vektor $\boldsymbol{\gamma} \in \mathbb{R}^m$ ist Subgradient von $L(\mathbf{u})$ in einem Punkt $\bar{\mathbf{u}}$ genau dann, wenn gilt

$$L(\mathbf{u}) \leq L(\bar{\mathbf{u}}) + \left(\mathbf{u} - \bar{\mathbf{u}}\right)^{\mathrm{T}}\boldsymbol{\gamma} \quad .$$

Falls $L(\mathbf{u})$ nach $\mathbf{u}$ differenzierbar ist, entspricht $\boldsymbol{\gamma}$ dem Richtungsvektor der Tangentengerade. Im nichtdifferenzierbaren Fall entspricht $L(\bar{\mathbf{u}}) + \left(\mathbf{u} - \bar{\mathbf{u}}\right)^{\mathrm{T}}\boldsymbol{\gamma}$ einer Geraden, die in mit $L(\mathbf{u})$ übereinstimmt, ansonsten aber $L(\mathbf{u})$ überschätzt; es gibt derer unendlich viele. Die Menge aller Subgradienten $\boldsymbol{\gamma}$ in einem Punkt nennt man auch Subdifferential $\mathcal{G}(\bar{\mathbf{u}})$ von $L(\mathbf{u})$ in $\bar{\mathbf{u}}$. Im differenzierbaren Fall enthält das Subdifferential nur einen Subgradienten: den Gradienten $\nabla L = \partial L/\partial \mathbf{u}$. Mit dem Konzept des Subgradienten kann nun das Optimum der nichtdifferenzierbaren Funktion $L(\mathbf{u})$ zum Beispiel mit Hilfe des in Abschnitt 8.1.3.3.1 beschriebenen Subgradientenverfahrens berechnet werden.

8.1.3.3.1 Das Subgradientenverfahren Das Subgradientenverfahren erweitert das Verfahren des steilsten Anstiegs [engl.: *method of steepest ascent*] auf stückweise-lineare konvexe oder konkave Funktionen wie $L(\mathbf{u})$. Zwar führt ein Schritt in Richtung eines Subgradienten nicht notwendigerweise zu einer Verbesserung von $L(\mathbf{u})$, dennoch ist es sinnvoll, der Richtung eines Subgradienten zu folgen und im Subgradientenverfahren wie folgt iterativ vorzugehen:

Schritt 0: Lösung der LP-Relaxierung, wobei die zu relaxierende Constraints im Modell mit enthalten sein müssen. Aus der Lösung dieses Problems folgen die Dualwerte der zu relaxierenden Constraints. Zudem wird eine primale Lösung benötigt, aus der sich eine obere Schranke berechnen lässt.

Schritt 1: Ein Iterationszähler k wird auf $k = 0$ initialisiert. Ein Vektor $\mathbf{u}^0 > 0$ muss bekannt sein; hier können zum Beispiel aus der LP-Relaxierung die mit den dualisierten Constraints assoziierten dualen Variablen verwendet werden. Schließlich wird noch eine Genauigkeitsschranke $\delta > 0$ definiert.

Schritt 2: Berechne den Subgradienten $\boldsymbol{\gamma}^k = \mathbf{g}(\mathbf{x}^k) = \mathbf{b} - \mathsf{A}\mathbf{x}^k$, wobei $\mathbf{g}(\mathbf{x}^k)$ im differenzierbaren Fall den Gradienten von $L(\mathbf{u})$ in $\mathbf{u}^k$ bezeichnet und $\mathbf{x}^k \in \Gamma(\mathbf{u}^k)$, d. h. man muss für festes $\mathbf{u} = \mathbf{u}^k$ den Wert der Lagrangefunktion $L(\mathbf{u}^k)$ sowie $\mathbf{x}^k$ als optimale Lösung des dualen Problems

$$\begin{aligned} L^k &= L(\mathbf{u} = \mathbf{u}^k) = \min_{\mathbf{x}} \left\{ \mathbf{b}^{\mathrm{T}}\mathbf{u} + \left(\mathbf{c} - \mathsf{A}^{\mathrm{T}}\mathbf{u}\right)^{\mathrm{T}} \mathbf{x} \right\} \\ \mathsf{B}\mathbf{x} &\geq \mathbf{d}, x_j \in \mathbb{N}_0 \quad \forall k \in k \end{aligned}$$

bestimmen.

Schritt 3: Setze $\mathbf{u}^{k+1} := \max\{0, \mathbf{u}^k + t_k \boldsymbol{\gamma}^k\}$ mit einer positiven (skalaren) Schrittweite t_k, die geeignet zu steuern ist.

Schritt 4: Falls $\Delta^k := \left\|\mathbf{u}^{k+1} - \mathbf{u}^k\right\| < \varepsilon$, wird terminiert, andernfalls wird j um eins erhöht und mit Schritt 2 fortgefahren.

Die Steuerung der Schrittweite t_j – *cf.* Bazaraa & Sherali (1981,[25]) – ist heuristischer Natur und bedarf einer problemspezifischen Anpassung, folgt aber dem bedeutenden theoretischen Resultat von Poljak (1967, [234]) abgeleiteten Resultat, dass die Folge $\{L^k\}$ gegen den Grenzwert $L(\bar{\mathbf{u}})$ konvergiert, wenn die Folge $\{t_k\}$ gegen Null konvergiert und für die Summe der Folgenglieder

$$\sum_{k=0}^{\infty} t_k = \infty$$

gilt. Diese Divergenz garantiert letztlich, dass die Schrittweiten *nicht zu klein* werden. Held *et al.* (1974,[132]) geben

$$t_k := \theta_k \frac{\left[L_{\mathrm{U}} - L^k\right]}{\left\|\boldsymbol{\gamma}_k\right\|^2}$$

mit $0 < \varepsilon \leq \theta_k \leq 2$ als Empfehlung, wobei L_{U} eine obere Schranke für $L(\mathbf{u})$ darstellt. Als Konsequenz des schwachen Dualitätstheorems kann man für L_{U} jede primale Lösung wählen. Die Lösung des primalen Problems zur Bestimmung von L_{U} zwingt uns aber, wenigstens eine ganzzahlig zulässige Lösung zu bestimmen; in manchen Fällen kann hier auch eine konstruktive Heuristik helfen. Bei der Initialisierung von $\mathbf{u}^0$ genügte der Rückgriff auf die primale Lösung der LP-Relaxierung. Anfänglich kann man θ_0 auf $\theta_0 = 2$ setzen und halbiert den Wert von θ_k, wenn L^k im Verlaufe mehrerer Iterationen nicht anwächst.

8.1.3.3.2 Beispiel: Erweitertes Zuordnungsproblem Als Beispiel betrachten wir eine Menge von Aufgaben $i \in I$, die verschiedenen Maschinen $j \in J$ zugeordnet werden sollen, wobei jede Aufgabe i nur genau einer Machine j zugeordnet wird. Die Zuordnung $i \to j$ verursacht Kosten c_{ij} und nimmt Maschinenkapazität C_{ij} in Anspruch. Sind wir an einer kostenminimalen Lösung interessiert, so kann diese durch das verallgemeinerte Zuordnungsproblem [engl.: *generalized assignment problem*]

$$\begin{aligned} \min \sum_i \sum_j c_{ij} x_{ij} & & & & & \\ \sum_j x_{ij} & = & 1 & , & \forall i & \quad (8.1.1) \\ \sum_j C_{ij} x_{ij} & \leq & d_j & , & \forall j & \quad (8.1.2) \\ x_{ij} & \in & \{0,1\} & , & \forall \{ij\} & \end{aligned}$$

berechnet werden. Denken wir darüber nach, LR zu verwenden, müssen wir uns zunächst darüber klar werden, welche Restriktionen im Lagrange-Sinne relaxiert werden soll. Hier kommt (8.1.1) oder (8.1.2) in Frage; wir entscheiden uns für die Zuordnungs-Gleichung (8.1.1), da diese schwieriger ist als die Knapsack-Ungleichung (8.1.2), die zu den eher gutartigen in der Klasse der MILP-Probleme gehört – überhaupt sind Gleichungen, die diskrete Variablen enthalten meist Kandidaten für Schwierigkeiten und unangenehmen Überraschungen. Diese Wahl, (8.1.2) im Sinne von Lagrange zu relaxieren, wird um so sinnvoller, je mehr eine größere Anzahl von Aufgaben einer kleineren Anzahl von Maschinen gegenüber steht. Betrachten wir für das Beispiel nun drei Aufgaben und zwei Maschinen Kapazitäten 13 und 11 sowie Kosten und Kapazitätsinanspruchnahme

$$c_{ij} := \begin{pmatrix} 9 & 2 \\ 1 & 2 \\ 3 & 8 \end{pmatrix} \quad , \quad C_{ij} := \begin{pmatrix} 6 & 8 \\ 7 & 5 \\ 9 & 6 \end{pmatrix} \quad .$$

Um im vektoriellen Bild zu bleiben, definieren wir

$$\mathsf{A} := \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix} \quad , \quad \mathbf{b} := \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

und die Vektoren

$$\mathbf{c}^{\mathrm{T}} := (c_{11}, c_{12}, c_{21}, c_{22}, c_{31}, c_{32})$$

bzw.

$$\mathbf{x} := (x_{11}, x_{12}, x_{21}, x_{22}, x_{31}, x_{32})^{\mathrm{T}} \quad .$$

Damit können wir die Zuordnungsgleichung als

$$\mathsf{A}\mathbf{x} = \mathbf{b}$$

schreiben.

In **Schritt 0** lösen wir das LP-relaxierte Problem, also

$$\min \sum_i \sum_j c_{ij} x_{ij} \tag{8.1.3}$$

$$\sum_j x_{ij} = 1 \quad , \quad \forall i \tag{8.1.4}$$

$$\sum_j C_{ij} x_{ij} \leq d_j \quad , \quad \forall j \tag{8.1.5}$$

$$\mathbf{x} \geq 0$$

und erhalten als Ergebnis $z_{LP} = 6.428$; nützlich sind noch die dualen Werte $(2, 2, 4.286)$, die mit (8.1.4) assoziiert sind. Für dieses kleine Problem können wir auch leicht die optimale ganzzahlige Lösung berechnen; wir erhalten $z_{IP} = 18$ – die LP-Relaxierung liefert also nur eine recht schwache Schranke von knapp 36%.

Nun wollen wir schauen, ob uns die Lagrange-Relaxierung helfen kann. Zunächst initialisieren wir $u^0 := (2, 2, 4.286)$. Eine obere Schranke L_{U} gewinnen wir aus der offensichtlichen Lösung $x_{11} = x_{11} = x_{32} = 1$ und $x_{ij} = 0$ für alle übrigen Kombinationen von i und j, also $\sum_i \sum_j c_{ij} x_{ij} = 19$. Halten wir noch einmal gedanklich fest, wie das Problem $L(u^k)$

$$\min \sum_i \sum_j c_{ij} x_{ij} + \sum_i u^j \sum_j (1 - x_{ij}) \tag{8.1.6}$$

$$6x_{11} + 7x_{21} + 9x_{31} \leq 13 \tag{8.1.7}$$

$$8x_{12} + 5x_{22} + 6x_{32} \leq 11 \tag{8.1.8}$$

$$x_{11}, x_{12}, x_{21}, x_{22}, x_{31}, x_{32} \in \{0, 1\}$$

aufgebaut ist, so können wir das Verfahren starten:

{Eingabe}
Eine obere Schranke L_{U}
Ein Anfangswert $u^0 \geq 0$
{Initialisierung}
$\theta_0 := 2$
{Subgradientiterationen}
for $k := 0, 1, \ldots$ **do**
$\boldsymbol{\gamma}^k := \mathbf{g}(x_j) = \mathbf{b} - \mathsf{A}\mathbf{x}^k$ {Subgradient of $L(\mathbf{u}^k)$}
$t^k := \theta_k \left[L_{\mathrm{U}} - L(\mathbf{u}^k)\right] \left\|\boldsymbol{\gamma}^k\right\|^{-2}$ {Schrittweite}
$u^{k+1} := \max\{0, \mathbf{u}^k + t_k \boldsymbol{\gamma}^k\}$
if $\left\|\mathbf{u}^{k+1} - \mathbf{u}^k\right\| < \varepsilon$ **then**
Stop
end if
if keine Verbesserung in mehr als K Iterationen **then**
$\theta_{k+1} := \theta_k / 2$
else
$\theta_{k+1} := \theta_k$
end if
$k := k + 1$
end for

In der nachfolgenden Tabelle fassen wir die Ergebnisse zusammen.

k	$L(\mathbf{u}^k)$	θ^k	$\|\gamma_k\|$	t^k	Δ^k	u_1^k	u_2^k	u_3^k
1	7.000	2.0000	2.00	12.000	12.000	14.0000	14.0000	4.2857
2	2.286	2.0000	2.00	16.714	16.714	0.0000	14.0000	21.0000
3	-8.000	1.0000	2.00	13.500	13.500	13.5000	14.0000	7.5000
4	5.500	1.0000	2.00	6.750	6.750	13.5000	7.2500	14.2500
5	12.250	1.0000	1.00	6.750	6.750	13.5000	14.0000	14.2500
6	6.000	1.0000	1.00	13.000	13.000	13.5000	1.0000	14.2500
7	6.000	0.5000	1.00	6.500	6.500	13.5000	7.5000	14.2500
8	12.250	0.5000	2.00	1.688	1.688	15.1875	7.5000	12.5625
9	9.375	0.2500	2.00	1.203	1.203	13.9844	7.5000	13.7656
10	11.781	0.2500	2.00	0.902	0.902	13.0820	7.5000	14.6680
11	11.414	0.1250	2.00	0.474	0.474	13.5562	7.5000	14.1938
12	12.362	0.1250	2.00	0.415	0.415	13.9710	7.5000	13.7790
13	11.808	0.1250	2.00	0.450	0.450	13.5215	7.5000	14.2285
14	12.293	0.0625	2.00	0.210	0.210	13.7311	7.5000	14.0189
15	12.288	0.0625	2.00	0.210	0.210	13.5213	7.5000	14.2287
16	12.293	0.0313	2.00	0.105	0.105	13.6261	7.5000	14.1239
17	12.498	0.0313	2.00	0.102	0.102	13.5245	7.5000	14.2255
18	12.299	0.0313	2.00	0.105	0.105	13.6292	7.5000	14.1208
19	12.492	0.0156	2.00	0.051	0.051	13.5784	7.5000	14.1716
20	12.407	0.0156	2.00	0.052	0.052	13.6299	7.5000	14.1201
21	12.490	0.0078	2.00	0.025	0.025	13.6045	7.5000	14.1455
22	12.459	0.0078	2.00	0.026	0.026	13.6300	7.5000	14.1200
23	12.490	0.0039	2.00	0.013	0.013	13.6173	7.5000	14.1327
24	12.485	0.0039	2.00	0.013	0.013	13.6300	7.5000	14.1200
25	12.490	0.0020	2.00	0.006	0.006	13.6237	7.5000	14.1263
26	12.497	0.0020	2.00	0.006	0.006	13.6300	7.5000	14.1200
27	12.490	0.0010	2.00	0.003	0.003	13.6269	7.5000	14.1231
28	12.496	0.0010	2.00	0.003	0.003	13.6237	7.5000	14.1263
29	12.497	0.0005	2.00	0.002	0.002	13.6253	7.5000	14.1247
30	12.499	0.0005	2.00	0.002	0.002	13.6237	7.5000	14.1263
31	12.497	0.0005	2.00	0.002	0.002	13.6253	7.5000	14.1247
32	12.499	0.0002	2.00	0.001	0.001	13.6245	7.5000	14.1255
33	12.499	0.0002	2.00	0.001	0.001	13.6253	7.5000	14.1247
34	12.499	0.0001	2.00	0.000	0.000	13.6249	7.5000	14.1251
35	12.500	0.0001	2.00	0.000	0.000	13.6253	7.5000	14.1247
36	12.499	0.0001	2.00	0.000	0.000	13.6249	7.5000	14.1251
37	12.500	0.0001	2.00	0.000	0.000	13.6251	7.5000	14.1249

Da die Subgradienten γ^k gemäß

$$\gamma_i^k := 1 - \sum_j x_{ij}^k$$

berechnet werden, erhalten wie in diesem Beispiel nur die Werte 1 oder 2. Die Konvergenz in Schrittweite t^k und in den Werten von u^k ist gut zu erkennen. In der Spalte t^k ist ersichtlich, dass jedesmal, wenn $K = 2$ aufeinander folgende Werte L^k nicht zu einer Verbesserung führten, der Wert von θ halbiert wurde.

Die LR liefert uns die Schranke 12.5 – dies sind immerhin knapp 70%. Derartige

Verbesserungen gegenüber der LP-Relaxierung ermutigen auch dazu, die LR direkt in B&B-Verfahren zur Lösung von MILP-Problemen zu integrieren.

8.1.3.3.3 Variationen bei Nichtlinearen Problemen Die LR kann im Prinzip auch auf nichtlineare Probleme angewendet werden. Bei den in Abschnitt 11.3 besprochenen Zuschnittproblemen und insbesondere bei der Platzierung von Kreisen in Abschnitt 11.3.1.1 wird man die Bedingungen

$$(\mathbf{x}_i - \mathbf{x}_j)^2 \geq (R_i + R_j)^2 \quad , \quad \forall\{\ (i,j) \mid i < j\}$$

dualisieren wollen. Bei nichtkonvexen Problemen ist jedoch mit erheblichen Dualitätslücken zu rechnen, d. h. die aus der LR berechneten unteren Schranken helfen nicht wirklich. Daher sind im nichtkonvexen Fall auf exakte Straffunktionen [engl.: *exact penalty functions*] basierende Verfahren vorzuziehen; *cf.* Pillo *et al.* (2010,[232]). Bei diesen Exakten-Strafverfahren wird die Strafkonstante so gewählt, dass die optimale Lösung des Problems, welches den Strafterm enthält, mit der des ursprünglichen übereinstimmt.

8.1.4 Primale Heuristiken

Primale Heuristiken ermöglichen uns, gute primale Lösungen als Approximation für die optimale Lösung oder Startwerte für die weitere Verwendung in Verbesserungsverfahren zu berechnen. Wir unterscheiden zwischen strukturierten primalen Heuristiken und Hybrid-Verfahren. Die ersteren folgen einem bestimmten Modellierungs- bzw. algorithmischen Verfahren und können recht umfassend eingesetzt werden. Hybrid-Verfahren kombinieren z. B. exakte MIP-Algorithmen mit konstruktiven Heuristiken oder Verbesserungsverfahren – auch Metaheuristiken genannt – wie *Simulated Annealing*, *Genetische Algorithmen* oder *Tabu Suche*. Wegen der konstruktiven Heuristiken sind sie problemspezifischer als primale Heuristiken. Zwar sind beide Ansätze problemspezifisch, aber für Hybridverfahren gilt dies noch stärker.

In ihrer einfachsten Form können lokale Suchverfahren die durch eine konstruktive Heuristik bestimmte Startlösung verbessern. Bei der Lösung von MIP-Problem besteht die Grundidee primaler Heuristiken darin, ein Unterproblem auf einer kleineren Anzahl von diskreten Variablen, meist Binärvariablen, wiederholt zu lösen. Die Binärvariablen können dabei zufällig gewählt oder durch eine Metaheuristik gesteuert werden. Die gewählten Binärvariablen sind dann Gegenstand der MIP-Optimierung, während die übrigen auf ihre bereits bekannten Werte fixiert werden. Die Nachbarschaft einer diskreten Lösung kann dabei sehr vielseitig gewählt werden. Für einen Vektor aus Binärvariablen kann jeder andere Vektor Nachbar genannt werden, wenn diese Vektoren sich nur in höchstens drei Komponenten unterscheiden. In Kombination mit MIP-Techniken führen solche Nachbarschaften auf eine andere polylithische Methode namens lokale Verzweigung [engl.: *local branching*]; siehe Fischetti & Lodi 2003,[85]). In diesem Zusammenhang verweisen wir auch auf die in CPLEX oder XPRESS implementierten RINS-Heuristiken (*cf.* Danna *et al.* 2005,[58]) sowie auf die von Fischetti & Glover (2005) vorgeschlagene *feasibility pump*, die ebenfalls als polylithisches Verfahren angesehen werden kann.

Falls möglich, sollte über den primal zulässigen Punkt hinaus noch eine Schranke für die Zielfunktion abgeleitet werden. Während die primale Heuristik eine obere Schranke für ein Optimierungsproblem liefert, kann in günstigen Fällen bei Kenntnis eines zulässigen Punktes auch eine untere Schranke des ursprünglichen Problems berechnet werden.

In anderen Fällen kann eine untere Schranke durch Lösung eines Hilfsproblems berechnet werden, wobei das Hilfsproblem eine Relaxierung des ursprünglichen Problems sein sollte, wodurch es auch einfacher zu lösen ist.

Eine besondere Klasse von Problemen, die es unter Verwendung primaler Heuristiken erlaubt, untere Schranken zu berechnen, sind MILP- oder MINLP-Probleme, die kleiner werden, d. h. weniger Variablen und Constraints haben, wenn sich die Zielfunktion dem Optimum nähert. Beispiele dazu sind:

1. *Musterminimierung bei Verschnittproblemen.* Wie in Abschnitt 8.3 beschrieben, erzeugt das Hybridverfahren Lösungen mit weniger Muster. Damit kann die Indexmenge $\mathcal{J}$ der Muster verringert werden, wodurch die Anzahl der Variablen und Constraints kleiner wird, sodass schließlich eine Problemgröße erreicht wird, die es erlaubt, die minimale Anzahl der Muster direkt zu berechnen bzw. die untere Schranke zu erhöhen.
2. *Berechnung optimaler Breakpointsysteme.* Bei der stetigen, stückweise linearen Approximation nichtlinearer Funktionen durch SOS-2-Variablen berechnen Rebennack & Kallrath (2013,[241]) mit verschiedenen Verfahren minimale Breakpointsysteme, bei denen der Approximator, Über- oder Unterschätzer einer vorgegebenen Genauigkeit δ genügt. Nicht immer kann das minimale Breakpointsystem direkt durch Lösung eines MINLP-Problems bestimmt werden. In den Fällen, in denen die direkte Berechnung nicht möglich ist, erzeugt ein Hybridverfahren zunächst ein zulässiges, aber nicht notwendigerweise minimales Breakpointsystem, welches der $\delta-$Genauigkeit genügt. Damit ist nun aber eine obere Schranke für die Anzahl der Breakpoints bekannt, wodurch im direkten MINLP-Modell die Menge $\mathcal{B}$ der möglichen Breakpoints verkleinert und die Lösung des MINLP-Problems möglich wird.

Primale Heuristiken ermöglichen in günstigen Fällen also die Berechnung oberer *und* unterer Schranken, wenn das Problem bzw. die Zielfunktion des Problems eine Grundstruktur hat, die die Anzahl aktiver Objekte minimiert und die Problemgröße in der Anzahl der Objekte wächst.

8.1.4.1 Strukturierte Primale Heuristiken

Strukturierte primale Heuristiken folgen bestimmten Regeln zur Berechnung zulässiger Punkte gemischt-ganzzahliger oder nichtkonvexer, nichtlinearer Optimierungsproblems.

8.1.4.1.1 LP-Guided Dives, Relax-and-Fix Primale zulässige Lösung können durch einen systematischen Umgang mit den kritischen diskreten Variablen, meist Binärvariablen, berechnet werden. Hierzu gehören die beispielsweise die Verfahren *dive-and-fix* (DF) und *relax-and-fix* (RF), manchmal auch *fix-and-relax* (FR) genannt, sowie *moving window technique* oder *sliding window technique*, *cf.* Van Dinter *et al.* (2011,[72]). In DF wird die LP-Relaxierung eines MIP-Problems berechnet gefolgt von einer Fixierung einer Untermenge der fraktionalen Variablen auf passende Schranken. *Near-integer-fix* (NIF) ist eine Variante von DF wobei die fixierten Variablen auf den nächsten ganzzahligen Punkt fixiert werden können.

Während NIF und DF leicht zu unzulässigen Problemen führen können, ist die Situation bei RF bzw. FR etwas vorteilhafter. Bei FR handelt es sich um Fortschreitungsverfahren, indem Zerlegungsstrukturen in Produkte, Aktivitäten, Zeit oder Geometrie ausgenutzt werden. Mit Hilfe von FR hat der Autor kleine, 6×6 Instanzen des Eternity II-Problems (*cf.* Benoist & Burreau 2008,[35], or Muñoz *et al.* 2009,[217]) als MILP gelöst. Bei diesem Problem handelt es sich um ein hochgradig kombinatorisches, NP-vollständiges Edge-Matching-Problem mit wenig Struktur aber $256! \cdot 4^{256}$ möglichen Kombinationen, von denen aber wahrscheinlich die meisten unzulässig sind.

FR erfordert, dass wir die diskreten Variables δ eines originalen Problems $\mathcal{P}$ in R disjunkte oder schwach überlappende Teilmengen $\mathcal{S}_r$, $r \in \mathcal{R} := \{1, \ldots, R\}$ abnehmender Bedeutung zerlegen können. Ein Beispiel dazu ist die rollierende Produktionsplanung, in der frühe Zeitperioden wichtiger als spätere sein können. Basierend auf diesen Zerlegungen $\mathcal{S}_r$, werden R MIP-Probleme $\mathcal{P}_r$ gelöst, die schließlich einen zusammengesetzten zulässigen Punkt von $\mathcal{P}$ ergeben. Im Problem $\mathcal{P}_r$, $2 \leq r \leq R$, werden die Werte von $\delta \in \mathcal{S}_{r-1}$ auf ihre Optimalwerte aus $\mathcal{P}_{r-1}$ fixiert; Ganzzahligkeit wird nur für $\delta \in \mathcal{S}_r$ gefordert. Da $\mathcal{P}_1$ eine Relaxierung von $\mathcal{P}$ darstellt, gibt in einem Minimierungsproblem der Zielfunktionswert von $\mathcal{P}_1$ eine untere Schranke des Zielfunktionswertes von $\mathcal{P}$. Beim Übergang $r-1$ nach r von kann es vorkommen, dass $\mathcal{P}_r$ unzulässig wird. Meistens genügt es aber schon, zur Sicherung der Zulässigkeit eine kleine Überlappung zwischen $\mathcal{S}_{r-1}$ und $\mathcal{S}_r$ einzubauen, in der zusätzliche freie diskrete Variablen zum Ende der Partition $\mathcal{S}_{r-1}$ eine Verbindung zur Partition $\mathcal{S}_r$ herstellen. Auch in Fällen ohne Zulässigkeitsproblem erhöht diese kleine Überlappung die Qualität des durch FR erzeugten Punktes (δ_R, x_R), der auch ein zulässiger Punkt des ursprünglichen Problems ist. Bei Verwendung von `GAMS/CPLEX` oder `GAMS/XPress` kann der MILP-Solver direkt auf den zulässigen Punkt aufbauen. In gutartigen Fällen kann nachfolgend die Ganzzahligkeitslücke zwischen oberer und unterer Schranke verkleinert oder sogar geschlossen werden.

FR funktioniert besonders gut in der Energiewirtschaft. Als Beispiel dazu mag ein Pumpspeicherwerk dienen. Bei einer stündlichen Diskretisierung über ein Jahr und den binären Entscheidungsvariablen (Verwendung des hochgepumpten Reservoirwassers oder Energieerzeugung durch ein thermisches Kraftwerk) kommen wir leicht auf einige zigtausend Binärvariablen sowie auf ähnlich große Anzahlen von Constraints (Bilanzierung und Kopplung der Reservoirniveaus in benachbarten Zeitperioden). Ziel ist die Minimierung der Startkosten für Pumpen und thermische Kraftwerke. Da späte und frühe Perioden wegen der fehlenden Speichermöglichkeiten kaum gekoppelt sind, genügen einige wenige Partitionen, die jeweils z. B. 3 Monate umfassen, um gute zulässige Startpunkte zu generieren und nachgeschaltet mit `GAMS/CPLEX` und der `mipstart`-Funktionalität die optimale Lösung des ursprünglichen und vollständigen Problems in 2 Minuten zu berechnen.

8.1.4.1.2 SOS-2 Basierte Lineare Approximation für NLP-Probleme

Beale & Forrest (1976,[29]) wendeten die Idee stückweise linearer Approximationen auf die Berechnung globaler Minima nichtkonvexer, nichtlinearer Funktionen an. Dazu führten sie die in Abschnitt 5.7. beschriebenen strukturierten Mengen (special ordered sets of type 2) mit speziellen Verzweigungsstrategien ein. Seitdem erschienen zahlreiche Beiträge zu SOS-2, darunter Farias *et al.* (2000,[67]; 2008,[68]), mit unstetigen, separierbaren, stückweise-linearen Funktionen, Vielma *et al.* (2009,[284]) mit einer Entwicklung eines einheitlichen Zugangs und Erweiterungen auf gemischt-ganzzahlige Modelle für nichtseparierbare stückweise linearer Funktionen sowie Misener & Floudas (2010,[214]) mit stückweise linearen Formulierungen für zwei- und dreidimensionale Funktionen.

8.1.4.1.3 Homotopie-Sequenzen von Modellen In der in Abschnitt 10.1 oder auch Kallrath (1999,[154]) beschriebenen Designstudie werden die Kosten für ein standortweites Produktionsnetzwerk minimiert. Diese umfassen die Kosten für die Rohprodukte, Investitionskosten und variable Kosten für Wiederaufbereitungsanlagen sowie einen Strafkostenterm für Restverunreinigungen, die an einem bestimmten Punkt des Produktionsnetzwerkes anfallen.

Dieses infolge seiner Struktur und Größe schwierige MINLP-Problem wird in einer Sequenz aus in der Modellierungssprache `GAMS` implementierten Untermodellen verschiedener Komplexität gelöst. In diesem Homotopieverfahren wird der Lösungsvorgang eines Modells $m+1$ mit den Ergebnissen aus Modell m initialisiert. Ein einfaches lineares Modell liefert so z. B. erste Ergebnisse, um die im Pooling-Problem auftretenden Konzentrationen c_{jk}^{in} zu berechnen und zwei vereinfachte nichtlineare Modelle zu initialisieren. Die Ganzzahligkeitsbedingungen an die binären Variablen wurden zunächst relaxiert und erst im letzten Schritt berücksichtigt. Zusammengefasst wurde die folgende Sequenz von Untermodellen zunehmender Komplexität verwendet:

Teilproblem		*Problembestandteile oder Ergebnisse*
LP	$\Rightarrow$	Approximation der Konzentrationen
NLP 1	$\oplus$	Pooling, lokale WAA
NLP 2	$\Rightarrow$	relaxierte semi-kontinuierliche Ströme
RMINLP 1		Verbindungen und freie Pools
RMINLP 2		lokale WAA
MINLP		volles Modell mit allen Binärvariablen

Im Allgemeinen können mit Hilfe derartiger Homotopiemethoden zulässige Punkte aus einer Sequenz von relaxierten Modellen erzeugt werden. Beispielsweise können in einem Schedulingproblemen $\mathcal{P}$ mögliche Termine relaxiert werden, was zum relaxierten Modell $\mathcal{R}$ führt. Die optimale Lösung oder jeder zulässige Punkt von $\mathcal{R}$ ist auch ein zulässiger Punkt von $\mathcal{P}$, wenn die Termine in $\mathcal{P}$ mit Hilfe von passenden unbeschränkten Schlupfvariablen modelliert werden. Im zweiten Hilfsmodell kann dann die Summe dieser Schlupfvariablen minimiert werden.

8.1.4.2 Hybrid-Verfahren

Bei Hybrid-Verfahren werden exakte Optimierungsalgorithmen zur Lösung von LPs, MILPs, NLPs oder MINLPs mit konstruktiven Heuristiken oder Verbesserungsverfahren (lokale Suchverfahren oder Metaheuristiken wie Simulated Annealing, Genetische Algorithmen oder Tabu-Suche) kombiniert, um zulässige Punkte des Optimierungsproblems zu berechnen.

In konstruktiven Heuristiken wird die Struktur des zu lösenden Optimierungsproblems ausgenutzt, um einen primal zulässigen Punkt zu bestimmen, der evtl. durch die oben erwähnten Verbesserungsverfahren noch weiter verbessert werden kann. Damit können in MIP-Problemen den diskreten Variablen zulässige Startwerte zugeordnet werden, was bei Verwendung von z. B. `GAMS/CPLEX` *mipstart* sehr nützlich sein kann. In gutartigen Fällen, können mit Hybrid-Verfahren auch untere Schranke (bei Minimierungsproblemen) berechnet werden; ein Beispiel dafür wird in Abschnitt 8.3 gegeben. Wie auch schon die strukturiert-primalen Heuristiken in Abschnitt 8.1.4.1 sind Hybrid-Verfahren problem-spezifisch; sie stellen aber mehr noch als die ersteren maßgeschneiderte Lösungen dar, die nicht leicht auf andere Probleme übertragbar sind. Die Entwicklung

von Hybrid-Verfahren ist daher die Kunst, mit möglichst wenig Rechenzeit gute zulässige Punkte zu berechnen und dabei gilt: Alles ist erlaubt.

8.2 Rollenverschnittminimierung

Aus Bestandsrollen, machmal in der Papierindustrie auch Masterrolle oder Mutterrolle genannt, gegebener und gleicher[2] Breite B und Länge L sollen Auftragsrollen (Tambour-e) gleicher Länge L, aber kleinerer Breiten $W_i \leq B$, $i \in \mathcal{I}$, geschnitten werden, sodass der Verschnitt minimiert wird. Bei exakter Nachfrageerfüllung ist minimaler Verschnitt gleichbedeutend damit, eine minimale Anzahl von Rollen zu verwenden, um die nachgefragten Aufträge (kleine Rollen) zu produzieren.

Im Folgenden beschreiben wir in Anlehnung an Kallrath *et al.* (2013,[165]) das mathematische Modell zur Minimierung der Anzahl der Rollen einer Breite B bzw. des Verschnitts und beginnen mit den Indizes, Daten und Variablen.

8.2.1 Preliminarien

8.2.1.1 Indizes und Mengen

In diesem Modell werden die folgenden Indizes verwendet:

$i \in \mathcal{I} := \{i_1, \ldots, i_{N^{\mathrm{I}}}\}$ die vorgegebenen Aufträge bzw. Breiten; $N^{\mathrm{I}} \leq 50$.

$p \in \mathcal{P} := \{p_1, \ldots, p_{N^{\mathrm{P}}}\}$ für die Muster (Formate); $N^{\mathrm{P}} = 4000$.
Die Muster werden entweder direkt im Sinne einer vollständigen Enumerierung oder dynamisch im Sinne von Spaltengenerierung erzeugt.

8.2.1.2 Eingabedaten

Hier stellen wir die relevanten Eingabegrößen zusammen:

B [L] die Breite der Bestandsrollen.

D_i [-] die Anzahl der Aufträge für die Breite i.

M [-] die Anzahl der zur Verfügung stehenden Messer.

W_i [L] die Breite des Auftragtyps i.

8.2.1.3 Variablen

Es werden die folgenden, meist ganzzahligen Variablen verwendet:

[2] In Kallrath *et al.* (2013,[165]) ist beschrieben, wie man Bestandsrollen verschiedener Breite berücksichtigen kann.

$\mu_p \in \mathbb{N}$ [−] gibt an, wie oft das Schnittmuster (Format) p verwendet wird.
Diese die Multiplizität messende Variable kann je nach Auftragslage Werte zwischen 0 und $\max\{D_i\}$ liegen. Falls das Schnittmuster p nicht verwendet wird, ist $\mu_p = 0$.

$\alpha_{ip} \in \mathbb{N}$ [−] gibt an, wie oft die Breite i im Schnittmuster p verwendet wird.
Diese Variable kann je nach Auftragslage Werte zwischen 0 und D_i annehmen.

$\nu_i \in \mathbb{N}$ [−] gibt an, wie oft die Breite i im neuen Muster verwendet wird.
Diese Variable, die im Pricing-Problem verwendet wird, kann je nach Auftragslage Werte zwischen 0 und $\min\{D_i, M\}$ annehmen.

$w_p \geq 0$ [L] gibt den Verschnitt von Muster p an.
Diese Variable kann je nach Muster Werte zwischen 0 und B annehmen.

8.2.2 Das Modell

Das Modell umfasst eine geeignete Zielfunktion

$$\min f(\alpha_{ip}, \mu_p) \quad , \tag{8.2.9}$$

die weiter unten spezifiziert wird. Primäre Nebenbedingung ist die exakte Erfüllung der Nachfrage

$$\sum_p \alpha_{ip}\mu_p = D_i \quad , \quad \forall i \quad . \tag{8.2.10}$$

$$\sum_i W_i \alpha_{ip} \leq B \quad , \quad \forall p \quad , \tag{8.2.11}$$

Die Gleichung (8.2.10) erzwingt die Erfüllung der Nachfrage, die Rucksack-Ungleichung (8.2.11) sichert die Verträglichkeit der Muster mit der Breite B der Bestandsrollen, μ_p ist die Mustermultiplizität und α_{ip} bezeichnet die Multiplizität von Auftragsbreite i in Muster p. Schließlich sind noch die Ganzzahligkeitsbedingungen

$$\alpha_{ip} \quad \in \quad \{0, 1, 2, 3, \ldots\} \quad , \quad \forall\{ip\} \tag{8.2.12}$$

$$\mu_p \quad \in \quad \{0, 1, 2, 3, \ldots\} \quad , \quad \forall\{p\} \tag{8.2.13}$$

zu beachten. Optional ist die Beschränkung der Anzahl der Messer. Diese führt auf die Nebenbedingung

$$\sum_i \alpha_{ip} \leq M + 1 \quad , \quad \forall p | w_p = 0 \tag{8.2.14}$$

und

$$\sum_i \alpha_{ip} \leq M \quad , \quad \forall p | w_p > 0 \quad , \tag{8.2.15}$$

wobei M die Anzahl der zur Verfügung stehenden Messer bezeichnet. Bei verschnittfreien Muster werden die zur verfügenden stehenden Messer besonders günstig ausgenutzt und erlauben $M + 1$ Streifen im Muster.

Da exakte Nachfrageerfüllung gefordert ist bzw. Überproduktion ebenfalls als Abfall zu betrachten wäre, entsprechen verschnittminimale Lösungen denen mit minimaler Anzahl von Bestandsrollen, d. h. (8.2.9) wird durch die gängige Zielfunktion

$$f(\alpha_{ip}, \mu_p) = \sum_p \mu_p \tag{8.2.16}$$

ersetzt, d. h. wir minimieren einfach die Anzahl der verwendeten Bestandsrollen.

8.2.3 Struktur des Problems und Lösung mittels Spaltenerzeugung

In dieser Form handelt es sich um ein gemischt-ganzzahliges nichtlineares Optimierungsproblem (MINLP). Diese Problemklasse ist an sich schon schwierig genug. Schwerwiegender wiegt aber, dass die Anzahl der Variablen α_{ip} leicht einige Millionen betragen kann. Daher wird das Problem nicht in dieser Form gelöst, sondern verwendet im Kern das Gilmore-Gomory Verfahren [106] der dynamischen Erzeugung von Mustern.

Die Idee der dynamischen Spaltengenerierung beruht darauf, in einem Masterproblem für eine vorgegebene Menge von Muster zu bestimmen, wie oft jedes Muster zu verwenden ist sowie geeignete Eingangsdaten für ein Unterproblem zu berechnen. In dem genannten Unterproblem werden neue Muster berechnet. Das Verfahren hat allerdings in der ursprünglichen Fassung von Gilmore & Gomory (1961,[106]) zwei Nachteile. Zum einen unterstützt es nicht die Nebenbedingung (8.2.10), also exakte Nachfrageerfüllung, sondern nur die Vermeidung der Untererfüllung. Zum anderen enthält die verschnittminimale Lösung viel mehr Muster als nötig. Um den zweiten Nachteil kümmern wir uns im Abschnitt 8.3. Näherungsweise kann man exakte Auftragserfüllung erreichen durch die folgenden Schritte mit einem *column generation* (CG-)Verfahren, das die Grundidee des Gilmore&Gomory-Verfahrens verwendet, aber in einigen Punkten abgeändert wurde:

1. Initialisierung des CG-Verfahrens (zulässige Muster, statt der trivialen Muster mit jeweils nur einer Auftragsbreite können zulässige Muster direkt aus einer Rucksackungleichung erzeugt werden).
2. Iterationsschleife des CG-Verfahrens (Masterproblem und Unterproblem; das Unterproblem wird auch Pricing-Problem PP genannt).
3. Eliminierung überflüssiger Auftragsbreiten aus der CG-Lösung wo möglich; dies ist ausführlich in Kallrath *et al.* (2013, [165]) beschrieben.

Betrachten wir nun das Master- und Pricing-Problem des CG-Verfahrens im Detail, wobei wir zunächst einmal von den Messerbedingungen (8.2.14) und (8.2.15) absehen. Das Masterproblem (MP) hat die Gestalt

$$\min \sum_p \mu_p \quad , \tag{8.2.17}$$

sowie die Randbedingung (Nichtunterschreitung der Nachfrage)

$$\sum_i N_{ip}\mu_p \geq D_i \quad , \quad \forall i \quad , \tag{8.2.18}$$

wobei N_{ip} angibt, wie oft die Auftragsbreite i im Muster p enthalten ist. Statt der Ganzzahligkeitsbedingung

$$\mu_p \in \{0, 1, 2, 3, \ldots\} \quad , \quad \forall\{p\} \tag{8.2.19}$$

werden in der Iterationsschleife des CG-Verfahrens nur die Positivitätsbedingungen

$$\mu_p \geq 0 \quad , \quad \forall\{p\} \tag{8.2.20}$$

verwendet; wir betrachten also nur das relaxierte Masterproblem.

Ausgabewerte des MPs sind neben den Multiplizitäten auch die Werte der dualen Variablen oder Schattenpreise P_i , die als Eingangsinformation für das PP benötigt werden. Beim PP handelt es sich um ein Knapsack-Problem mit Zielfunktion

$$\min_{\nu_i} \left\{ 1 - \sum_i P_i \nu_i \right\} \quad , \tag{8.2.21}$$

sowie der Randbedingung (Beachtung der Breite der Rohmaterialrollen)

$$\sum_i W_i \nu_i \leq B \quad , \quad \forall i \quad . \tag{8.2.22}$$

und der Ganzzahligkeitsbedingung

$$\nu_i \in \{0, 1, 2, 3, \ldots\} \quad , \quad \forall\{i\} \quad . \tag{8.2.23}$$

Die Zielfunktion (8.2.21) entspricht den reduzierten Kosten des MPs; in abstrakter Form lauten diese für Nichtbasisvariable j

$$\bar{c}_j := c_j - \boldsymbol{\pi}^{\mathrm{T}} \mathbf{A}_j = c_j - \sum_i \pi_i A_{ij} \quad .$$

mit den dualen Variablen $\boldsymbol{\pi}$ und Spaltenvektor $\mathbf{A}_j$. In (8.2.17) sind alle Zielfunktionskoeffizienten 1, $\pi_i = P_i$ und der Spaltenvektor entspricht $\mathbf{A}_j^{\mathrm{T}} := (\nu_1, \ldots, \nu_I)^{\mathrm{T}}$; beim CG-Verfahren wird dieser Spaltenvektor erzeugt. Die Konstruktion des PP als duales Problem zum MP ist ausführlicher in Kallrath *et al.* (2013,[165]) behandelt.

Die Messerbedingungen können nun in das PP eingebaut werden, indem für verschnittbehaftete Muster die Bedingung

$$\sum_i \nu_i \leq M \tag{8.2.24}$$

hinzugefügt wird; es ist bei der Initialisierung des CG-Verfahrens zu beachten, dass dort auch nur Initialmuster zugelassen werden, die diese Bedingung erfüllen.

Da es sich beim MP um ein Minimierungsproblem handelt, terminiert das aus Lösung des MP und PP bestehenden Iterationsverfahrens, wenn $\bar{c}_j = 0$, bzw. aus numerischen Gründen, wenn $\bar{c}_j \geq -\varepsilon$, wobei ε eine passend kleine, positive Zahl ist, z. B. $\varepsilon = 10^{-3}$.

Nach Beendigung der CG-Schleife, wird das MP mit der gegebenen Mustermenge unter Berücksichtigung der Ganzzahligkeitsbedingung

$$\mu_p \in \{0, 1, 2, 3, \ldots\} \quad , \quad \forall\{p\} \tag{8.2.25}$$

gelöst. Es ist eine Besonderheit des CG-Verfahrens, dass die Existenz einer optimalen Lösung stets aus der vorhandenen Mustermenge nach Beendigung der CG-Schleife erzeugt werden kann, wobei

$$z_* := \lceil z_{\mathrm{LP}} \rceil + 1 \tag{8.2.26}$$

zu gelten scheint; es ist dies eine Vermutung [259] – streng bewiesen ist es noch nicht. Es ist nicht klar, ob dies auch für die modifizierte Variante des CG-Verfahrens gilt, aber bisher wurde kein Gegenbeispiel gefunden. Im Falle $z_* := \lceil z_{\mathrm{LP}} \rceil$ kann man sicher sein, die optimale Lösung gefunden zu haben. Im Falle $z_* := \lceil z_{\mathrm{LP}} \rceil + 1$ gibt es eine kleine Wahrscheinlichkeit dafür, die optimale Lösung nicht gefunden zu haben; dies kann seine Ursache in der Wahl des Wertes von ε haben.

8.3 Minimierung der Anzahl der Muster

In diesem Abschnitt beschreiben wir einen polylithischen Lösungsansatz zur Minimierung der Anzahl der Muster, was zu einer Minimierung des Rüstaufwandes der Schneidemaschinen führt; siehe hierzu auch Kallrath *et al.* (2013,[165]).

Die Minimierung des Verschnitts ließ sich, wie in Abschnitt 8.2.3 beschrieben, elegant mit einem Spaltengenerierungsansatz lösen. Die dabei erzeugten Schnittmuster (Formate) genügen aber nicht, um das Minimum bezüglich der Anzahl der Muster zu finden. In Belov & Scheithauer (2006,[32]) befindet sich eine Beschreibung eines Branch&Price-Algorithmus zur Lösung des eindimensionalen Verschnittproblems mit dem Ziel, die Anzahl der Muster zu minimieren. Dieses Verfahren bzw. seine Implementierung ist jedoch recht aufwendig. Denkbar wäre auch die Implementierung des Algorithmus von Vanderbeck (2000,[280]), was ebenfalls recht aufwendig ist.

Es gibt noch einen zweiten Grund, warum wir diese aufwendige Implementierungen vermeiden wollen. Im Kern geht es nämlich gar nicht allein darum, die Anzahl der Muster zu minimieren, sondern eher darum, eine praxistaugliche Lösung zu erhalten, die Verschnitt und Anzahl der Muster minimiert. In den meisten Fällen widersprechen sich jedoch beide Ziele; vor uns liegt ein typisches zwei-kriterielles Problem. Betrachten wir dazu eine Bestandsrolle der Breite 5050 mm und 5 Aufträge i mit Auftragsbreiten B_i

i	1	2	3	4	5
B_i	1250	1490	750	1700	1100
D_i	1	1	8	4	4

Das Spaltenerzeugungsverfahren nach Gilmore & Gomory berechnet die verschnittminimale Lösung mit 4 Rollen und 4 Mustern bei einem Verschnitt von 260 bzw. 1.29%:

p	μ_p	1	2	3	4	5	W_p
1	1		1	1	1	1	10
2	1			2	2		150
3	1				1	3	50
4	1	1		5			50

Bei 4 Rollen enthält die musterminimale Lösung jedoch nur 3 Mustern bei gleichem Verschnitt von 260 bzw. 1.29%,

p	μ_p	1	2	3	4	5	W_p
1	2			3	1	1	0
2	1	1		2	2		150
3	1	1	1			2	110

,

p	μ_p	1	2	3	4	5	W_p
1	4			2	1	1	750
2	1	1	1				2310

während bei 5 Rollen, die musterminimale Lösung nur noch aus 2 Mustern besteht, jedoch einem Verschnitt von 5310 bzw. 21.03% hat.

Die theoretische untere Schranke N_{m}^* für die Anzahl der Muster N_{m} können wir aus einem Bin-Packing-Problem berechnen. Das Bin-Packing-Problem erhält man aus dem Verschnittproblem, indem man die Auftragsmultiplizitäten D_i durch $D_i = 1$ ersetzt und dann die verschnittminimale Lösung mit dem Gilmore&Gomory-Verfahren berechnet. In diesem Falle ergibt sich $N_{\mathrm{m}}^* = 2$.

Dieses einfache Beispiel zeigt deutlich, die Gegenläufigkeit der beiden Ziele. Da sich die beiden gegenläufigen Ziele nur schwer auf einer gleichen monetären Basis vergleichen lassen, da die Kosten für die Umstellung der Schneidemaschinen kaum zu erfassen sind, sollen dem Benutzer verschiedene Lösungen hinsichtlich der Anzahl der Rollen und Anzahl der Muster zur Auswahl gestellt werden.

Ebenfalls gut erkennbar ist die zunehmende Mustermultiplizität in den verschnittminimalen Lösungen, die die weiter unten und in Kallrath *et al.* (2013,[165]) beschriebene Heuristik nahelegt. Mehrzielcharakter und heuristische Idee führen daher zur Entwicklung und Implementierung der nachfolgenden einfacher zu implementierenden Ansätzen:

- V1: Direkte Verwendung des Modells von Johnston & Salinlija (2004,[152]) für eine kleine Anzahl $N^I \leq 14$ von Aufträgen und $D_{\max} \leq 10$ nach einer vorherigen Berechnung geeigneter zulässiger Ungleichungen [engl.: *cuts*] bzw. oberer und unterer Schranken für die Variablen.
- V2: Heuristisches Ausschöpfungsverfahren, bei dem sukzessive Verschnittmuster durch Maximierung ihrer Multiplizitäten erzeugt werden. Dieses Verfahren wird durch $W_{\max}$, $1 \leq W_{\max} \leq 99$, parametrisiert, wobei $W_{\max}$ den zulässigen prozentualen Anfangsverschnitt der so erzeugten Muster angibt. Nach Erzeugung einiger Muster, die einen maximalen Verschnitt von $W_{\max}$ Prozent haben, können unter Umständen keine neuen Muster mit dieser Vorgabe erzeugt werden. Möglicherweise sind jedoch noch nicht alle Aufträge durch die so erzeugten Formate abgedeckt. In diesem Fall schaltet das Verfahren auf den Ansatz V1 um, wobei auf die durch $W_{\max}$ vorgegebene Bedingung nicht mehr berücksichtigt wird.

8.3.1 Indizes und Mengen

In diesem Modell werden die folgenden Indizes aus der Arbeit von Johnston & Salinlija (2004,[152]) verwendet:

$i \in \mathcal{I} := \{i_1, \ldots, i_{N^{\mathrm{I}}}\}$ die vorgegebenen Aufträge bzw. Breiten; $N^{\mathrm{I}} \leq 50$.

$j \in \mathcal{J} := \{j_1, \ldots, j_{N^{\mathrm{P}}}\}$ für die Muster (Formate); $N^{\mathrm{J}} \leq N^I$.
Die Muster werden entweder direkt durch V1 ermittelt, oder in V2 dynamisch durch Maximierung ihrer Multiplizitäten erzeugt.

$k \in \mathcal{K} := \{k_1, \ldots, k_{N^{\mathrm{P}}}\}$ für Häufigkeit, mit denen eine Breite in einem Verschnittmuster (Format) auftritt; $N^{\mathrm{K}} = 7$.
Die Häufigkeit kann durch die Breite der Aufträge und der Breite der großen Rolle begrenzt werden.

8.3.2 Eingabedaten

Hier stellen wir die relevanten Eingabegrößen zusammen:

B [L] die Breite der Bestandsrollen.

D_i [-] die Anzahl der Aufträge für die Breite i.

W_i [L] die Breite des Auftragtyps i.

8.3.3 Variablen

Es werden die folgenden – meist ganzzahligen – Variablen verwendet:

$a_{ijk} \in \mathbb{N}$ [–] gibt an, wie oft das Schnittmuster j verwendet wird, falls $x_{ijk} = 1$. Falls $x_{ijk} = 1$, gilt automatisch $a_{ijk} = r_j$.

$p_j \in \{0, 1\}$ [–] gibt an, ob das Schnittmuster j verwendet wird.

$r_j \in \mathbb{N}$ [–] gibt an, wie oft das Schnittmuster j verwendet wird.
Diese die Multiplizität messenden Variable kann je nach Auftragslage Werte zwischen 0 und $D_{\max} := \max\{D_i\}$ liegen. Falls das Schnittmuster j nicht verwendet wird, ist $r_j = p_j = 0$.

$\alpha_{ij} \in \mathbb{N}$ [–] gibt ab, wie oft die Breite i im Schnittmuster j verwendet wird.
Diese Variable kann je nach Auftragslage Werte zwischen 0 und D_i annehmen.

$x_{ijk} \in \{0, 1\}$ [–] gibt an, ob in Schnittmuster j die Breite i mit Multiplizität k enthalten ist.
Mit Hilfe dieser Binärvariablen werden letztlich Produktterme ganzzahliger Variablen vermieden; es ist dies die eigentlich bedeutsame Idee des Modells.

8.3.4 Das Ausschöpfungsverfahren im Detail

Die Idee Ausschöpfungsverfahren besteht darin, in jeder Iteration höchstens zwei Muster zu verwenden, deren Verschnitt durch den Parameter $W_{\max}$ begrenzt ist. Die Zielfunktion besteht darin, die Multiplizität dieser Muster zu maximieren. Die Lösung dient als Eingangsinformation für die nächste Iteration; die entsprechenden Variablen werden fixiert und bereits bediente Aufträge der weiteren Betrachtung entzogen. Sollte das Problem unzulässig werden, was durch die maximale Verschnittvorgabe $W_{\max}$ verursacht wird, so wird zu einem anderen Modell umgeschaltet, bei dem die Anzahl der Muster minimiert wird, wobei aber nur die Nachfrage für die unerledigten Aufträge erfüllt werden muss.

8.3.5 Das Modell

Im Kern verwenden wir die Ungleichungen (2,3,5,6,7,8,9) des von Johnston & Salinlija (2004,[152]) entwickelten MILP-Modells zur Lösung eindimensionaler Verschnittprobleme. Der nichtlineare Term $\alpha_{ij}\mu_j$ in (8.2.10) wird letztlich durch Einführung von Binärvariablen δ_{ijk} vermieden, wobei $\delta_{ijk} = 1$ indiziert, dass die Auftragsbreite i im Muster j mit der Multiplizität k auftritt. Wegen der recht hohen Anzahl von Binärvariablen ist zu erwarten, dass dieses Modell nur bei kleinen Multiplizitäten k sowie nicht zu vielen Auftragsbreiten und Mustern funktioniert.

Es ist sinnvoll und notwendig, untere und obere Schranken P^{L} und P^{U} auf die Anzahl der Muster, die in der optimalen Lösung erwartet werden, zur Verfügung zu stellen. Bezeichnet N^I die Anzahl der Auftragsbreiten, so stellt $P^{\mathrm{U}} = N^I$ eine schwache obere Schranke dar. Die untere Schranke P^{L} lässt sich mit Hilfe des CG-Verfahrens durch Lösung des zum Verschnittproblem gehörenden Bin-Packing-Problems berechnen, welches man durch Fixierung der Nachfrage der Auftragsbreiten auf $D_i = 1$ erhält.

Das Modell von Johnston & Sadinlija erlaubt uns leicht, auch andere Zielfunktionen und Nebenbedingungen zu implementieren. Nachfolgend sind die relevanten Beziehungen, die in unserem Ausschöpfungsverfahren eine Rolle spielen, zusammengestellt.

Die Binärvariable p_j möge indizieren, ob das Muster $j \in \mathcal{J}$ in der optimalen Lösung vertreten ist. Die Mustermultiplizität μ_j unterliegt den unteren und oberen Schranken M^{L} und M^{U}, d. h.

$$\mu_j \geq M^{\mathrm{L}} \quad , \quad j \in \mathcal{J}_{\mathrm{act}} := \{1, \ldots, P_{\mathrm{L}}\} \tag{8.3.27}$$

sowie

$$M^{\mathrm{L}} p_j \leq \mu_j \leq M^{\mathrm{U}} p_j \quad , \quad j \in \mathcal{J}_{\mathrm{pot}} := \{P_{\mathrm{L}} + 1, \ldots, P_{\mathrm{U}}\} \quad . \tag{8.3.28}$$

Johnston & Sadinlija überlassen es dem Anwender, M^{L} und M^{U} passend zu wählen. Für M_{L} wählt man im einfachsten Fall $M_{\mathrm{L}} = 1$, weiter wählen wir $M_{\mathrm{U}} = \max_i\{D_i\}$.

Zu jeder Binärvariable δ_{ijk} gehört eine ganzzahlige Variable α_{ijk} , die derartig konstruiert ist, dass $\delta_{ijk} = 1$ sogleich $\alpha_{ijk} = \mu_j$ impliziert. Anstatt die Nachfrage D_i zu fixieren, erlauben Johnston & Sadinlija sowohl Unter- als auch Überproduktion, d. h. $D_i^{\mathrm{L}} \leq D_i \leq D_i^{\mathrm{U}}$, und daher

$$D_i^{\mathrm{L}} \leq \sum_{j \in \mathcal{J}} \sum_k k \alpha_{ijk} \leq D_i^{\mathrm{U}} \quad , \quad \forall i \quad ; \tag{8.3.29}$$

für unseren Fall ist aber nur $D_i^{\mathrm{U}} = D_i$ relevant. Die binären Variablen x_{ijk} und die ganzzahligen Variablen α_{ijk} und μ_p sind durch

$$\alpha_{ijk} \leq M^{\mathrm{U}} x_{ijk} \quad , \quad \forall\{ijk\} \quad , \tag{8.3.30}$$

$$\sum_k x_{ijk} \leq 1 \quad , \quad \forall\{ij\} \quad , \tag{8.3.31}$$

$$\sum_k \alpha_{ijk} \leq \mu_j \quad , \quad \forall\{ij\} \quad , \tag{8.3.32}$$

und

$$M^{\mathrm{U}} \sum_k x_{ijk} - \sum_k \alpha_{ijk} + \mu_j \leq M^{\mathrm{U}} \quad , \quad \forall\{ij\} \tag{8.3.33}$$

verknüpft. Die Ungleichungen (8.3.30) und (8.3.31) garantieren, dass nur ein x_{ijk} und die zugeordneten α_{ijk} für jedes Paar (ij) ausgewählt werden kann, während (8.3.32) und (8.3.33) sicherstellen, dass positive Werte von α_{ijk} sogleich $\alpha_{ijk} = \mu_j$ implizieren.

Das Modell wird durch die Rucksack-Ungleichungen

$$\sum_i \sum_k W_i k x_{ijk} \leq B \quad , \quad \forall j \tag{8.3.34}$$

vervollständigt. Im Ausschöpfungsverfahren interessieren wir uns bei nicht erlaubter Überproduktion für die Maximierung der Mustermultiplizitäten

$$\sum_j \mu_j \tag{8.3.35}$$

sowie die Minimierung der verwendeten Muster

$$P^{\mathrm{L}} + \sum_{j \in \mathcal{J}_{\mathrm{pot}}} p_j \quad . \tag{8.3.36}$$

Schließlich benötigen wir noch die Ganzzahligkeits- bzw. Binärbedingungen

$$r_j, \alpha_{ijk} \in \{0, 1, 2, 3, \ldots\} \tag{8.3.37}$$
$$p_j, x_{ijk} \in \{0, 1\} \quad . \tag{8.3.38}$$

Im iterativem Ausschöpfungsverfahren wird das exakte MILP-Modell mehrfach mit

$$k\alpha_{ijk} \leq \tilde{D}_i$$

angewendet, wobei $\tilde{D}_i$ die Anzahl der Restaufträge für Breite i bedeutet. Insbesondere sollte gelten:

$$k\alpha_{ijk} > \tilde{D}_i \quad \Longrightarrow \quad \alpha_{ijk} = 0 \quad , \quad x_{ijk} = 0$$

und

$$\alpha_{ijk} \leq \left\lceil \frac{\tilde{D}_i}{k} \right\rceil \quad \text{bzw.} \quad \alpha_{ijk} \leq \left\lceil \frac{\tilde{D}_i + S_i}{k} \right\rceil \quad ;$$

hierbei bezeichnet S_i erlaubte Überproduktion, in unserem Fall soll jedoch stets $S_i = 0$ gelten.

Schließlich kann noch eine Modellverschärfung durch die zulässigen Ungleichung

$$r_{ij} > D_i \quad \Rightarrow \quad y_{ij} = \sum_k x_{ijk} = 0 \quad , \quad \forall\{ij\}$$

erfolgen. Die Idee ist hierbei (Überproduktion nicht erlaubt), dass eine Weite nicht im Muster j sein kann, wenn dies eine Multiplizität hat, die größer als D_i hat.

Die zulässigen Ungleichungen haben die folgende Form:

$$r_{ij} \leq D_i + A_{ij}\,(1 - y_{ij}) \quad , \quad A_{ij} := M_j - D_i \quad , \quad \forall\{ij\}$$

bzw.

$$r_j \leq M_j + (D_i - M_j)\, y_{ij} \quad , \quad \forall(ij|D_i < M_j) \quad .$$

Man bemerkt, dass im Falle $y_{ij} = 0$ die Bedingung $r_{ij} \leq M_j$ folgt, d. h. es wird nur gefordert, dass r_j seine natürliche obere Schranke nicht überschreitet. Dagegen impliziert $y_{ij} = 1$ wie gewünscht $r_j \leq D_i$.

8.3.6 Typisches Ergebnis

Mit obigem Verfahren wurde z. B. das folgende typische Ergebnis berechnet.

```
  0   4               8  99   Garantieschranke: minimale Anzahl Muster
 27   9 pat00.out     9  99   Garantieschranke: minimale Anzahl Rollen
 30   6 pat01.out     1  20
 28   6 pat02.out     1  15
 28   6 pat03.out     1  10
 27   5 pat04.out     0   8   verschnittminimal, wahrsch. musterminimal
 28   6 pat05.out     1   6
 29   6 pat06.out     1   4
Die beste gefundene Loesung enthaelt 5 Muster !
Verschnittminimale Loesungen enthalten genau 27 Rollen !

Verbesserte und optimale Garantieschranke : 5
Loesungen mit 5 Mustern sind musterminimal !
```

Zu sehen ist in der ersten Spalte die Anzahl der Rollen variierend zwischen 27 und 30, in der zweiten Spalte die Anzahl der Muster, und in der ganz rechten Spalte die Variation der Startwerte $W_{\max}$ für den maximal erlaubten Verschnitts von 20% bis 4%, die Garantieschranke 4 aus dem Bin-Packing Problem. Beachtenswert ist die Tatsache, dass diese theoretische Schranke verbessert werden kann, wenn aus der Heuristik ($W_{\max} = 8$) schon bekannt ist, dass z. B. nur 5 Muster nötig sind. Damit wird das an Johnson & Salinjaha (2004,[152]) angelehnte Modell erheblich kleiner und lässt die minimale Bestimmung der Anzahl der Muster direkt zu. In diesem Beispiel ist die Lösung mit 27 Rollen und 5 Mustern sogar hinsichtlich beider Zielkriterien optimal.

Für den praktischen Gebrauch wird der Benutzer hier die Lösung mit 27 Rollen und 5 Mustern verwenden, da diese minimal in der Anzahl der Rollen und der Anzahl von Mustern ist. Dies muss aber nicht immer so sein, wie die nachfolgende Lösung für eine Auftragssituation mit 12 Auftragsbreiten zeigt.

```
  0   5               8  99   Garantieschranke: minimale Anzahl Muster
 30  10 pat00.out     9  99   Garantieschranke: minimale Anzahl Rollen
 34   7 pat01.out     0  20
 31   9 pat02.out     1  15
 30   8 pat03.out     0  10   Verschnittminimal
 32   9 pat04.out     1   8
 30   8 pat05.out     0   6   Verschnittminimal
 31   8 pat06.out     1   4
Die beste gefundene Loesung enthaelt 7 Muster !
Verschnittminimale Loesungen enthalten genau 30 Rollen !

neue Loesung mit nur 6 Muster und 36 Rollen gefunden: patnew.out
 36   6   patnew.out  0  99

Verbesserte Garantieschranke :  6
Loesungen mit 6 Mustern sind musterminimal !
```

Hier wird der Anwender wahrscheinlich die Lösung mit 30 Rollen und 8 Mustern bevorzugen, da der Verschnitt bei 7 Mustern schon erheblich ist und 4 zusätzliche Rollen kostet; bei 6 Rollen sind es sogar 36 Rollen.

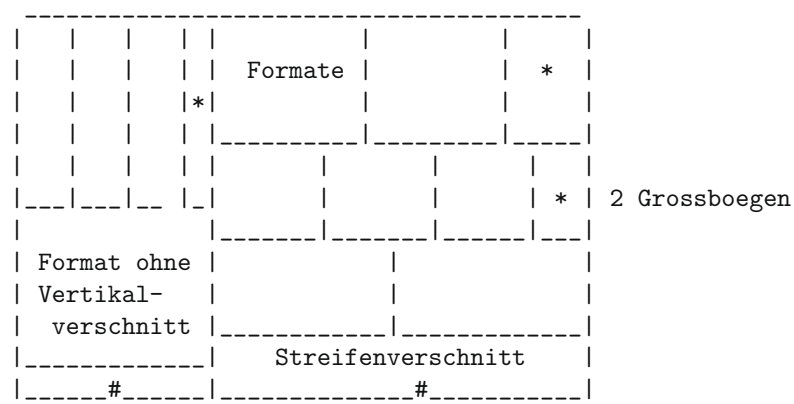
8.4 2D-Format-Produktion – Verschnittminimierung

In diesem Anwendungsbeispiel – siehe hierzu auch Kallrath *et al.* (2013,[165]) – sollen rechteckige Papierformate, z. B. DIN-A3 oder DIN-A4 Papier oder andere Größen aus gegebenen Bestandrollen geschnitten werden. Dabei wird operativ wie folgt vorgegangen. Von den Bestandsrollen $r \in \mathcal{R}$ verschiedener Breite B_r, z. B. 152 bzw. 102 cm, werden gleiche Großbögen [engl.: *sheets*] am Querschneider abgeschnitten; dabei wird angenommen, dass von jeder Breite genügend Mutterrollen vorhanden sind (keine Lager bzw. Verfügbarkeitsproblematik). Die Länge eines Großbogens ergibt sich aus der Verteilung der einzelnen Formate auf dem Bogen. Die Einzelformate (für diese liegt jeweils eine bestimme Auftragsnachfrage D_a vor) werden auf dem Planschneider durch Quer-/und Längsschnitte erstellt.

Die Lage der Einzelformate auf dem Großbogen soll immer gleich sein und definiert ein Muster oder Format, kann jedoch bei Wechsel der Ausgangsbreite der Rolle (von 150 auf 120 cm) auch wechseln. Die Einzelformate können auf dem Großbogen quer oder längs angeordnet werden (Rotation). Der Benutzer soll die Möglichkeiten erhalten, vorzugeben, ob die Einzelformate nur Längs (=Schmalbahnen) oder Quer (=Breitbahn) angeordnet werden sollen. Die Schneidereihenfolge ist also wie folgt:

1. Abschneidung eines Großbogens der Länge ℓ von der Mutterrolle.
2. Partitionierung eines Großbogens in Längsstreifen (verschiedener) Breite; dabei fällt ein Streifenverschnitt an.
3. Jeder Längsstreifen wird dann mittels Querschnitte in die Einzelformate; dabei kann in jedem Längsstreifen ein Vertikalverschnitt anfallen.

Die Länge der Großbögen kann nur zwischen $L_{\min} \le \ell \le L_{\max}$ mit $L_{\min} = 38$ cm und $L_{\max} = 163$ cm variieren; dies ist durch den Querschneider bedingt. Zugelassen sind auch nur die Breiten $B_{\min} \le b \le B_{\max}$ mit $B_{\min} = 50$ cm und $B_{\max} = 162$ cm. Die maximale Breite $B_{\max}$ entspricht der Breite der größtmöglichen Rollen.



Ein Beispiel mit 2 Großbögen (Rolle liegend) ist in obiger Abbildung dargestellt; mit # ist Streifenverschnitt, mit * Vertikalverschnitt bezeichnet. Der erste Großbogen enthält 2 Längsstreifen; der zweite 3. In jedem Großbogen gibt es jeweils einen Längsstreifen, der gar keinen Verschnitt hat. Ein typisches Auftragsbeispiel – hier mit 3 Auftragsrechtecken und $D_a = 8000, 8000$ bzw. 20000 – könnte so aussehen:

```
A1 ,      36.0 x  80.0 ,                 8000 ,    8000
A2 ,      26.0 x  85.0 ,                 8000 ,    8000
A3 ,      24.0 x  33.0 ,                20000 ,   20000

The sheets in detail:

roll width sheet  NPAT    A(i)   B(i)    ALPHA NTOT  VLOSS
                                             n    L       sTRL tTRL  %TRL

r1, 102.0,   p1 , 3 x [ 33.0 , 24.0 ] x    2    6      0.0
                                         1996 ,  48.0 ,   3.0 , 144 2.94

r1, 102.0,   p4 , 3 x [ 26.0 , 85.0 ] x    1    3      0.0
                  1 x [ 24.0 , 33.0 ] x    2    2    456.0
                                           1 ,  85.0 ,   0.0 , 456 5.26

r2, 152.0,   p4 , 3 x [ 24.0 , 33.0 ] x    2    6    432.0
                  1 x [ 80.0 , 36.0 ] x    2    2      0.0
                                         1335 ,  72.0 ,   0.0 , 432 3.95

r2, 152.0,   p7 , 2 x [ 26.0 , 85.0 ] x    1    2    572.0
                  3 x [ 33.0 , 24.0 ] x    4   12      0.0
                                           1 ,  96.0 ,   1.0 , 668 4.58

r2, 152.0,   p21, 2 x [ 36.0 , 80.0 ] x    1    2    360.0
                  3 x [ 26.0 , 85.0 ] x    1    3      0.0
                                         2665 ,  85.0 ,   2.0 , 530 4.10
```

Im Beispiel schön ersichtlich die Rotation: 24x33 wird als 33x24 geschnitten. Es bedeuten:

ALPHA	α	Multiplizität im Großbogen
L	ℓ_{rj}	Länge des Großbogen $= B_i\alpha$ für Streifen ohne Vertikalverschnitt
n	μ_{rj}	Multiplizität des Bogens in der gewählten Rolle vom Typ r
NPAT		Multiplizität im L-Muster
NTOT		Anzahl der Formate $A_i \times B_i$ im Großbogen
sTRL		Streifenverschnitt in Längeneinheiten
tTRL		Gesamtverschnitt (in Flächeneinheiten) $= \sum_i L \times sTRL$
%tTRL		prozentualer Gesamtverschnitt bezogen auf die Fläche eines Bogens
VLOSS		Vertikalverschnitt (in Flächeneinheiten) $= \sum_i NPAT \times A_i\,(L - B_i\alpha)$

Die geforderten Auftragsmengen D_a sollen nicht unterschritten werden; Überproduktion ist erlaubt, es soll aber möglich sein, diese durch Spezifikation von D_a^{over} zu beschränken und im Einzelfall auch komplett durch Fixierung von $D_a^{\text{over}} = 0$ zu verbieten. Besondere

Auftragssituationen liegen im sogenannten *Schnipselfall* vor; dieser zeichnet sich durch kleine Rechtecke, z. B. 1 cm x 1 cm, und sehr große Werte von $D_a \approx 1,000,000$ aus.

Im Folgenden wird das mathematische Modell zur Minimierung des aus Streifen- und Vertikalverschnitt resultierenden Gesamtverschnitts beschrieben. Das Lösungsverfahren nutzt die Schneidereihenfolge aus und löst ein mehrstufiges Verschnittproblem mit Guillotine-Schnitten. Dadurch wird erst eine Dekomposition in Master- und Unterproblem möglich; andernfalls hätte man ein MINLP-Problem zu lösen. Das Dekompositionsverfahren ist exakt. Ergebnis sind die folgenden verschnittoptimalen Entscheidungsvorschläge berechnet werden:

1. Auswahl der Masterrollen bestimmter Breite B_r,
2. die Länge ℓ und Anzahl n der Großbögen, die von Masterrolle r geschnitten werden, und
3. die Formate (Längsstreifen) innerhalb der Großbögen.

Zu beachten ist, dass die Teilprobleme exakt gelöst werden. Einzig bei der Erzeugung der Streifenmuster wird der Lösungsraum mit Hilfe der zulässigen Obergrenze für den Streifenverschnitt reduziert.

8.4.1 Indizes und Mengen

In diesem Modell werden die folgenden Indizes verwendet:

$a \in \mathcal{A} := \{a_1, \ldots, a_{N^{\mathrm{I}}}\}$ die vorgegebenen Auftragsrechtecke; $N^{\mathrm{A}} \leq 25$.
Die Rechtecke (Einzelformate) sind hierbei noch generisch hinsichtlich ihrer Orientierung in den Längsstreifen beschrieben. Wir bezeichnen dies mit $[A_a, B_a]$, wobei es hier aber nicht auf die Reihenfolge nicht ankommt, d. h. $[A_a, B_a] = [B_a, A_a]$. Die Schreibweise $[A_a, B_a]$ oder $[B_a, A_a]$ referiert also lediglich auf ein Rechteck mit Seiten A_a und B_a ohne dabei spezifisch auf Länge und Breite zu achten. Ein zusätzliches Attribut zeigt an, ob die Rechtecke als orientierte Rechtecke (1) oder als solche angesehen werden sollen, die noch rotiert werden dürfen (2).

$i \in \mathcal{I} := \{i_1, \ldots, i_{N^{\mathrm{I}}}\}$ die vorgegebenen orientierten Auftragsrechtecke; $N^{\mathrm{I}} \leq 50$.
Hier ist die Orientierung vorgegeben. Man erhält die orientierten Auftragsrechtecke durch Duplizierung der unorientierten Auftragsrechtecke. Die Seite A_i wird parallel zur Breite W_r eines Großbogens gemessen; die Seite B_i verläuft parallel zum Längsstreifen. Wir schreiben hierfür $[A_i \times B_i]$ und dürfen die Reihenfolge nicht vertauschen.

$j \in \mathcal{J} := \{j_1, \ldots, j_{N^{\mathrm{P}}}\}$ für die Muster (Großbögen, Formate); $N^{\mathrm{J}} = 4000$.
Die Muster der Großbögen, d. h. die Breiten der Längsstreifen werden durch vollständige Enumerierung erzeugt.

$r \in \mathcal{R} := \{r_1, \ldots, r_{N^{\mathrm{R}}}\}$ für die Bestandsrollen; $N^{\mathrm{R}} = 10$.
Die Bestandsrollen sind durch ihre Breite W_r bestimmt; die könnte noch ergänzt werden durch ihren Lagerbestand S_r, was hier aber nicht weiter betrachtet werden soll.

8.4.2 Eingabedaten

Hier stellen wir die relevanten Eingabegrößen zusammen:

B_r [L] die Breite der Bestandsrollen r.

$B_{\min}, B_{\max}$ $[mm]$ minimale und maximale Breite der Großbögen; $B_{\max} = \max_r\{B_r\}$

D_i [−] die Anzahl der Aufträge für die das Auftragsrechteck i.

I_i^{R} [−] Indikation, ob das Auftragsrechteck i rotiert werden darf ($I_i^{\mathrm{R}} = 2$) oder nicht ($I_i^{\mathrm{R}} = 1$).

$L_{\min}, L_{\max}$ [L] minimale und maximale Länge der Großbögen.

M [−] die Anzahl der zur Verfügung stehenden Messer.

W_i [L] die Breite des Auftragtyps i.

8.4.3 Variablen

Es werden die folgenden ganzzahligen Variablen verwendet:

$\mu_{rj} \in \mathbb{N}$ [−] gibt an, wie oft der Großbogen j von Rolle r geschnitten werden wird.
Diese Variable wird nur im Masterproblem (Partitionierungsproblem) verwendet.

$\alpha_{rij} \in \mathbb{N}$ [−] gibt an, wie oft das Rechteck $[A_i \times B_i]$ im Großbogen j von Rolle r auftritt.
Diese Variable, die im Unter-Problem verwendet wird, ist die einzige unabhängige Variable des Unterproblems und kann explizit ermittelt werden.

$\ell_{rj} \in \mathbb{N}$ [L] spezifiziert die Länge des Großbogens j von Rolle r.
Diese kontinuierlich Variable kann als Funktion aller α_{rij} berechnet werden.

$w_p \geq 0$ [−] gibt den Verschnitt von Muster p an.
Diese Variable kann je nach Auftragslage Werte zwischen 0 und B annehmen.

8.4.4 Überblick über die Algorithmischen Komponenten

Die Rechenschritte des Verfahrens stellen sich wie folgt dar:

1. Durch Länge und Breite bestimmte Auftragsrechtecke $[A_a, B_a]$ werden dupliziert und ergeben so orientierte Rechtecke $[A_i \times B_i]$. Dabei sind die Indizes i und a wie folgt miteinander verknüpft

$$a(i) := \begin{cases} i & , \quad 1 \leq i \leq N^{\mathrm{A}} \\ i - N^{\mathrm{A}} & , \quad N^{\mathrm{A}} + 1 \leq i \leq 2N^{\mathrm{A}} \end{cases}$$

und somit

$$[A_i \times B_i] := \begin{cases} [A_{a(i)} \times B_{a(i)}] & , \quad 1 \leq i \leq N^{\mathrm{A}} \\ [B_{a(i)} \times A_{a(i)}] & , \quad N^{\mathrm{A}} + 1 \leq i \leq 2N^{\mathrm{A}} \end{cases}$$

bzw.

$$[B_{i'} \times A_{i'}] = [A_i \times B_i] \quad , \quad i' := i + N^{\mathrm{A}} \quad , \quad \forall\{i | 1 \le i \le N^{\mathrm{A}}\} \quad .$$

Die Nachfrage für das Auftragsrechteck $[A_a, B_a]$ kann somit durch die orientierten Rechtecke i und i' gedeckt werden.

2. Für jede Rolle r mit Breite W_r werden durch vollständige Enumerierung

   (a) alle mit $\{A_i\}$ kompatiblen Streifenpartionierung (Großbögen) erzeugt (damit ist der Streifenverschnitt bekannt und die Breite der einzelnen Längsstreifen) und

   (b) das zugehörige Unterproblem (Minimierung des Vertikalverschnitts) gelöst (damit ist der Vertikalverschnitt und somit der Gesamtverschnitt des Bogens sowie seine Länge ℓ_{rj} bekannt und auch wieviele Auftragsrechtecke $[A_a, B_a]$ durch diesen Bogen abgedeckt werden.

3. Beim Masterproblem handelt es sich um ein Partitionierungsmodell. Dieses MILP-Problem berechnet, wieviele Großbögen der Länge ℓ_{rj} benötigt werden und die nachgefragte Menge von Auftragsrechtecke $[A_a, B_a]$ zu produzieren.

8.4.5 Master- und Unterproblem

8.4.5.1 Das Masterproblem: Partitionierungsmodell

Das Mastermodell minimiert den Gesamtverschnitt, d. h. das Modell umfasst eine geeignete Zielfunktion

$$\min \sum_r \sum_j W_{rj} \mu_{rj} \quad , \tag{8.4.39}$$

wobei W_{rj} den Verschnitt für Großbogen rj und die ganzzahlige Variable μ_{rj} die Bogenmultiplizität bezeichnet. Die Nachfragebedingungen lauten

$$D_a \le \sum_r \sum_j N^{\mathrm{tot}}_{raj} \mu_{rj} \le D_a + D^{\mathrm{over}}_a \quad , \quad \forall a \tag{8.4.40}$$

bzw.

$$D_i \le \sum_r \sum_j N^{\mathrm{tot}}_{rij} \mu_{rj} + \sum_r \sum_j N^{\mathrm{tot}}_{r,i+N^{\mathrm{A}},j} \mu_{rj} \le D_i + D^{\mathrm{over}}_i \quad , \quad \forall\{i | 1 \le i \le N^{\mathrm{A}}\} \quad . \tag{8.4.41}$$

Die Multiplizitätsvariablen μ_{rj} können durch

$$\mu_{rj} \le \max_a \left\{ \left\lceil \frac{D_a + D^{\mathrm{over}}_a}{\max\{1, N^{\mathrm{tot}}_{raj}\}} \right\rceil \right\} \quad , \quad \forall\{rj\}$$

beschränkt werden. Die Ganzzahligkeitsbedingung lauten

$$\mu_{rj} \in \{0, 1, 2, 3, \ldots\} \quad , \quad \forall\{rj\} \quad . \tag{8.4.42}$$

Aus operativen Gründen kann es sinnvoll sein, nur Rollen r_* einer bestimmten Breite B_{r_*} zu verwenden, da diese übereinanderliegend zu viert geschnitten werden können. Das durch (8.4.39) bis (8.4.42) definierte Modell wird dann ersetzt durch die Sequenz von N^R Modellen, bei denen jeweils nur eine der N^R Rollenbreite verwendet wird; es bleibt dem Benutzer, dann auszuwählen. Vorsicht ist geboten bei der Beschränkung der Überproduktion; die könnte bei Berücksichtigung einzelner Rollenbreiten leicht zu unzulässigen Lösungen führen. Denkbar wäre auch ein Goalprogrammingansatz (siehe Abschnitt 6.4), der zunächst die Überproduktion minimiert, dann den Verschnitt; oder auch umgekehrt.

Alternativ, könnte man auch eine Binärvariable δ_r einführen, die misst, ob die Rolle der Breite r verwendet wird, und diese Variablen der Bedingung

$$\sum_r \delta_r = 1$$

unterwerfen.

Die Lagerproblematik bzw. Berücksichtigung endlicher Bestandsmengen könnte durch den folgenden Ansatz abgebildet werden: Gegeben sei eine Menge von Bestandsrollen t mit den Informationen, von welchem Rollentyp r diese ist und welche Restlänge L_t von dieser Rolle noch verfügbar ist. Es müssten dann die Bedingungen

$$\mu_{rj} = \sum_{t|R_t=ord(r)} \mu_{rjt} \quad , \quad \forall\{rj\}$$

und

$$\sum_{t|R_t=ord(r)} L_{rj}\mu_{rjt} \leq L_t \quad , \quad \forall\{rjt\}$$

erfüllt sein, wobei die ganzzahlige Variable μ_{rjt} beschreibt, wieviele Großbögen vom Typ rj von Bestandsrolle t geschnitten werden sollen. Denkbar wäre, sich Kriterien zu überlegen, die die Bestandsrollen optimal nutzen.

8.4.5.2 Das Unterproblem

Bei gegebener Aufteilung der Längsschnitte für einen Großbogen rj, d. h. einem System von Werten A_i, $1 \leq i \leq 2N^{\mathrm{A}}$, besteht die Aufgabe darin, die Länge ℓ_{rj} des Großbogens so zu bestimmen, dass der vertikale Verschnitt w_{rj}^{ver}, der Gesamtverschnitt w_{rj}^{tot} bzw. der w_{rj}^{rel} Verschnitt minimal wird. Bezeichne die ganzzahlige Variable α_{rij}, wie oft das orientierte Rechteck $[A_i \times B_i]$ im durch A_i definierten Längsstreifen i mit Mustermultiplizität N_{rij}^{pat} enthalten ist, so ist der vertikale Verschnitt durch

$$w_{rj}^{\mathrm{ver}} := \sum_{i|N_{rij}^{\mathrm{pat}}>0} \left(N_{rij}^{\mathrm{pat}} A_i\right) s_{rij} \quad , \quad s_{rij} := \ell_{rj} - B_i\alpha_{rij}$$

definiert. Der absolute Verschnitt ist durch

$$w_{rj}^{\mathrm{tot}} := W_r\ell_{rj} + w_{rj}^{\mathrm{ver}} \quad .$$

gegeben. Betrachten wir den relativen Gesamtverschnitt

$$w_{rj}^{\mathrm{rel}} := W_r + w_{rj}^{\mathrm{ver}}/\ell_{rj} \quad ,$$

so führt dies auf ein MINLP-Problem. Der Vorteil in der Wahl des relativen Verschnitts als Zielkriteriums liegt darin, dass tendenziell eher große Großbögen nahe am Limit $L_{\max}$ generiert werden. Diese enthalten dann aber auch mehr Einzelformate.

Das Modell unterliegt den Schranken

$$L_{\min} \leq \ell_{rj} \leq L_{\max}$$

und

$$\alpha_{rij} = 0 \quad ; \quad \forall\{rij | N^{\text{pat}}_{rij} = 0\} \quad ,$$

$$\alpha^{\min}_{rij} := \left\lceil \frac{L_{\min}}{B_i} \right\rceil \leq \alpha_{rij} \leq \left\lfloor \frac{L_{\max}}{B_i} \right\rfloor =: \alpha^{\max}_{rij} \quad ; \quad \forall\{rij | N^{\text{pat}}_{rij} = 0\} \quad .$$

8.4.5.3 Explizite Lösung des Unterproblems

Aus Gründen der Recheneffizienz ist es sinnvoll, die Lösung des Unterproblems und alle abgeleiteten Daten (α_{rij}, ℓ_{rj}) simultan bei der vollständigen Enumerierung zu berechnen. Die Struktur des Unterproblems erlaubt es, dieses Problem explizit nach α_{rij} aufzulösen. Das Flussdiagram zur expliziten Berechnung von α_{rij} sieht für Großbogen rj wie folgt aus:

- Initialisierung des Verschnitts: $z_{\min} := +9999999$
- begin do: $i = 1, \ldots, N^{\text{W}} = 2N^{\text{A}}$

  1. if $N^{\text{pat}}_{ij} = 0$ goto 9000
  2. berechne $\alpha^{\min}_i := \left\lceil \frac{L_{\min}}{B_i} \right\rceil$ und $\alpha^{\max}_i := \left\lfloor \frac{L_{\max}}{B_i} \right\rfloor$
  3. begin do: $k = \max\{1, \alpha^{\min}_i\}, \ldots, \alpha^{\max}_i$; Schleife über alle zulässigen α_i
     (a) set $\alpha_i := k$, $\ell := B_i k$, $w^{\text{strip}} := W\ell$, $w^{\text{ver}} := 0$
     (b) begin do: $i\prime = 1, \ldots, 2N^{\text{A}}$; innere Schleifen über alle $i' \neq i$
        i. if $N^{\text{pat}}_{ri'j} = 0$ or $i\prime = i$ goto 8000
        ii. $\alpha_{i'} := \left\lfloor \frac{\ell}{B_{i'}} \right\rfloor$, $w^{\text{ver}}_{i'} := N^{\text{pat}}_{ij} A_i \, (\ell - B_{i'}\alpha_{i'})$, $w^{\text{ver}} := w^{\text{ver}} + w^{\text{ver}}_{i'}$
     (c) end do: $i\prime = 1, \ldots, 2N^{\text{A}}$; entry point 8000
     (d) berechne den absoluten Verschnitt

$$z := w^{\text{strip}} + w^{\text{ver}}$$

bzw. den relativen Verschnitt

$$z := \frac{w^{\text{strip}} + w^{\text{ver}}}{\ell}$$

     (e) prüfe, ob z den früheren Wert $z_{\min}$ verbessert

if $z < z_{\min}$ then
$\ell_* := \ell$
$w_*^{\text{strip}} := w^{\text{strip}}$
$w_*^{\text{ver}} := w^{\text{ver}}$
begin do: $i' = 1, \ldots, 2N^{\text{A}} | N_{i'j}^{\text{pat}} > 0$
$\alpha_{i'}^* := \alpha_{i'}$
$w_{i'}^{\text{ver}} := N_{ri'j}^{\text{pat}} A_{i'} \left(\ell_* - B_{i'} \left\lfloor \frac{\ell_*}{B_{i'}} \right\rfloor \right)$
end do
end if

4. end do: $k = \max\{1, \alpha_i^{\min}\}, \ldots, \alpha_i^{\max}$

- end do: $i = 1, \ldots, N^{\text{W}} = 2N^{\text{A}}$ (entry point: 9000)

Zu bemerken ist, dass dieses explizite Verfahren sowohl für absoluten als auch relativen Verschnitt funktioniert.

9 Nichtlineare Optimierung in der Praxis

In diesem Kapitel werden einige typische Probleme beschrieben, die auf nichtlineare, kontinuierliche Formulierungen führen. Hierzu zählen Probleme aus der Mineralölindustrie, insbesondere das *Pooling-Problem*, für das häufig spezielle Lösungstechniken (Rekursion und Sequentielle Lineare Programmierung) angewendet werden. Weiterhin wird gezeigt, wie man quadratische Optimierungsprobleme als gemischt-ganzzahlige Probleme löst.

9.1 Rekursion und sequentielle lineare Programmierung

Rekursion[1] [engl.: *recursion*] und das leicht modifizierte, besser unter der Bezeichnung *Sequentielle Lineare Programmierung* [engl.: *Successive Linear Programming* (SLP)] bekannte Verfahren sind Methoden, die mit Hilfe von LP-Algorithmen lokale Optima spezieller nichtlinearer Optimierungsprobleme berechnen, in denen nur nichtlineare Terme der Form $g(x)x$ auftreten. In der Mineralölindustrie oder Petrochemie repräsentiert $g(x)$ dabei z. B. typischerweise die Konzentration eines bestimmten Rohöls. Das Verfahren berechnet ausgehend von einem geschätzten Anfangswert für $g(x)$ die Lösung eines LP-Problems, daraus wiederum die neuen Werte für $g(x)$, damit wiederum die Lösung eines LP-Problems, usw. Liegt der zu Beginn geratene Wert nahe genug am Lösungswert, so konvergiert das Verfahren gegen ein lokales[2], aber nicht notwendigerweise globales Optimum. Praktikabel ist das Verfahren meist nur dann, wenn nur einige wenige dieser Koeffizienten $g(x)$ im Problem auftreten.

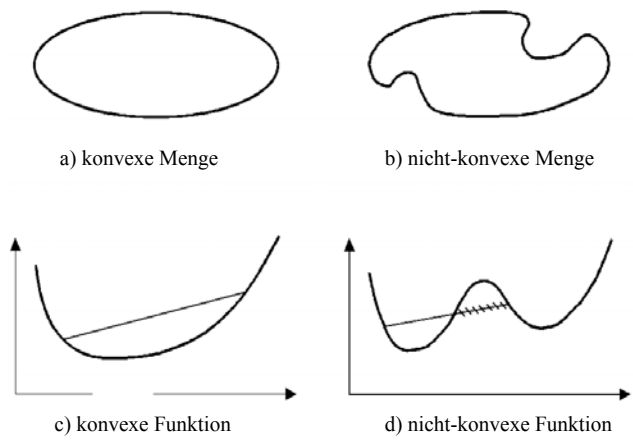
Im Abschnitt 9.1.1 beschreiben wir das Rekursionverfahren, in Abschnitt 9.1.2 erläutern wir anhand des Pooling-Problems die Sequentielle Lineare Programmierung und ein unter der Bezeichnung *Distributive Rekursion* bekanntes Verfahren.

[1] Der Begriff *Rekursion* hat natürlich in der Mathematik eine viel weitere Bedeutung. Besonders aus Optimierungsproblemen der Mineralölindustrie hat sich der Begriff der Rekursion bzw. im angelsächsischen *recursion* eingebürgert und meint manchmal auch nicht ganz zutreffend SLP.

[2] Ein zulässiger Punkt, manchmal auch zulässige Lösung genannt, eines Optimierungsproblems wird lokales Optimum genannt, wenn es in einer *nahen Umgebung* keinen Punkt gibt, der hinsichtlich der Zielfunktion besser ist. Die in Abbildung 9.1 gezeigte nichtkonvexe Funktion besitzt zwei lokale Minima; das linke davon ist das globale Minimum.

J. Kallrath, *Gemischt-ganzzahlige Optimierung: Modellierung in der Praxis*,
DOI 10.1007/978-3-658-00690-7_9, © Springer Fachmedien Wiesbaden 2013

Abbildung 9.1 Konvexe und nichtkonvexe Mengen. Konvexe und nichtkonvexe Funktionen. Bei konvexen Funktionen liegt die Tangente an einem bestimmten Punkten des Graphens dieser Funktion stets komplett unterhalb des Graphen; die Differenz zwischen dem Sekantengraph und dem Funktionsgraph für zwei beliebige Punkte des Graphen ist bei konvexen Funktionen stets positiv.



9.1.1 Rekursion

Zur Illustrierung eines Rekursionsverfahrens greifen wir das Beispiel im Abschnitt 2.3 mit $g(x) = \lambda_{cj}$ auf. Bei Kenntnis aller Strömungsraten können die relativen Anteile λ_{cj} mit Hilfe von (2.3.7) berechnet werden. Nimmt man für einen Moment an, dass die λ_{cj} bekannt sind und wie alle anderen konstanten Daten D in das Optimierungsproblem eingehen, so handelt es sich um ein LP-Problem. Nach Lösung dieses LP-Problems können die λ_{cj} mit (2.3.7) erneut berechnet werden. Daraus lässt sich die Iterationsschleife $\lambda_{cj}^{k+1} = \phi_*(\mathsf{D}) = \phi(\lambda_{cj}^k, \mathsf{D})$ konstruieren, für die im Idealfall die Konvergenzeigenschaft $\lim_{k\to\infty}(\lambda_{cj}^{k+1} - \lambda_{cj}^k) = 0$ zutrifft. Hierbei folgt $\phi(\lambda_{cj}^k, \mathsf{D})$ aus den Eingangsdaten λ_{cj}^k, den konstanten Daten D und der optimalen Lösung des LP-Problems. Sobald die Werte aller Variablen bekannt sind, erhält man mit $\phi(\lambda_{cj}^k, \mathsf{D})$ und (2.3.7) die neuen Werte λ_{cj}^{k+1}. Die Konvergenzeigenschaften dieses Verfahrens sind nur schwer zu beeinflussen; man kann nur hoffen, dass das Verfahren gegen ein lokales Optimum konvergiert.

Daher wird in der Praxis auch häufig ein leicht abgeändertes, unter der Bezeichnung *Sequentielle Lineare Programmierung* [engl.: *sequential linear programming, SLP*] bekanntes Verfahren verwendet. Dieses unterscheidet sich von der Rekursion nur dadurch, dass es die Konvergenz des Verfahrens garantiert, indem es die Koeffizienten durch Umformulierung etwas anders wählt und die Betragsdifferenz zwischen der angenommenen und der iterierten Lösung $g(x)$ beschränkt.

9.1.2 Das „Pooling"-Problem und Distributive Rekursion

Das Pooling-Problem tritt sehr häufig in der Petrochemie und in der Modellierung von Raffinerien sowie bei einigen mehrkomponentigen Netzwerkflussproblemen auf. Das gemeinsame Charakteristikum all dieser Probleme, die auf das Pooling-Problem führen ist, dass sowohl die Erhaltung des Gesamtflusses als auch der Zusammensetzung der einzelnen Ströme gefordert wird und somit Strömungsraten und Konzentrationen benötigt werden. Als Beispiel sei die Mischung mehrerer Ströme kontaminierten Wassers genannt, das in einem realen Sammeltank („pool") oder Knoten zusammengeführt wird und daraus wieder in mehreren Strömen abfließt. Das Pooling-Problem führt zu einem speziellen nichtlinearen Optimierungsproblem, das z. B. mit SLP und verwandten Techniken gelöst werden kann. Um das Problem genauer zu verstehen, sei ein „Pool" oder Tank mit einer flüssigen Substanz, z. B. Öl betrachtet. Mehrere Rohrleitungen führen Rohöl mit verschiedenen Strömungsraten und Konzentrationen eines bestimmten Anteils der Rohöle, für den wir uns hier interessieren, in diesen Tank. Aus diesem Tank fließen in mehreren Rohrleitungen Ströme verschiedener Strömungsraten aber gleicher Zusammensetzung ab. Diese Bedingung, dass die Zusammensetzung der aus einem „Pool" abfließenden Ströme identisch ist, wird auch *implizites pooling* genannt. Eine detaillierte Diskussion des Pooling-Problems findet sich bei Fieldhouse (1993,[83]) und Main (1993,[201]). Im Folgenden wird das Problem etwas allgemeiner für einen petrochemischen Anlagenverbund formuliert, in dem n Ströme mit unbekannten Strömungsraten x_i verschiedener Qualität[3], hier charakterisiert durch die Konzentration C_i aromatischer Komponenten, zusammengeführt werden. Im Tank ergibt sich eine mittlere Aromatenkonzentration c,

$$c = \sum_{i=1}^{n} C_i x_i / x \quad \text{bzw.} \quad cx = \sum_{i=1}^{n} C_i x_i \quad , \quad x = \sum_{i=1}^{n} x_i \tag{9.1.1}$$

mit der in den Tank eingeflossenen Ölmenge bzw. Flussrate x. Zu beachten ist, dass die Größen x, x_i und c unbekannte Variablen repräsentieren, während die Konzentrationen C_i hier zur Vereinfachung als bekannt und konstant angenommen sind (natürlich könnten dies auch Variablen sein). Wie in Abbildung 9.2(a) gezeigt, verlassen mehrere Ströme y_j den Tank mit gleicher Konzentration c; die jeweiligen absoluten Aromatenströmungsraten sind durch $\bar{y}_j := cy_j$ gegeben. Implizites Pooling führt somit auf die bilineare Bedingungen

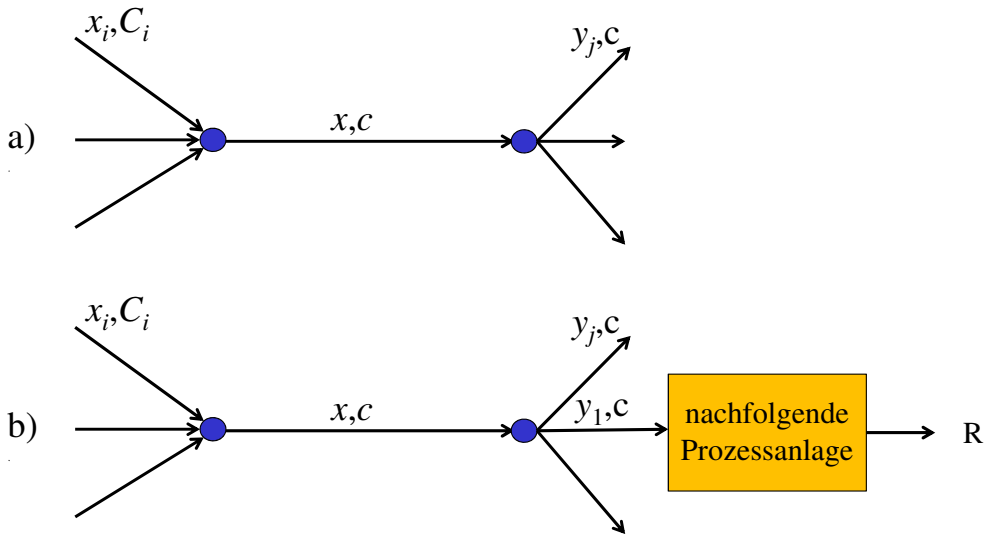
$$\frac{\bar{y}_{j_1}}{y_{j_1}} = \frac{\bar{y}_{j_2}}{y_{j_2}} \quad \Longleftrightarrow \quad \bar{y}_{j_1} y_{j_2} = \bar{y}_{j_2} y_{j_1} \quad . \tag{9.1.2}$$

Führt ein solcher Strom in eine nachfolgende Prozessanlage [Abbildung 9.2(b)], so könnte für den ersten Strom die in diese Anlage eintretende, erlaubte Absolutmenge $\bar{y}_1 := cy$ durch eine obere Schranke A^+ begrenzt sein. Damit ergibt sich die Qualitätsbedingung

$$cy_1 \leq A^+ \quad , \tag{9.1.3}$$

[3] In der Mineralölindustrie ist es üblich, die Ausdrücke *Qualität* und *Qualitätsrestriktionen* zu verwenden. Die Qualität kann dabei, z. B. im Falle von Konzentrationen einer Komponente in Strömungen, in dimensionslosen Einheiten, z. B. Tonnen/Tonne, oder, z. B. im Falle der Viskosität, in passenden physikalischen Einheiten gemessen werden.

Abbildung 9.2 Das Pooling-Problem.



wobei y_1 die aus dem Tank abgehende Strömungsrate und cy_1 die in die Anlage einfließende Aromatenströmungsrate bezeichnet. Die Gleichungen (9.1.1) und (9.1.2) sowie die Ungleichung (9.1.3) enthalten Produkte der Variablen. Diese bilinearen Terme sind unvermeidlich, wenn die Konzentration c explizit benötigt wird, wie in der Ungleichung (9.1.3), oder wenn implizites Pooling auftritt.
Nachfolgend wird das Problem nun in eine Form gebracht, wie sie in der SLP und auch *Distributiven Rekursion*[4] verwendet wird.

Hierzu werden die nichtlinearen Produktterme cx und cy_1 in (9.1.1) und (9.1.3) durch ihre Taylorreihenentwicklung ersetzt. Sind c^0 und x^0 die aktuellen, geschätzten oder anderweitig bekannten Werte für c und x, so lautet die Taylorreihenentwicklung erster Ordnung für cx

$$cx \cong c^0x^0 + x^0(c - c^0) + c^0(x - x^0) = c^0x + x^0\Delta c \tag{9.1.4}$$

bzw. für cy_1

$$cy_1 \cong c^0y_1^0 + y_1^0(c - c^0) + c^0(y_1 - y_1^0) = c^0y_1 + y_1^0\Delta c \quad . \tag{9.1.5}$$

Unter der Annahme, dass $\left|c - c^0\right|$ und $\left|x - x^0\right|$ „klein" sind, ist dies eine sinnvolle Vorgehensweise. Die rechte Seite der Gleichung (9.1.5) ist linear; in ihr treten x und die hier neu definierte Variable Δc,

$$\Delta c := c - c^0 \tag{9.1.6}$$

auf. Die Konzentration c im Tank, die eigentliche nichtlineare Variable, wird hier also ersetzt durch die die Änderung messende, im Vorzeichen unbeschränkte Variable Δc. Diese

[4] Im Prinzip lassen sich beide Verfahren auf Probleme mit sehr vielen Variablen anwenden, aber mit zunehmender Problemgröße steigt die Anforderung an die Genauigkeit der Anfangswerte.

folgt aus der Lösung eines LP-Problems und liefert die neue Konzentration $c = c^0 + \Delta c$, die in der nächsten Iteration anstelle von c^0 tritt. Die meisten SLP-Implementierungen in kommerziellen Softwarepaketen beschränken die erlaubten Änderungen durch

$$-S \leq \Delta c \leq S \quad ,$$

wobei die Schranke S so gewählt werden wird, dass eine hinreichende Genauigkeit der Taylorreihenentwicklung erwartet werden darf; SLP und DR verwenden verschiedene Heuristiken zur Steuerung der Schranke S. Verwendet man die in Zhang *et al.* (1985,[301]) entwickelte Methode, so ist die Konvergenz des SLP-Verfahrens gegen ein lokales Optimum für jede differenzierbare Nebenbedingung und Zielfunktion sichergestellt.

Die freie Variable Δc entspricht dem Koeffizienten $g(x)$ in dem im vorherigen Abschnitt beschriebenen Rekursionsverfahren. Im Verfahren der *Distributiven Rekursion* wird statt der Variablen Δc die Fehlervariable

$$e = x^0 \Delta c$$

verwendet; e hat dieselbe physikalische Dimension wie die Variablen x, also im vorliegenden Fall die Strömungsrate, und wird wie Δc auch als freie Variable behandelt. Damit nehmen die Nebenbedingungen des Poolings-Problems die Form

$$x = \sum_{i=1}^{n} x_i \quad ,$$

$$\sum_{i=1}^{n} C_i x_i = c^0 x + e$$

und

$$c^0 y_1 + \frac{y_1^0}{x^0} e \leq A^+$$

an. Dieses Verfahren wird *Distributive Rekursion* genannt, da es die Fehlervariable e mit dem Distributionsfaktor

$$\frac{y_1^0}{x^0} \leq 1 \tag{9.1.7}$$

verteilt. Es bietet erhebliche numerische Vorteile, weil das LP-Problem wesentlich bessere Skalierungseigenschaften aufweist; die Fehlervariable e tritt mit Koeffizient Eins oder einem zwischen Null und Eins liegenden Koeffizienten auf. Die nächste Iteration wird durch

$$c^0 \leftarrow c^0 + \frac{e}{x^0}$$

initialisiert. Ähnlich wie im SLP-Verfahren verwendet man hier die Dämpfungsheuristik

$$-E^- \leq e \leq E^+ \quad .$$

Das Pooling-Problem gehört wegen der auftretenden nichtlinearen Gleichungen zur Klasse der nichtlinearen nichtkonvexen Optimierungsprobleme. Daher ist auch hier zu erwarten, dass mehrere lokale Minima auftreten und wir mit dem hier vorgestellten lokalen Lösungsverfahren keine Aussage über das globale Optimum treffen können. Das Pooling-Problem besitzt in der Tat mehrere lokale Optima, die sich allesamt physikalisch sinnvoll interpretieren lassen. Je nach Wahl der Startwerte c^0 folgen verschiedene lokale Lösungen; die Erfahrung des Modellierers ist hier sehr wichtig. Bei der Wahl der Startwerte sollte beachtet werden, dass diese nicht mit dem Wert Null initialisiert werden, da dies in der Regel zu numerischen Problemen führt. Ist einmal eine gute lokale Lösung gefunden, so sollte diese gespeichert und zur Initialisierung weiterer Berechnung verwendet werden.

9.2 Quadratische Programmierung

Quadratische Programmierung[5] ist ein Spezialfall nichtlinearer Optimierung. In diesem Abschnitt soll lediglich eine nützliche Formulierung vorgestellt werden, die es erlaubt, mit einem MILP-Algorithmus gemischt-ganzzahlige oder kontinuierliche quadratische Optimierungsprobleme zu lösen; im konvexen Fall wird das globale Optimum, im nichtkonvexen Fall lediglich ein stationärer Punkt bestimmt, der einem lokalen Optimum oder einem Sattelpunkt entspricht. Für eine umfassendere Behandlung quadratischer Optimierungsprobleme verweisen wir den interessierten Leser z. B. auf Gill *et al.* (1981,[104]).

Betrachtet sei das quadratische Optimierungsproblem (QP)

$$\min \frac{1}{2}\mathbf{x}^{\mathrm{T}}\mathsf{G}\mathbf{x} + \mathbf{g}^{\mathrm{T}}\mathbf{x}$$

mit den Nebenbedingungen

$$\mathsf{A}^{\mathrm{T}}\mathbf{x} \geq \mathbf{b} \quad , \quad \mathbf{x} \geq \mathbf{0} \quad .$$

Führt man Lagrange-Multiplikatoren $\boldsymbol{\lambda}$ für die Nebenbedingung $\mathsf{A}^{\mathrm{T}}\mathbf{x} \geq \mathbf{b}$ und $\boldsymbol{\pi}$ für die Ungleichungen $\mathbf{x} \geq \mathbf{0}$ ein, so erhält man die Lagrange-Funktion

$$L(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\pi}) = \frac{1}{2}\mathbf{x}^{\mathrm{T}}\mathsf{G}\mathbf{x} + \mathbf{g}^{\mathrm{T}}\mathbf{x} - \boldsymbol{\lambda}^{\mathrm{T}}(\mathsf{A}^{\mathrm{T}}\mathbf{x} - \mathbf{b}) - \boldsymbol{\pi}^{\mathrm{T}}\mathbf{x} \quad .$$

Mit den Schlupfvariablen $\mathbf{r} := \mathsf{A}^{\mathrm{T}}\mathbf{x} - \mathbf{b}$ ergeben sich die Kuhn-Tucker-Bedingungen

$$\boldsymbol{\pi} - \mathsf{G}\mathbf{x} + \mathsf{A}\boldsymbol{\lambda} = \mathbf{g} \quad , \tag{9.2.1}$$

$$\mathbf{r} - \mathsf{A}^{\mathrm{T}}\mathbf{x} = -\mathbf{b}$$

und

$$\boldsymbol{\pi},\, \mathbf{r},\, \mathbf{x},\, \boldsymbol{\lambda} \geq \mathbf{0} \quad , \quad \boldsymbol{\pi}^{\mathrm{T}}\mathbf{x} = 0 \quad , \quad \mathbf{r}^{\mathrm{T}}\boldsymbol{\lambda} = \mathbf{0} \quad ,$$

wobei die beiden letzten Gleichungen als Komplementaritätsbedingungen bezeichnet werden. Sie besagen, dass für jede Variable $\mathbf{x}$, diese selbst oder ihr assoziierter Lagrange-Multiplikator Null ist bzw. beide gleichzeitig den Wert Null annehmen; entsprechendes gilt für $\mathbf{r}$ und das assoziierte $\boldsymbol{\lambda}$. Jede Lösung von QP muss notwendigerweise die Kuhn-Tucker-Bedingungen erfüllen. Ist G positiv definit, so ist das QP Problem konvex und jede Lösung, die die notwendigen Bedingungen erfüllt, ist optimal. Ist G nicht positiv definit, so ist lediglich ein lokales Optimum bestimmt. Enthält das Problem auch ganzzahlige Variablen (MIQP), so werden die Verhältnisse komplizierter.

Sind die Variablen $\mathbf{x}$ nach oben beschränkt, so lässt sich dieser Ansatz entsprechend erweitern. Beispielsweise kann die Ungleichung $\mathbf{x} \leq \mathbf{U}$ als

$$-\mathbf{x} \geq -\mathbf{U}$$

geschrieben werden, was die Einführung der Schlupfvariablen $\mathbf{s} := \mathbf{U} - \mathbf{x}$ und einem entsprechenden Lagrange-Multiplikator $\boldsymbol{\sigma}$ für diese Ungleichungen nahelegt. Damit ergeben sich die Komplementaritätsbedingungen

$$\boldsymbol{\sigma}^{\mathrm{T}}\mathbf{s} = 0$$

[5] Unter quadratischen Optimierungsproblemen werden in der Regel Probleme mit kontinuierlichen Variablen, quadratischer Zielfunktion und linearen Nebenbedingungen verstanden.

und (9.2.1) wird erweitert zu

$$\boldsymbol{\pi} - \boldsymbol{\sigma} - \mathsf{G}\mathbf{x} + \mathsf{A}\boldsymbol{\lambda} = \mathbf{g} \quad .$$

Zu beachten ist, dass Gleichungsbedingungen zu Lagrange-Multiplikatoren $\boldsymbol{\mu}$ ohne Vorzeichenbeschränkungen, also freien Variablen, führen würden.

Die Kuhn-Tucker-Bedingungen bilden ein System nichtlinearer Gleichungen mit einer sehr speziellen Gestalt; lediglich die Komplementaritätsbedingungen $\boldsymbol{\pi}^{\mathrm{T}}\mathbf{x} = 0$ und $\mathbf{r}^{\mathrm{T}}\boldsymbol{\lambda} = 0$ sind nichtlinear und insbesondere bilinear. Das Skalarprodukt zweier nichtnegativer Vektoren kann aber nur dann den Wert Null annehmen, wenn die Komponentenprodukte $\pi_i x_i$ verschwinden, wenn höchstens eine der beiden Variablen von Null verschieden ist bzw. anders herum formuliert: Es muss jeweils π_i oder x_i verschwinden, oder beide. Damit lassen sich die Komplementaritätsbedingungen mit Hilfe der in Abschnitt 5.7.1 eingeführten SOS-1-Mengen formulieren, wobei jedes x_i und sein assoziierter Lagrange-Multiplikator π_i eine solche Menge bilden. Die Grundidee besteht also darin, quadratische Probleme mit einem MILP-Verfahren zu lösen, in dem die Komplementaritätsbedingungen durch SOS-1-Mengen formuliert werden. Alternativ zur Verwendung von SOS-1-Mengen können die Bedingung $\pi_i x_i = 0$ auch mit Hilfe von Binärvariablen zu formuliert werden. Sind $\prod$ und X obere Schranken für π_i und x_i, so ergibt sich mit

$$x_i \leq X\delta \quad , \quad \pi_i \leq \prod(1-\delta) \tag{9.2.2}$$

und $\delta = 0$ oder $\delta = 1$, dass π_i oder x_i den Wert 0 annehmen. Die Ungleichungen (9.2.2) schließen nicht den Fall aus, dass beide Variablen den Wert Null annehmen.

Zu beachten ist, dass das aus den Kuhn-Tucker-Bedingungen resultierende Problem keine Zielfunktion mehr besitzt; es kann für den MILP-Algorithmus aber jede beliebige Zielfunktion verwendet werden.

Ein Beispiel soll nun die Funktionsweise der Methode verdeutlichen:

$$\min \quad 3x^2 + 2y^2 + z^2 + xy + \frac{1}{2}zx + xy - \frac{2}{5}zy + \frac{1}{2}xz - \frac{2}{5}yz$$

mit den Nebenbedingungen

$$\begin{array}{rcl} x + y + z & \geq & 1.00 \\ 1.3x + 1.2y + 1.08z & \geq & 1.12 \end{array} \quad , \quad \begin{array}{l} 0 \leq x \leq 0.75 \\ 0 \leq y \leq 0.75 \\ 0 \leq z \leq 0.75 \end{array} \quad .$$

In die allgemeine Schreibweise eines QP-Problems überführt, d. h.

$$\mathbf{x}^{\mathrm{T}}\mathsf{G}\mathbf{x} = \sum_i \sum_j G_{ij} x_i x_j \quad ,$$

ergibt sich

$$\mathsf{G} := \begin{pmatrix} 6.0 & 2.0 & 1.0 \\ 2.0 & 4.0 & -0.8 \\ 1.0 & -0.8 & 2.0 \end{pmatrix} \quad , \quad \mathbf{g} := \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

und

$$\mathsf{A}^{\mathrm{T}} := \begin{pmatrix} 1.0 & 1.0 & 1.00 \\ 1.3 & 1.2 & 1.08 \end{pmatrix} \quad , \quad \mathbf{b} := \begin{pmatrix} 1.00 \\ 1.12 \end{pmatrix} \quad , \quad \mathbf{U} := \begin{pmatrix} 0.75 \\ 0.75 \\ 0.75 \end{pmatrix} \quad .$$

Mit SOS-1-Mengen, wie oben beschrieben, lässt sich das komplette Beispiel z. B. in der Modellierungssprache `Xpress-MP` wie folgt kodieren:

```
! To minimise 1/2x'Gx + g'x      (' indicates transpose)
!  subject to
!    A'x >= b , x>= 0 , x<= U    [ A is m*n , G is n*n ]

! Lagrangian L(x,lam,sigma,pi) =
!  1/2x'Gx +g'x -lam'(A'x-b)-sigma'(U-x)-pi'x

! Kuhn-Tucker conditions imply
!  pi-sigma-Gx+A*lam = g
!      i.e.:    pi + A*lam = g + sigma + Gx
!  A'x-r   = b
!  x+slack = U
!
!  pi'*x=0; slack'*sigma=0; r'*lam=0 (complementary slackness)
!  lam is free if A'x=b, and the corresponding r doesn't exist
!  sigma and slack do not exist if U(i) infinite

LET N=3                 ! Dimension of x (three variables)
LET M=2                 ! Number of linear constraints
LET HUGE = 1.0e10       ! Upper bounds > HUGE are assumed not to exist

TABLES
 G   (N,N)
 B   (M)
 AT  (M,N)
 g   (N)
 U   (N)
 IFEQ(M)                 ! 1 if row i is equality, else 0

DATA                     ! entries not specified are zero
 G(1,1)  = 6 ,  2   ,  1
 G(2,1)  = 2 ,  4   , -0.8
 G(3,1)  = 1 , -0.8 ,  2
 B(1)    = 1 , 1.12
 AT(1,1) = 1   ,  1   , 1
 AT(2,1) = 1.3 ,  1.2 , 1.08
 g(1)    = 0    , 0   , 0
 U(1)    = 0.75 , 0.75, 0.75
 IFEQ(1) = 0 ,  0

VARIABLES
 x  (N)
 pi (N)
 lam(M)
 r(i=1:M | IFEQ(i).ne.1)
 slack(i=1:N | U(i) < HUGE)
 sigma(i=1:N | U(i) < HUGE)
```

```
CONSTRAINTS

! Reference row
 dummy:                                          &
  -SUM(i=1:N) x (i)                              &
  +SUM(i=1:N) pi(i)                              &
  -SUM(i=1:M | IFEQ(i).eq.0) r(i)                &
  +SUM(i=1:M | IFEQ(i).eq.0) lam(i)              &
  -SUM(i=1:N | U(i) < HUGE ) slack(i)            &
  +SUM(i=1:N | U(i) < HUGE ) sigma(i)      $

 PI(i=1:N):  pi(i) + SUM(j=1:M) AT(j,i) * lam(j)          &
           = g(i) + sigma(i) + SUM(j=1:N) G(i,j) * x(j)
 R(i=1:M):  -r(i) + SUM(j=1:N) AT(i,j) * x(j)  = B(i)
 UPx(i=1:N | U(i) < HUGE):     x(i) + slack(i) = U(i)

BOUNDS
 lam(i=1:M | IFEQ(i).eq.1) .FR. ! Free for = constraints

SETS      ! For complementary slackness
 SSpi(i=1:N): pi(i) + x(i)                       .S1. dummy
 SSl(i=1:M | IFEQ(i).eq.0):     r(i) + lam(i)    .S1. dummy
 SSs(i=1:N | U(i) < HUGE) : slack(i) + sigma(i) .S1. dummy

END
```

Löst man dieses Problem, das in Kallrath & Wilson (1997, 123ff.) unter der Bezeichnung *quadrat* geführt wird, mit `Xpress-MP`, so erhält man das Ergebnis:

$$\begin{array}{rcl} x &=& 0 \\ y &=& 0.368421 \\ z &=& 0.631579 \end{array} \quad , \quad \begin{array}{l} \pi_1 = 0.4 \\ \pi_2 = 0.0 \\ \pi_3 = 0.0 \end{array} \quad , \quad \begin{array}{l} \lambda_1 = 0.968421 \\ \lambda_2 = 0 \end{array} \quad , \quad \begin{array}{l} \sigma_1 = 0 \\ \sigma_2 = 0 \\ \sigma_3 = 0 \end{array} \quad .$$

Diese Lösung ist auch die optimale Lösung des Problems, da die Matrix G die positiven Eigenwerte

$$\phi_1 = 1.097\,0 \quad , \quad \phi_2 = 3.629\,6 \quad , \quad \phi_3 = 7.273\,4$$

hat und somit positiv definit ist. Damit ist das Optimierungsproblem aber konvex; folglich ist gemäß Theorem 6 im Anhang 4.4.2 jede lokale Lösung auch globale Lösung.

In praktischen Problemen kann es sehr schwierig sein, den Nachweis zu führen, dass G positiv definit ist. Dennoch ist es den Versuch wert, dies zu untersuchen. Manchmal liegt ein besonderer Spezialfall vor, der in Abschnitt 5.6.2 beschrieben wurde, und mit dem hier beschriebenen Verfahren ideal gelöst werden kann.

10 Gemischt-ganzzahlige nichtlineare Optimierung in der Praxis

Gemischt-ganzzahlige nichtlineare Optimierung wird zunehmend in der Verfahrenstechnik der chemischen Industrie, in der Produktionsplanung und bei Schedulingproblemen in Raffinerien, beim Design von Stromkreisen oder auch bei der Konstruktion von Anlageinstrumenten in der Finanzwelt angewendet. Winter & Zimmermann (2000,[294]) formulieren eine Fragestellung aus dem Verkehrswesen, ein Strassenbahnabfertigungs- und Rangierproblem, als quadratisches Zuordnungsproblem und lösen dieses mit Hilfe exakter Methoden (dynamische Optimierung) und Heuristiken.

In diesem Kapitel werden zwei umfangreiche Probleme behandelt, die sich mit Hilfe gemischt-ganzzahliger nichtlinearer Modelle beschreiben lassen. Bei dem ersteren handelt es sich um ein Netzwerkdesignproblem, beim zweiten um die Auslegung eines Prozesses. Beide Probleme und ihre Modellierung erschienen in einer vorläufigen Form bereits in Kallrath (1999,[154]); daher werden hier einige Aspekte verkürzt dargestellt bzw. andere Punkte ergänzt.

In diesem Kapitel werden Beispiele für gemischt-ganzzahlige nichtlineare Optimierung vorgestellt, bei denen gute zulässige Lösungen gewonnen wurden, von denen aber letztlich der Nachweis der globalen Optimalität noch aussteht. Dabei lässt sich feststellen, dass diese Probleme recht aufwendig in der Modellierung sein können, spezielle Heuristiken zur Gewinnung guter Startwerte – Homotopie-Verfahren, auf Modellsequenzen angewendet, können ein nützliches Mittel sein – und einen sorgfältigen Umgang mit den vorhandenen Lösungsalgorithmen erfordern und jedes MINLP-Problem seinen speziellen Lösungszugang benötigt.

10.1 Eine Integrierte Standortanalyse

In dieser Designstudie sollen die Kosten für ein standortweites Produktionsnetzwerk minimiert werden. Diese umfassen die Kosten für die Rohprodukte, Investitionskosten und variable Kosten für Wiederaufbereitungsanlagen sowie einen Strafkostenterm für Restverunreinigungen, die an einem bestimmten Punkt des Produktionsnetzwerkes anfallen.

Es werden drei Anlagen- bzw. Prozesstypen betrachtet: solche, die den Rohstoff komplett verbrauchen (Senkprozesse), solche, die keinen brauchen, aber ihn in einer bestimmten Reinheit produzieren (Quellprozesse) und solche, die ihn brauchen und produzieren (Dualprozesse); die Flussraten eines Prozesses bzw. einer Anlage sowie die Qualität des Rohstoffes im Ausgangsstrom sind *a priori* bekannt. Die Rohprodukte stehen in drei verschiedenen Qualitäten nur begrenzt zur Verfügung. Da zudem die Kosten hoch sind, ist man bemüht, die im System entstehenden Rohprodukte weiter- bzw. wiederzuverwenden.

J. Kallrath, *Gemischt-ganzzahlige Optimierung: Modellierung in der Praxis*,
DOI 10.1007/978-3-658-00690-7_10, © Springer Fachmedien Wiesbaden 2013

Dies erfordert zum einen neue Verbindungen zwischen den Anlagen bzw. eine veränderte Flusstopologie mit neuen Sammelpunkten [engl.: *pools*], zum anderen, wegen der geforderten Eingangsqualität an den Rohstoff, mögliche lokale Wiederaufbereitungsanlagen (WAA). Investitionskosten steigen mit der Kapazität und Länge der Verbindungsleitungen; im Falle einer WAA sind neben den kapazitätsabhängigen auch variable Kosten zu berücksichtigen, neue Pools bleiben kostenfrei. Die von einer WAA extrahierten Komponenten eines Stroms können verkauft werden. Schließlich existiert noch eine zentrale WAA; falls bestimmte Komponenten sie passieren, werden dafür Strafkosten erhoben.

10.1.1 Das Mathematische Modell

Zunächst fassen wir die auftretenden Objekte und Indizes des Modells zusammen:

	#	*Beschreibung*
	3	Rohproduktqualitäten
N	~ 60	Quellprozesse
M	~ 7	Verunreinigungen
L	≥ 1	WAA
P	~ 60	Pools

i	Quellen
k	Verunreinigungen
m	Wiederaufbereitungsanlage (WAA)
p	Pools
q	Verbindungskapazität
s	Senken

Insgesamt werden also $3 + N + L + P \sim 125$ Prozesse betrachtet, hierin sind auch WAA als Prozesse enthalten, und durch die folgenden Daten charakterisiert:

Daten	Dim.	Beschreibung
K_{ik}^{out}	$[ppm]$	spezifische Ausgangskonzentration von Verunreinigung k des Prozesses P_i
K_{jk}^{in}	$[ppm]$	Eingangsspezifikation für Verunreinigung k für Prozess P_j
X_i	$[t/h]$	Eingangsstrom des Prozesses P_i, $X_i = 0$ für Quellprozesse P_i
Y_i	$[t/h]$	Ausgangsstrom des Prozesses P_i, $Y_i = X_i$ für alle Dualprozesse und $Y_i = 0$ für Senkprozesse; für lokale WAA gilt $Y_i \leq X_i$, da der Eingangsstrom in einen Hauptstrom A und einen kleineren Nebenstrom B niedrigerer Qualität geteilt wird
$F_{mk}(c_{mk}^{in})$	$[-]$	Extraktionsrate der Verunreinigung k durch die Anlage WAA P_m. Diese Rate ist eine Funktion der Eingangskonzentration c_{mk}^{in} und ist von der Grössenordnung 0.7....0.95.
G_m	$[-]$	Aufteilungsrate des Einganggstrom in der Anlage WAA m.
C_{ij}^{PI}		Investitionskosten für den Pipelinebau von P_i nach P_j
C_m^{TI}		Investitionskosten für eine lokale WAA P_m. Diese Kosten sind eine Funktion der Eingangs-Strömungsrate sobald klar ist, welche Verunreinigung extrahiert werden sollen, d. h. $C_m^{TI} = C_m^{TI}(x_{im}, z_{mk}^{in})$.
$C_{mk}^{TV}(z_{mk}^{in})$		variable Kosten für WAA P_m. Diese sind eine Funktion des Produktes der Extraktionsrate und der Massenbeladung an Verunreinigung k, d. h. $f_{mk}(c_{mk}^{in})z_{mk}$
C_i^{RM}		Einkaufspreis für den Kauf von Rohprodukt in der Qualitätsstufe i
C_k^{PEN}		Strafkosten für den Systemabfluss der Verunreinigung k
V_{ijq}^{PI}	$[t/h]$	Kapazität der Pipeline von i nach j ist vom Typ (Grösse) q

Zur Vereinfachung werden die folgenden Indexmengen eingeführt:

$\mathcal{K}^P$	die Menge (i,j) aller zulässigen Verbindungen
$\mathcal{K}^N$	die Menge (i,j) nicht existierender Verbindungen ($\mathcal{K}^N \subset \mathcal{K}^P$)
$\mathcal{P}^{SO}$	$i \in \mathcal{P}^{SO} \Leftrightarrow P_i$ ist ein Quellprozess [engl.: *source process*]
$\mathcal{P}^{SI}$	$j \in \mathcal{P}^{SI} \Leftrightarrow P_j$ ist ein Senkprozess [engl.: *sink process*]
$\mathcal{P}^{ST}$	$i \in \mathcal{P}^{ST} \Leftrightarrow P_i$ ist ein Dualprozess [engl.: *stream process*]
$\mathcal{P}^T$	$m \in \mathcal{P}^T \Leftrightarrow P_m$ ist eine Wiederaufbereitungsanlage (WAA)
$\mathcal{P}^P$	$m \in \mathcal{P}^P \Leftrightarrow P_p$ ist ein Pool
$\mathcal{P}_1$	$\mathcal{P}_1 := \mathcal{P}^{SO} \cup \mathcal{P}^{ST} \cup \mathcal{P}^P \cup \mathcal{P}^T$
$\mathcal{P}_2$	$\mathcal{P}_1 := \mathcal{P}^{SI} \cup \mathcal{P}^{ST} \cup \mathcal{P}^P \cup \mathcal{P}^T$

10.1.1.1 Variablen

Zunächst werden die folgenden kontinuierlichen nichtnegativen Variablen benötigt

kontinuierl. Variablen	Dim.	Beschreibung
$x_{ij}^A \equiv x_{ij}$	t/h	Ausgangsstrom von Prozess P_i zu Prozess P_j (bei Reinigungsanlagen WAA: Ausgangsstrom (A), hohe Qualität)
$x_m^B = x_{im}^A\,(1 - G_m)$	t/h	der Ausgangsstrom (B) niedriger Qualität von Prozess P_i; für Prozesse, die nicht lokale Reinigungsanlagen sind, gilt $x_i^B = 0$
c_{ik}^{in} (c_{ik}^{out})	ppm	Eingangskonzentration (Ausgangskonzentration) der Verunreinigung k in Prozess P_i
$z_{jk}^{in} := \sum_{i,(i,j)\in K^P} c_{ik}^{out} x_{ij}$	t/h	Eingangsmassenstrom von Verunreinigung k in Prozess P_j summiert über alle Prozesse P_i, die das Rohprodukt in diese einströmen lassen.

Darüber hinaus werden die binären Variablen μ_{ij}, ε_{ijq} und ν_m eingeführt, die indizieren, ob eine Verbindung von P_i nach P_j existiert, ob eine Verbindung P_i nach P_j die Kapazität C_{ijq} besitzt und ob der Prozess P_m für eine WAA existiert.

10.1.1.2 Nebenbedingungen

Die Produktionsprozesse (Senken und Quellen) benötigen eine konstante Menge des Rohmaterials und produzieren eine konstante Menge dieses Materials mit bestimmten Verunreinigungen. Für die Gesamtströme in jedem Knotenpunkt gelten daher die *Massenbilanzen*

$$\sum_{i|(i,j)\in K^P} x_{ij} = X_j \quad , \quad \forall j \in \left\{\mathcal{P}^{SI} \cup \mathcal{P}^{ST}\right\}$$

und

$$\sum_{j|(i,j)\in K^P} x_{ij} = Y_i \quad , \quad \forall i \in \left\{\mathcal{P}^{SO} \cup \mathcal{P}^{ST}\right\} \quad ,$$

sowie für die Massenerhaltung für Dual- und Pool-Prozesse

$$X_i = Y_i \quad , \quad \forall i \in \left\{\mathcal{P}^P \cup \mathcal{P}^{ST}\right\} \quad .$$

In WAA wird der Eingangsstrom in einen Hauptstrom (A) relativ hoher und einen Nebenstrom (B) geringerer Qualität geteilt. Da auch hier Massenerhaltung gefordert wird, kann der Fluss in einer WAA P_m ähnlich wie ein Dualprozess modelliert werden, d. h.

$$\sum_{i|(i,m)\in K^P} x_{im} = \sum_{j'|(m,j')\in K^P} x_{m,j'} + x_m^B \quad , \quad \begin{array}{l} \forall i \in \left\{\mathcal{P}^{SO} \cup \mathcal{P}^{ST} \cup \mathcal{P}^P\right\} \\ \forall j \in \left\{\mathcal{P}^{SI} \cup \mathcal{P}^{ST} \cup \mathcal{P}^P\right\} \\ \forall m \in \mathcal{P}^T \end{array} \quad .$$

Die Teilung wird durch einen Faktor G_m beschrieben, der das Verhältniss der Strömungsrate in Strom (B) zur gesamt einfließenden Strömungsrate beschreibt ($G_m = 0$ impliziert also, dass sämtliches Rohmaterial im Hauptstrom verbleibt):

$$G_m = \frac{x_m^B}{\sum_{i|(i,m)\in K^P} x_{im}} \quad \Leftrightarrow \quad x_m^B = \sum_{i|(i,m)\in K^P} G_m x_{im} \quad , \quad \forall m \in \mathcal{P}^T \quad .$$

Der Pool P_j, der verschiedene Eingangsströme P_i vereinigt, wird durch

$$c_{jk}^{in} X_j = \sum_{i|(i,j)\in K^P} c_{ik}^{out} x_{ij} \quad , \quad \begin{array}{l} \forall i \in \mathcal{P}_1 \\ \forall k, \forall j \in \mathcal{P}^P \end{array} \quad ,$$

also die aus dem *Pooling*-Problem bekannten bilinearen Gleichungen beschrieben. Die Konzentrationsgrenzen für bestimmte Verunreinigungen aus den Prozessen P_i, die in den Prozess P_j einfließen, ergeben hier die Ungleichungen

$$\sum_{i|(i,j)\in K^P} c_{ik}^{out} x_{ij} =: z_{jk}^{in} \le K_{jk}^{in} X_j \quad , \quad \forall j \in \mathcal{P}_2 \quad , \quad \forall k \quad .$$

In Abhängigkeit von der Konzentration der Verunreinigung k und des Stromes aus Prozess P_i, der in Prozess P_j eintritt, kann die Konzentration c_{jk}^{in} im Pool, d. h. die Eingangskonzentration für Prozess P_j, berechnet werden

$$\begin{aligned} c_{jk}^{in} \sum_{i|(i,j)\in K^P} x_{ij} \quad &= \quad \sum_{i|(i,j)\in K^P} c_{ik}^{out} x_{ij} \\ \Leftrightarrow \quad & c_{jk}^{in} X_j = z_{jk}^{in} \quad , \quad \forall j \in \mathcal{P}_2, \quad \forall k \quad . \end{aligned}$$

Die Ausgangskonzentration eines Stromes besteht aus der Summe der bereits im Eingangsstrom vorhandenen Konzentration sowie des zusätzlich durch den Prozess verursachten Anteils, d. h.

$$c_{jk}^{out} = c_{jk}^{in} + K_{ik}^{out} \quad , \quad \forall j \in \left\{\mathcal{P}^{SO} \cup \mathcal{P}^{ST}\right\} \quad , \quad \forall k \quad .$$

Die Ausgangsbelastung eines Dualprozesses (dies gilt nicht für WAA) ist somit

$$z_{ik}^{out} X_i = z_{ik}^{in} + K_{ik}^{out} X_i \quad , \quad \forall i \in \mathcal{P}^{ST} \quad , \quad \forall k \quad .$$

Die Ausgangskonzentration für die Verunreinigung k in einer WAA P_j hängt von der Eingangskonzentration und der Extraktionsrate der WAA bezüglich der Komponente k ab. Allgemein kann dies durch die nichtlineare Funktion $z_{jk}^{out} = H(z_{jk}^{in})$ beschrieben werden. Nachstehend sind zwei typische Beispiele gezeigt, die die Wirkungsweise einer WAA verdeutlichen:

$$\begin{array}{lcl} z_{jk}^{out} & = & z_{jk}^{in}\left(1-F_{jk}\left(z_{jk}^{in}\right)\right) \\ & \text{oder} & \\ z_{j1}^{out} & = & z_{j1}^{in}-\sum_{k=2}^{6} z_{jk}^{in} F_{jk}(z_{jk}^{in}) \end{array} \qquad \begin{array}{lcl} z_{jk_1}^{out} & = & z_{jk_*}^{in}-z_{j,k_2}^{in} \\ z_{j,k_2}^{out} & = & 0 \\ z_{j,k_2}^{out} & = & z_{j,k_3}^{in} \end{array} \quad .$$

Die Konstruktion von Verbindungen von i nach j wird durch die Ungleichungen

$$x_{ij} \leq \min_{ij}\{Y_i, X_j\}\,\mu_{ij} \quad , \quad x_{ij} \leq \sum_q V_{ijq}^{PI}\varepsilon_{ijq} \quad , \quad \forall (i,j) \in \mathcal{K}^N$$

beschrieben, wobei die zweite Ungleichung die benötigte Kapazität der Verbindung auswählt. Die Gleichung

$$\mu_{ij} = \sum_{q=1}^{Q} \varepsilon_{ijq} \quad , \quad \forall i \in \mathcal{P}_1 \quad , \quad \forall j \in \mathcal{P}_2$$

stellt sicher, dass nur eine Kapazität gewählt wird. Hierbei bietet es sich an, während der Variablenwahl die Entscheidung „*eine Verbindung existiert* ($\mu_{ij} = 1$) bzw. *es existiert keine Verbindung* ($\mu_{ij} = 0$)" durch entsprechende Prioritäten auf die Binärvariablen μ_{ij} bevorzugt zu treffen.

Schließlich stellen die Ungleichungen

$$\sum_{i|(i,m)\in K^P} x_{im} \leq \left(\sum_{j'|(m,j')\in K^P} x_{m,j'} + x_m^B\right)\nu_m \quad , \quad \begin{array}{l} \forall i \in \mathcal{P}^{SO} \setminus \mathcal{P}^T \\ \forall m \in \mathcal{P}^T \\ \forall t \end{array} \quad ,$$

sicher, dass der Bau einer WAA P_m korrekt durch die Binärvariable ν_m erfasst wird.

Damit enthält das Modell die folgenden Klassen von Nebenbedingungen:

	LP	NLP	IP
Massenerhaltung (Gesamtstrom)	×		
Massenerhaltung (Verunreinigung)		×	
Pooling		×	
Rückführung [engl.: *re-cycle*]		×	
semi-kontinuierliche Strömungsraten			×
Existenz von Verbindungen			×
Wahl der Verbindungskapazität			×
Wiederaufbereitungsanlagen		×	×

.

10.1.1.3 Zielfunktion

Die zu minimierende Zielfunktion besteht aus der Summe aller Investitions- und Strafkosten, der Summe der variablen Kosten, sowie einem Erlösterm aus dem Verkauf extrahierter Komponenten, d. h.

$$\begin{aligned} Z \quad : \quad & = \min \sum_{i,j|(i,j)\in K^P} C_{ij}^{PI}\mu_{ij} \\ & + \sum_{i,j|(i,j)\in K^P} \left\{ C_j^{TI}(x_{ij}, z_{jk}^{in})\nu_j + \sum_k C_j^{TV} z_{jk}^{in} \right\} \\ & + C_{ij}^{RM} x_{ij} - \sum_{j|(i,j)\in K^P} \sum_k S_k\left(z_{jk}^{in} - z_{jk}^{out}\right) + \sum_k C_k^{PEN} z_{ctr,k}^{out} \quad . \end{aligned} \tag{10.1.1}$$

Der erste Summand repräsentiert die Investitionskosten für den Bau einer Verbindung von Prozess i nach j, der zweite die variablen Kosten und die Investitionskosten für den Bau der WAA j, die die Ströme aus den Prozessen i zusammenfasst, der dritte Term gibt schließlich die Kosten für den Strom von Prozess i nach j, $i \in \{1, 2, 3\}$, also die Kosten für den Kauf des Rohmaterials. Der vorletzte Term stellt den Erlös für den Verkauf der Komponente k, die aus den Prozessen i stammt und im Prozess j extrahiert wird; der letzte Term in (10.1.1) erfasst die Strafkosten für die Komponente k, falls diese das System durch die zentrale WAA P_{ctr} verlässt.

10.1.2 Lösungsweg „Homotopie-Verfahren" und Ergebnisse

Das Problem wird in einer Sequenz aus mit Hilfe der Modellierungssprache `GAMS` formulierten Untermodellen verschiedener Komplexität gelöst. In diesem Homotopieverfahren wird der Lösungsvorgang eines Modells $m + 1$ mit den Ergebnissen aus Modell m initialisiert. Ein einfaches lineares Modell liefert so z. B. erste Ergebnisse, um die im Pooling-Problem auftretenden Konzentrationen c_{jk}^{in} zu berechnen und zwei vereinfachte nichtlineare Modelle zu initialisieren. Die Ganzzahligkeitsbedingungen an die binären Variablen wurden zunächst relaxiert und erst im letzten Schritt berücksichtigt. Zusammengefasst wurde die folgende Sequenz von Untermodellen verwendet:

Teilproblem		*Problembestandteile oder Ergebnisse*
LP	$\Rightarrow$	Approximation der Konzentrationen
NLP 1	$\oplus$	Pooling, lokale WAA
NLP 2	$\Rightarrow$	relaxierte semi-kontinuierliche Ströme
RMINLP 1		Verbindungen und freie Pools
RMINLP 2		lokale WAA
MINLP		volles Modell mit allen Binärvariablen

Mit Hilfe der Modellierungssprache `GAMS` [siehe z. B. Broocke *et al.*, 1992, [45])] wurde das Modell kodiert. Als Lösungsverfahren wurde der `DICOPT`-Algorithmus (Viswanathan & Grossman, Carnegie Mellon University) mit dem Programm `CONOPT` [ARKI Consulting & Development A/S, Dänemark) zur Lösung von NLP-Problemen in Verbindung mit `CPLEX` [39] zur Lösung der MILP-Masterprobleme verwendet; zu dieser Zeit –etwa 1996– war `DICOPT` (Viswanathan & Grossmann, 1990, [286]) die einzige verfügbare Software zur Lösung von MINLP-Problemen (1.8.1) mit einigen hundert Variablen und Nebenbedingungen. Insbesondere verfügt der implementierte OA-Algorithmus über einige Erweiterungen zur Lösung nichtkonvexer MINLP-Probleme. Zur Initialisierung des Verfahrens wird die erste Linearisierung aus der Lösung der kontinuierlichen Relaxierung des MINLP-Problems abgeleitet, d. h. der Benutzer muss keine Anfangslösung bereitstellen, die alle Ganzzahligkeitsbedingungen erfüllt. Das Abbruchkriterium unterscheidet sich etwas von einem reinen „crossing bounds" Verfahren. In einem nichtkonvexen Minimierungsproblem terminiert der Algorithmus, wenn die Lösung des NLP-Problems die obere Schranke nicht mehr verbessert.

Die Rechenzeiten lagen bei etwa bei 2 Stunden auf einem PC. Zu beachten ist, dass möglicherweise nur ein lokales, nicht aber das globale Optimum berechnet wurde. Das Modell und seine Ergebnisse wurden vom Kunden sehr begrüßt, weil es sein Problem mit hoher Genauigkeit abbildete, insbesondere die Massenerhaltung der Verunreinigungen exakt berücksichtigte sowie die freien Pools richtig beschreiben konnte. Das Modell bestätigte frühere Vorschläge der im Projekt involvierten Ingenieure, was

für die Akzeptanz des Modells sehr wesentlich war, konnte darüber hinaus aber noch einige nichtintuitive Vorschläge hinsichtlich der Topologie machen, die zu erheblichen finanziellen Einsparungen führten.

10.2 Simultanes Prozessdesign und Produktionsplanung

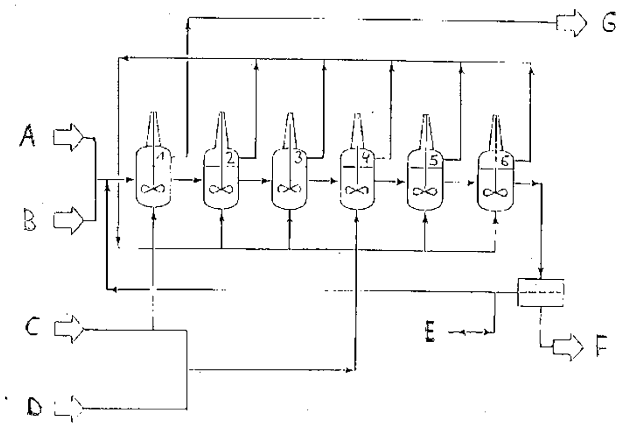
In diesem Optimierungsproblem soll ein Netzwerk von chemischen Reaktoren topologisch optimal verknüpft und verschiedene Prozessparameter so bestimmt werden, dass ein bestimmtes Wertprodukt mit bestmöglicher Ausbeute oder Selektivität bestimmt wird. Die auftretenden nichtlinearen Terme beschreiben hierbei die Reaktionskinetik (exponentiell), die Flusszusammensetzung (rational) und interpolierende Ausdrücke für die Dichte und Viskosität (polynomial). Binäre und ganzzahlige Variablen erfassen die Anzahl der Reaktoren, die Existenz von Verbindungen, die Länge von Reaktorketten und die Auswahl der Reaktorkapazität. Die Strömungsraten und Konzentrationen werden mit Hilfe kontinuierlicher Variablen beschrieben.

10.2.1 Mathematische Formulierung des Modells

Im Modell wird die folgende Indexmenge $r \in \mathcal{R} \cup \mathcal{T} := \{1, \ldots, N^R\} \cup \{p\text{-Tank}\}$ zur Beschreibung von Reaktoren und Tanks verwendet. Die meisten Modellvariablen sind die nichtnegativen (kontinuierlichen) Strömungsvariablen

$$m_{pr} = M_p n_{pr} \quad , \quad p \in \mathcal{P} := \mathcal{L} \cup \mathcal{G} := \{A, B, C, P\} \cup \{G_1, G_2, G_3\} \quad .$$

Abbildung 10.1 Typisches Netzwerk verbundener Reaktoren. Bei den Komponenten A und B handelt es sich um Flüssigkeiten, bei C und D um Gase. Mit F ist das Wertprodukt P bezeichnet.



Sie beschreiben den in einen Knoten r (dies kann ein Reaktor oder ein Produkttank sein) ein- oder abfließenden Gesamtstrom eines Produktes p oder eines Gases g. Es ist sinnvoll, zwischen Flüssigkeiten $\mathcal{L}$ und Gasen $\mathcal{G}$ zu unterscheiden, da für sie unterschiedliche topologische Bedingungen gelten. Die in Tonnen/Stunde gemessenen Variablen m sind mit den in kmol/Stunden spezifizierten Variablen n durch die Molmassen M_p verbunden. Weitere Variablen sind die Temperatur T_r und der Druck p_r^P in Reaktor r, die im Reaktor r verwendete Rührenergie e_r und als Hilfsvariablen die Gewichtsanteile w_r^p.

Das Wertprodukt P wird in einem System von Reaktoren produziert. Die Reaktoren können in bestimmten Mustern verknüpft werden: einfache Ketten fester Länge oder parallele Ketten verschiedener Länge

$$\rightarrow\ \rightarrow\ \rightarrow\ \rightarrow\ \rightarrow\ \rightarrow \quad , \qquad \begin{matrix} \rightarrow & \rightarrow & \rightarrow & \rightarrow & \rightarrow & \\ \rightarrow & \rightarrow & \rightarrow & \rightarrow & \rightarrow & \rightarrow \\ \rightarrow & \rightarrow & \rightarrow & \rightarrow & & \end{matrix} \quad ,$$

oder verknüpfte, parallele Ketten:

$$\begin{matrix} \rightarrow & \rightarrow & \rightarrow & \rightarrow & \rightarrow & \rightarrow \\ & \searrow & & \searrow & & \\ \rightarrow & \rightarrow & \rightarrow & \rightarrow & \rightarrow & \\ \rightarrow & \rightarrow & \searrow & & & \\ \rightarrow & \rightarrow & \rightarrow & \rightarrow & \rightarrow & \rightarrow \end{matrix} \quad .$$

Während der Synthese werden Nebenprodukte wie A, B oder G_1 produziert, diverse Rohmaterialen bleiben erhalten. Die flüssigen Komponenten werden einem oder mehreren nachgeordneten Reaktoren oder durch einen Filter dem Produkttank zugeführt. Der Fluss der gasförmigen Komponenten durch das Reaktorsystem wird durch Inzidenztabellen gesteuert. Der gasförmige Ausstoß einiger Reaktoren kann das System durch eine Abgasverwertungsanlage verlassen oder den anderen Reaktoren wieder zugeführt werden.

10.2.1.1 Massenbilanzierung der Reaktoren

Die Eingangsseite der Reaktoren wird durch

$$n_{pr}^i = I_{pr}^S n_{pr}^S + \sum_{s \in \mathcal{R} | I_{psr}^I = 1} x_{psr} n_{ps}^o \quad , \quad \forall r \in \mathcal{R} \cup \mathcal{T}, \quad \forall p \in \mathcal{P} \tag{10.2.1}$$

beschrieben. Für eine feste Topologie der Reaktoren spezifiziert die Inzidenztabelle I_{pr}^S, ob der Reaktor r mit dem Zufluss-Tank von Produkt p verbunden ist; die Variable n_{pr}^S beschreibt den Strom des Produktes p aus dem Zufluss-Tank zum Reaktor r. Im Falle einer komplett freien Topologie setzen wir $I_{pr}^S = 1$ für alle Kombinationen. Der zweite Term in (10.2.1) beschreibt den Strom des Produktes p von allen möglichen Reaktoren $r_s \in \mathcal{R}$ zum Reaktor r. Die Fraktionen, $0 \leq x_{psr} \leq 1$, verteilen den Ausgangsstrom eines bestimmten Reaktor auf alle nachgeordneten Reaktoren; sie unterliegen der Flusserhaltung

$$\sum_{d \in \mathcal{R} | I_{prd}^I = 1} x_{prd} = 1 \quad , \quad \forall r \in \mathcal{R} \cup \mathcal{T} \quad , \quad \forall p \in \mathcal{P} \quad .$$

Für feste Topologie mit unverknüpften parallelen Ketten gilt $x_{pr,r+1} = 1$, und $x_{prd} = 0$ für alle anderen Kombinationen. Für alle Produkte p innerhalb eines Stroms von s nach r wird die Erhaltung der Zusammensetzung durch die Pooling-Gleichung

$$x_{psr} = x_{p'sr} \quad , \quad \forall r \in \mathcal{R} \cup \mathcal{T} \quad , \quad \forall p \in \mathcal{P}$$

garantiert. Während der Synthese des Produktes P muss ein Katalysator zugeführt werden; dieser wird allerdings nicht verbraucht und kann deshalb durch eine einzige kontinuierliche Variable m^{CAT} beschrieben werden. Die Masse des Katalysators wird separat vom übrigen Produktfluss betrachtet und in den Reaktoren wiederverwendet.

Wieviel kmol/h einer Substanz p den Reaktor r verlassen, wird durch die Gleichung

$$n^o_{pr} = n^i_{pr} + \sum_{p' \in \mathcal{P} | p' \neq p \wedge S_{pp'} \neq 0} S_{pp'} \Delta n_{p'r} \quad , \quad \forall p \in \mathcal{P} \quad , \quad \forall r$$

beschrieben, wobei

$$\Delta n_{p'r} := n^o_{p'r} - n^i_{p'r} \quad , \quad \forall p \in \mathcal{P} \quad , \quad \forall r$$

die produzierte Menge bezeichnet. Aus den Durchsätzen $\Delta n_{p'r}$ des Produktes p' im Reaktor r, lassen sich mit Hilfe des bekannten, durch die stöchiometrischen Koeffizienten $S_{pp'}$ beschriebenen Reaktionsschemas die Produktions- und Verlustterme des Produktes p berechnen. Die den Reaktor r verlassende Menge des Produktes p' ist die Menge, die in r eingeflossene plus die in r produzierte Menge. Die produzierte Menge $\Delta n_{p'r}$ hängt von der Reaktionsrate $r_{p'r}$ und dem Reaktorvolumen V_r ab:

$$\Delta n_{p'r} = n^o_{p'r} - n^i_{p'r} = r_{p'r} V_r \quad , \quad \forall p' \in \mathcal{P} \quad , \quad \forall r \quad .$$

Im vorliegenden Fall wird das Produkt C bei der Herstellung des Produktes P und seinem Nebenprodukt B verbraucht, d. h. S_{CP} und S_{CB} sind von Null verschieden. Produkt A fließt von einem bestimmten zu allen zulässigen nachgeordneten Reaktoren, aber es fällt auch als Nebenprodukt der P-Synthese an, genauer während der Produktion von B und G_1, d. h. S_{AG_1} und S_{AB} sind von Null verschieden.

G_1 fällt in einer unerwünschten Reaktion von G_2 und G_3 an. Daher setzt sich die aus dem Reaktor abströmende Menge wieder zusammen aus der Eingangsmenge plus der im Reaktor produzierten Menge, die hier von der Reaktionsrate $r_{G_1 r}$ und dem Reaktorvolumen V abhängt. Die gasförmigen Einsatzstoffe G_2 und G_3 werden während der Reaktion verbraucht; der Verbrauch hängt von den produzierten Mengen P, B und G_1 ab, d. h. es müssen $S_{G_3 P}$, $S_{G_3 B}$ und $S_{G_3 G_1}$ berücksichtigt werden.

10.2.1.2 Reaktionsraten und Gewichtsanteile

Die Mengen, die an P, B und G_1 im Reaktor r produziert werden, hängen wie erwähnt von den Reaktionsraten ab. Diese sind aber wiederum Funktionen der Temperatur, Dichte, Katalysatorkonzentration und anderer Prozessparameter der chemischen Synthese.

Im folgenden sind die Reaktionsraten für die beteiligten Substanzen beschrieben. Für elementare Reaktionen wären die Raten von der Form

$$r_{AB} = k_0 c_{\mathrm{A}} c_{\mathrm{B}} e^{-E/kt} \quad ,$$

wobei c_{A} und c_{B} die Konzentrationen der Komponenten A und B bezeichnen. Im vorliegenden Reaktionsschema sind die Zusammenhänge jedoch komplizierter und hängen in nichtlinearer Weise u.a. von der Dichte und Viskosität ab; zur Beschreibung dieser Abhängigkeit werden interpolierende Funktionen verwendet. Die Reaktionsraten r_{Pr} und r_{Br} für die Synthese von P und B folgen aus

$$r_{Pr} = r_{Pr}^{(1)} - r_{G_1r}^{(2)} \quad , \quad r_r^B = C_1 m^{CAT} f_5(c_r^l) \quad ,$$

wobei c_r^l eine Hilfsgröße bezeichnet, die in der Interpolation der Reaktionskinetik benötigt wird, $r_{Pr}^{(1)}$ wird gemäß Formel (10.2.3) berechnet und C_1 ist eine bekannte Konstante. Die den Verbrauch von G_2 beschreibende Rate $r_{G_2r}^{(2)}$ ist die Summe der Einzelraten

$$r_{G_2r} = r_{Pr} + r_{Br} + r_{G_1r} \quad ,$$

und $f_n(x)$ beinhaltet die Katalysatordaten $P_1 = 25700$ und $P_2 = 400$ und ist als

$$f_n(x) := h_n(x) e^{-P_1 h_1(x)} \quad , \quad h_n(x) := \frac{x}{(1 + P_2 x)^n} \tag{10.2.2}$$

definiert. Da c_r^l von der Größenordnung $5 \cdot 10^{-5}$ ist, ist die folgende Substitution $x = s(y)$ in der Partialbruchzerlegung

$$x = 5 \cdot 10^{-5}(1 + y) \quad , \quad \frac{1+y}{\alpha + \beta y} = \frac{1}{\alpha} + \frac{\alpha - \beta}{\alpha} \frac{y}{\alpha + \beta y} \quad ,$$

sinnvoll, da (10.2.2) durch den numerisch stabileren Ausdruck

$$f_n(x) = g_n(y) := \frac{5 \cdot 10^{-5}\,(1+y)}{(1.02 + 0.02y)^n} e^{-\frac{1.825}{1.02}} e^{-\frac{1.825}{1.02} \frac{y}{1.02 + 0.02y}}$$

ersetzt werden kann. Schließlich berechnen wir die Reaktionsrate $r_{G_1r} = r_{G_1r}^{(1)} + r_{G_1r}^{(2)}$ zur Synthese von G_1

$$\begin{pmatrix} r_{G_1r}^{(1)} \\ r_{G_1r}^{(2)} \\ r_{Pr}^{(1)} \end{pmatrix} = m^{CAT} \begin{pmatrix} C_2 c_r^l f_2(c_r^l) \\ C_3 c_r^l \; c_r^P e^{-n_{Cr}^i} \\ C_4 f_3(c_r^l) \end{pmatrix} \quad , \quad c_{Pr} = \frac{n_{Pr}^o}{V_r} \quad . \tag{10.2.3}$$

Die Konzentration c_{Pr} von Produkt P im Reaktor r ist direkt verfügbar. Dagegen kann c_r^l nur implizit aus der Arrhenius-Gleichung

$$g(c_r^l) := c_r^a h_{-2}(c_r^l) + 2000 m^{CAT} f_0(c_r^l) - c_{G_2r} c_r^a \left(1 + P_2 c_r^l\right)^2 = 0 \tag{10.2.4}$$

berechnet werden, wobei c_r^a

$$c_r^a = C_5 e_r^{0.7} (1000 \rho_r)^{0.27} (0.001 \eta_r)^{-\frac{5}{6}}$$

den von der Energie e_r abhängigen Gas-Flüssigkeitsaustauschkoeffizient im Reaktor r bezeichnet. Die Konzentration c_{G_2r} von G_2 an der Phasengrenze ist durch

$$c_{G_2r} = \frac{\alpha_r}{22.4} p_{G_2r}$$

gegeben und hängt von der Gaslöslichkeit α_r ab. Sowohl α_r als auch die Viskosität η_r im Reaktor r lassen sich hinreichend genau durch Polynome vom Grad 3 in zwei Variablen

$$\alpha_r = C_6 + \left(\mathbf{v}^{\mathrm{T}} \mathsf{H}_1 \mathbf{v}\right) \quad , \quad \eta_r = C_7 + \left(\mathbf{v}^{\mathrm{T}} \mathsf{H}_2 \mathbf{v}\right) \quad , \quad \mathbf{v}^{\mathrm{T}} = (w_{Pr}, w_{Cr}) \quad ,$$

beschreiben, wobei H_1 und H_2, als auch die weiter unter benötigten Symbole H_3 und H_4 konstante Matrizen passender Dimension sind. Die mittlere Dichte ρ_r im Reaktor r wird durch ein Polynom vom Grad 4 beschrieben, d. h.

$$\rho_r = C_8 + \left(\mathbf{u}^{\mathrm{T}}\mathsf{H}_3\mathbf{u}\right)\left(\mathbf{u}^{\mathrm{T}}\mathsf{H}_4\mathbf{u}\right) \quad , \quad \mathbf{u}^{\mathrm{T}} = (w_{Pr}, w_{Br}, w_{Cr}, T_r) \quad .$$

Schließlich hängen die Gewichtsfraktionen nichtlinear von den molekularen Strömungsraten ab

$$w_{pr} = m^o_{pr} \Big/ \sum_{p' \in \mathcal{P}_p} m^o_{p'r} \quad , \quad \begin{array}{c} \forall r \in \mathcal{R} \\ \forall p \in \mathcal{P} \\ \mathcal{P}_p := \{p' \in \mathcal{P} \mid p' \neq p\} \end{array} \quad .$$

10.2.1.3 Diskrete Strukturen im Modell

Die Forderung, dass die Strömungsraten zwischen den Reaktoren bestimmte Mindestmengen nicht unterschreiten dürfen, stellt die wesentliche diskrete Bedingung im Modell dar. Diese semi-kontinuierliche Bedingung wird hier durch die Binärvariablen δ_{sr} modelliert, die im Falle $\delta_{sr} = 1$ indiziert, dass der Reaktor s eine Verbindung zum Reaktor r hat und den Ungleichungen

$$C^{\min}_{sr}\delta_{sr} \leq x_{psr} \leq C^{\max}_{sr}\delta_{sr} \quad , \quad \forall p \in \mathcal{P}_{sr}$$

genügt. Drei zusätzliche Ungleichungen stellen sicher, dass jeder Reaktor zumindest einen Zugangs- und einen Ausgangsstrom hat und dass die Anzahl nachgeordneter Reaktoren eine bestimmte Obergrenze N^{SR} nicht überschreitet, d. h.

$$\sum_{r' \in \mathcal{R}} \delta_{r'r} \geq 1 \quad , \quad 1 \leq \sum_{r' \in \mathcal{R}} \delta_{rr'} \leq N^{SR} \quad , \quad \forall r \in \mathcal{R} \quad .$$

Eine ähnliche Bedingung garantiert, dass zumindest ein Reaktor mit dem Filter verbunden ist. Dass die Flüssigkeiten A und C immer nur gleichzeitig verwendet werden können, führt auf die logische Bedingung

$$\delta_{Arr'} = \delta_{Crr'} \quad .$$

Schließlich muss noch die Reaktorgröße gewählt werden. Dies erfolgt ähnlich wie die Auswahl der Größe der Verbindungskapazität in Abschnitt 10.1.1.2 und wird hier nicht wiederholt.

10.2.1.4 Die Zielfunktion

In dieser Untersuchung werden vier alternative Zielfunktionen verwendet: die erste maximiert die als Quotient r_P/r_{G_2} definierte Selektivität des Produktes P bezüglich des teuersten Rohmaterials G_2. Hierbei ließen sich in der Tat hohe Selektivitäten erzielen, allerdings mit nur niedriger Produktausbeute von Produkt P. Daher wurde zusätzlich verlangt, dass eine Mindestmenge an P produziert wird. Die zweite Zielfunktion maximiert die produzierte Menge P. Die dritte minimiert die variablen Produktionskosten bei gleichzeitiger Erfüllung einer Mindestproduktionsmenge. Eine vierte Zielfunktion minimiert den Gesamtverbrauch an Rührenergie über alle Reaktoren.

10.2.2 Lösungsansatz

Das resultierende MINLP-Problem wurde ähnlich wie das in Abschnitt 10.1 mit `GAMS` modelliert und gelöst; Details hierzu wurden bereits auf Seite 266 spezifiziert. Die Konvergenz des Problems hängt empfindlich von der Wahl der Startwerte der Variablen ab; zu Beginn zeigten sich erhebliche Divergenzprobleme. Die numerischen Eigenschaften des Modells verbesserten sich nach einer Re-Skalierung der Variablen und Nebenbedingungen signifikant. Zudem wurde es erforderlich, einige Prozessvariablen, z. B. die Temperatur und den Druck, durch Schranken im physikalisch sinnvollen Bereich zu halten. Als Beispiel für eine nützliche Skalierung sei die Variable c_r^l genannt, die an mehreren Stellen des Modells als Argument des Ausdrucks (10.2.2) auftritt. Schließlich war es sehr wichtig, dass sich mit Hilfe von `GAMS` gute Startwerte aus der Minimierung der Verletzung einige der nichtlinearen Gleichungen gewinnen ließen. Am schwierigsten erwies sich hierbei (10.2.4). In diesem Falle konnten mit den Hilfsvariablen $\mathbf{v}^+$ und $\mathbf{v}^-$ in der Relaxierung $\mathbf{g}(\mathbf{c}) = \mathbf{v}^+ - \mathbf{v}^-$ der nichtlinearen Gleichung $\mathbf{g}(\mathbf{c}) = 0$ gute Startwerte in kurzer Zeit gewonnen werden. Für eine vorgegebene Topologie und guten Startwerten lässt sich das Problem in wenigen Minuten lösen. Diese Vorgehensweise ist typisch für die Verwendung des Modells in der operativen Produktionsplanung. Die aktuellen Anfangswerte werden gespeichert und im nächsten Rechenlauf wieder verwendet.

Das eigentliche Design- und damit das volle MINLP-Problem wurde nur von den Experten verwendet, um neue Verfahren auszuarbeiten. Die Ganzzahligkeitslücke der besten MINLP-Lösung bezüglich der NLP-Lösung ist kleiner als zwei Prozent. Hierbei handelt es sich möglicherweise nur um eine lokale NLP-Lösung, wenngleich einige offensichtlich schlechten lokalen Lösungen vom Kunden sogleich identifiziert und verworfen werden konnten.

10.2.3 Validierung und Implementierung

Nach Fertigstellung des Modells und der eigentlichen mathematischen Arbeit wurde das Modell einem eingehenden Validierungsprozess unterworfen. Prozessdaten, die über einen längeren Zeitraum gesammelt zur Verfügung standen, wurden mit den Ergebnissen der Optimierungsläufe verglichen. Dabei zeigte sich eine gute Übereinstimmung, die das Vertrauen in das neue Optimierungswerkzeug erhöhten.

Das Modell wurde darauf hin in eine attraktive und einfach zu handhabende, auf `MS-EXCEL` und `Visual Basic` basierende Oberfläche eingebettet. Die Software wurde vom Kunden für seine täglichen Planungsaufgaben (Planungshorizont von 6 Monaten auf Monatsbasis) verwendet und erlaubte ihm, sein Anlagensystem an die aktuellen Daten (Veränderung in den Kosten, Kapazitätsfluktuationen, Auftragsattribute, etc.) anzupassen. In der Designphase half das Werkzeug, die Reaktoranordnung auszulegen, wobei die als optimal oder zumindest mit einer Qualität versehenen Lösungen das Leben erheblich erleichterten im Vergleich zu sonst zu vergleichenden Simulationsszenarien. Die neuen Reaktoranordnungen reduzieren den Bedarf an Rohmaterialien, minimieren die Produktion unerwünschter Nebenprodukte und erhöhen die Kapazität des Gesamtsystems.

10.2.4 Ergebnisse und Vorteile für den Kunden

Gemeinsam mit dem Kunden untersuchten wir typische Referenzszenarien und die Auswirkungen der dem Optimierungsproblem innewohnenden Freiheitsgrade. Hierbei gingen wir, wie in Abb. 10.1 gezeigt, von einer festen Kette von 6 Reaktoren, die bereits vor Projektbeginn in Betrieb waren, und den folgenden Zielfunktionen und speziellen Nebenbedingungen aus, wobei sich im Vergleich zu den Referenzszenarien die folgenden Resultate ergaben:

- Mit Hilfe einiger Berechnungen, die auf die *optimale Katalysatorkonzentration* abzielten, konnte eine Verständnisbasis für einige bis dahin quantitativ unverstandene, in mehreren Anlagen beobachtete Effekte gewonnen werden. Die Auswirkungen der Katalysatorkonzentration auf die Produktionsraten und die Selektivität hängen dabei stark von den Operationsbedingungen der Reaktoren und den geforderten Produktionsmengen ab, wobei diese wiederum von Standort zu Standort variieren. Diese Effekte lassen sich nun quantitativ modellieren, sodass jeder Standort für sich nun die optimale Katalysatorkonzentration wählen kann, wobei sowohl die Produktionsbedürfnisse als auch die technischen und ökonomischen Bedingungen Berücksichtigung finden. Darüber hinaus konnten die Katalysatoreffekte hinsichtlich erhöhter Anlagekapazitäten extrapoliert werden. Damit wurden bereits Maßnahmen vorbereitet, die einige in der Zukunft zu erwartenden Engpässe beseitigen.
- In einer zweiten Studie wurde der *Einfluss der Rückströmung* von Produkten aus dem letzten Reaktor in den ersten untersucht. Hierbei konnte gezeigt werden, dass sich die Rückströmung negativ auf die gesamt produzierte Menge auswirkt. Die optimale Lösung schlägt keine Rückströmung vor. Zwar ist eine Rückführung derzeit aus technischen Gründen auch nicht möglich, da sich aber nun die Kostenersparnis durch Wegfall der Rückströmung mit den Kosten eines Anlagenumbaus quantitativ vergleichen lässt, konnte die Basis für ein entsprechendes Projekt gelegt werden.
- Bei der Untersuchung der Gaszufuhr und der *Topologie der Gasströmungen* konnten einige unerwartete Ergebnisse gewonnen werden. Hierbei wurde offensichtlich, dass die Beschränkung der Kapazität des Gaskompressors wichtig ist; die Optimierung zeigte dennoch Möglichkeiten auf, die begrenzten Ressourcen effizienter zu nutzen. Bestimmte Verschaltungen der Reaktoren ermöglichten eine Erhöhung der Produktionskapazität um etwa 12-18% bei gleichbleibender Selektivität.
- Die *Vermeidung von Nebenprodukten* ist ein wesentlicher Aspekt. Daher war es ein Ziel einer bestimmten Studie, die Prozessparameter derartig einzustellen, dass die Produktionsmenge maximal, die Menge an Nebenprodukten jedoch auf demselben niedrigen Niveau wie im Referenzszenario zu halten ist. Es war überraschend, dass sich tatsächlich ein Produktionszuwachs in Höhe von 5% erzielen ließ, ohne dabei irgendwelche technische Randbedingungen zu verletzen oder einen Selektivitätsverlust hinnehmen zu müssen.
- Bei der Minimierung der Rührenergie bei gleichzeitiger Erfüllung der aus dem Referenzszenario bekannten Mindestproduktionsmenge konnte gezeigt werden, dass bei einem geringfügig höherem Verbrauch des Gases G_2 und etwas weniger Verbrauch des Gases G_3 die gesamte Energie, die benötigt wurde, etwa 8% niedriger als im Referenzszenario lag.

Je nach Zielfunktion ergeben sich nicht ganz unerwartet verschiedene Designlösungen. Mit Hilfe des Modells und der verfügbaren Software ist es nun auch möglich, die Stabilität dieser Designlösungen hinsichtlich der verschiedenen Zielfunktionen zu untersuchen.

Das Programm wird meist genutzt, um die verschiedenen Designoptionen (Zahl, Größe und Topologie der Reaktoren) quantitativ zu bewerten. Infolge der Komplexität der Abhängigkeiten der Prozessparameter untereinander ist es schwierig, die Auswirkungen bestimmter Änderungen auf Selektivität, variable Kosten und Produktionskapazität zu überschauen. Daher ist es sinnvoll, für die jeweiligen, durch verschiedene technische und ökonomische Randbedingungen fixierten Szenarien die optimalen Lösungen miteinander zu vergleichen. Das MINLP-Modell ermöglicht dies. Insbesondere ist es ein riesiger Vorteil, nach einer Produktionsmaximierung zu fragen, ohne dabei an Selektivität zu verlieren oder die Kosten zu erhöhen. Mit Hilfe der graphischen Oberfläche lassen sich vom Benutzer bestimmte Verschaltungen in einer Verbindungsmatrix innerhalb von Sekunden sehr leicht ändern und durchrechnen. Bei fester Topologie dauert ein typischer Optimierungslauf etwa 10 Sekunden auf einem Standard 266 MHz Pentium PC mit 92 MByte RAM.

Die Evaluierung von etwa 20 konzeptionell verschiedenen Designoptionen zeigte ein Potential zur Produktionserhöhung von etwa 12-18% auf, sowie eine Liste von möglichen Verschaltungen, die mit einem rein intuitiven Ansatz wohl nicht gefunden worden wären. Einige dieser vielversprechenden Verschaltungen haben sich bereits in der Produktion bewährt. Zudem wurden einige nicht sehr teuere Designänderungen identifiziert, die zu einer weiteren Kapazitätserhöhung führen. Die MINLP-Lösungen zeigen, dass unter den gegenwärtigen Randbedingungen die zur Zeit existierende Verschaltung recht sinnvoll und nahe dem Optimum liegt; es ist nun mit der Software aber möglich, besser das Potential abzuschätzen, wenn man verschiedene Randbedingungen relaxiert oder ganz wegfallen lässt.

11 Globale Optimierung in der Praxis

Globale Optimierungstechniken, siehe beispielsweise Horst & Pardalos (1995,[136]), Floudas (2000[89]), Floudas & Gounaris (2009,[90]) oder Misener & Floudas (2012,[213]), eignen sich zur Lösung nichtkonvexer NLP- oder MINLP-Probleme; wir meinen hierbei deterministische globale Optimierung, bei der ein global-optimaler Zielfunktionswert bis auf ein vorgegebenes $\varepsilon > 0$ berechnet werden kann. Nähert sich ε der Maschinengenauigkeit, so stellen sich grundsätzliche Fragen, die in der Disziplin *Reliable Computing* besser aufgehoben sind; wir beschränken uns daher hier auf $\varepsilon > 10^{-6}$, was für die meisten praktischen Probleme auch völlig hinreichend ist. Limitierend wirkt in der Praxis der in der Anzahl der nichtlinear auftretenden Variablen expontiell anwachsende Rechenaufwand. Seit 2002 wurde eine Reihe von kommerziell verfügbaren Solvern, darunter BARON [101], LINDOGLOBAL [261] und GLOMIQO [213], entwickelt, die systematisch in der Literaturübersicht des Artikels von Misener & Floudas (2012,[213]) beschrieben werden.

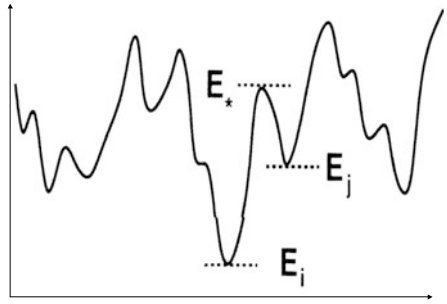
In der Mineralölindustrie wird versucht, Planungs- und Scheduling-Probleme, die im Kern *Pooling-Probleme* enthalten, mit Hilfe von Methoden der globalen Optimierung zu lösen; verwandt damit ist auch die Produktion von Naturgas. Fragestellung im Umfeld von Wassersystemen oder Nahrungsmittelproduktion enthalten häufig ebenfalls Pooling-Probleme und bedürfen damit auch globaler Optimierungstechniken. Das in diesem Kapitel dargestellte Verschnittproblem stammt aus Adjiman (1999,[9]). Weitere Anwendungsbeispiele finden sich in Floudas *et al.* (1999,[92]), Floudas (2000,[89]) oder Kallrath (2004,[158]). Besonders erwähnenswert ist auch, dass es mit den Methoden der globalen Optimierung wie in Maranas & Floudas (1994,[276]) möglich ist, sämtliche Nullstellen eines Systems nichtlinearer Gleichungen zu bestimmen. Dies findet z. B. Anwendung in der Bestimmung aller stationären Zustände chemischer Reaktoren; Maranas & Floudas (1994,[276]) wenden es auf die Bestimmung möglicher Energieniveaus von Molekülen, Ratschek & Rokne (1993,[238]) auf Schaltkreis-Design-Probleme an. Anwendungen auf Parameterschätzung und Robotik finden wir in Jaulin *et al.* (2001,[147]). Misener & Floudas (2012,[213]) enthält darüber hinaus noch Anwendungen aus dem Bereich *Comptutional Geometry.*

11.1 Energieminimale Molekülkonfiguration

Das hier diskutierte Beispiel stammt ursprünglich aus Maranas & Floudas (1994,[276]). Es enthält nur eine einzige Variable und keine Nebenbedingungen, zeigt aber, wie man im Rahmen der Globalen Optimierung mit konvexen Unterschätzern und dem uniformen diagonalen Shift-Verfahren arbeiten kann. Betrachtet wird ein Pseudo-Äthanmolekül, d. h. all die Wasserstoffatome des Äthanmoleküls wurden durch C-, N- oder O-Atome ersetzt. Das Lennard-Jones-Potential hängt vom Diederwinkel t, $0 \leq t \leq 2\pi$, wie folgt

J. Kallrath, *Gemischt-ganzzahlige Optimierung: Modellierung in der Praxis*,
DOI 10.1007/978-3-658-00690-7_11, © Springer Fachmedien Wiesbaden 2013

Abbildung 11.1 Energiefunktion in einem quantenchemischen Problem als Funktion einer räumlichen Koordinate. Die lokalen Energieminima entsprechen stabilen Zuständen eines Moleküls.



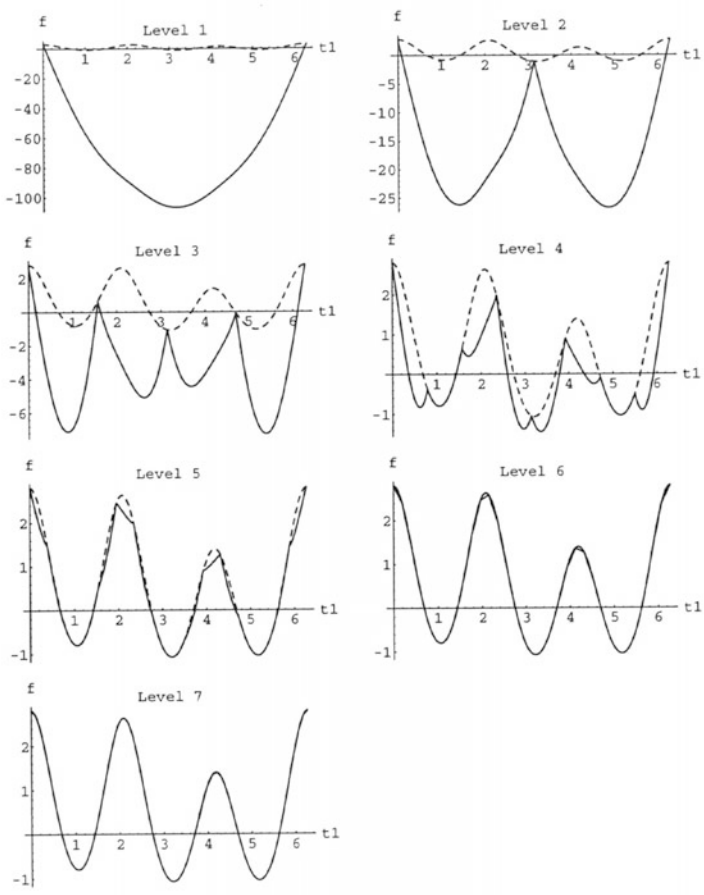
$$
\begin{aligned}
f(t) \quad = \quad & \sum_{k=-1}^{+1} \frac{A_k}{\left\{3r_0^2 - 4r_0^2\cos\theta - 2r_0^2\left[\sin^2\theta\cos\left(t+\frac{2}{3}\pi k\right) - \cos^2\theta\right]\right\}^6} \\
& - \sum_{k=-1}^{+1} \frac{B_k}{\left\{3r_0^2 - 4r_0^2\cos\theta - 2r_0^2\left[\sin^2\theta\cos\left(t+\frac{2}{3}\pi k\right) - \cos^2\theta\right]\right\}^3}
\end{aligned}
$$

ab, wobei die Beiträge von Atomwechselwirkungen, die nicht von t abhängen, hier nicht aufgenommen wurden, da sie nur einen konstanten Beitrag liefern. Hierbei bezeichnet r_0 die kovalente Bindungslänge (0.154nm); θ den kovalenten Bindungswinkel (109°.5); die Konstanten A_k und B_k haben – in kcal/mol – die Werte

$$
\begin{array}{rr}
A_{-1} = 588600 & B_{-1} = 1079.1 \\
A_0 \ = 600800 & B_0 \ = 1071.5 \\
A_{+1} = 481300 & B_{+1} = 1064.6
\end{array} \quad .
$$

Gesucht ist der Winkel t, für den die Funktion $f(t)$ ihr Minimum annimmt. In Abb. 11.2 (Level 7) sieht man, dass $f(t)$ im Intervall $\mathcal{I} = \mathcal{I}_1^1 = [0, 2\pi]$ drei Minima hat. Das mittlere von ihnen bei $t \approx 3.2$ oder $t \approx 183°.45$ ist das globale Minimum mit einem minimalen Funktionswert von -1.0711 kcal/mol.

Abbildung 11.2 Funktion und konvexe Unterschätzer, die mit exakt berechneten α-Werten ermittelt wurden. Diese Abbildung wurde mit freundlicher Genehmigung von Claire Adjiman aus [9], Abb. 3.3, entnommen; sie ist ebenfalls in [8] enthalten. Der Vergleich mit Abb. 11.3 zeigt in Level 3 und Level 4 wesentlich genauere Unterschätzer.



Level	Intervall/π	α	$d_{\max}$	α	$d_{\max}$
1	[0.00, 2.00]	10.7	105.6	38000.0	380000.0
2	[0.00, 1.00]	10.7	26.4	9500.0	23000.0
2	[1.00, 2.00]	10.7	26.4	8600.0	21000.0
3	[0.00, 0.50]	10.7	6.6	944.2	582.4
3	[0.50, 1.00]	10.4	6.4	298.9	184.4
3	[1.00, 1.50]	7.9	4.9	262.0	161.6
3	[1.50, 2.00]	10.7	6.9	869.2	536.2
4	[0.00, 0.25]	10.7	1.7	56.8	8.8
4	[0.25, 0.50]	0.0	0.0	93.5	14.4
4	[0.50, 0.75]	10.7	1.6	41.3	6.4
4	[0.75, 1.00]	5.5	0.8	87.0	13.4
4	[1.00, 1.25]	4.0	0.6	75.8	11.7
4	[1.25, 1.50]	7.9	1.2	34.5	5.3
4	[1.50, 1.75]	0.0	0.0	84.4	13.0
4	[1.75, 2.00]	10.7	1.7	56.7	8.7

Abbildung 11.3 Funktion und konvexe Unterschätzer; die α-Werte wurden mit der Intervall-Matrix-Methode nach Hertz (1992,[133]) berechnet. Diese Abbildung wurde mit freundlicher Genehmigung von Claire Adjiman aus [9], Abb. 3.4, entnommen; sie ist ebenfalls in [8] enthalten.

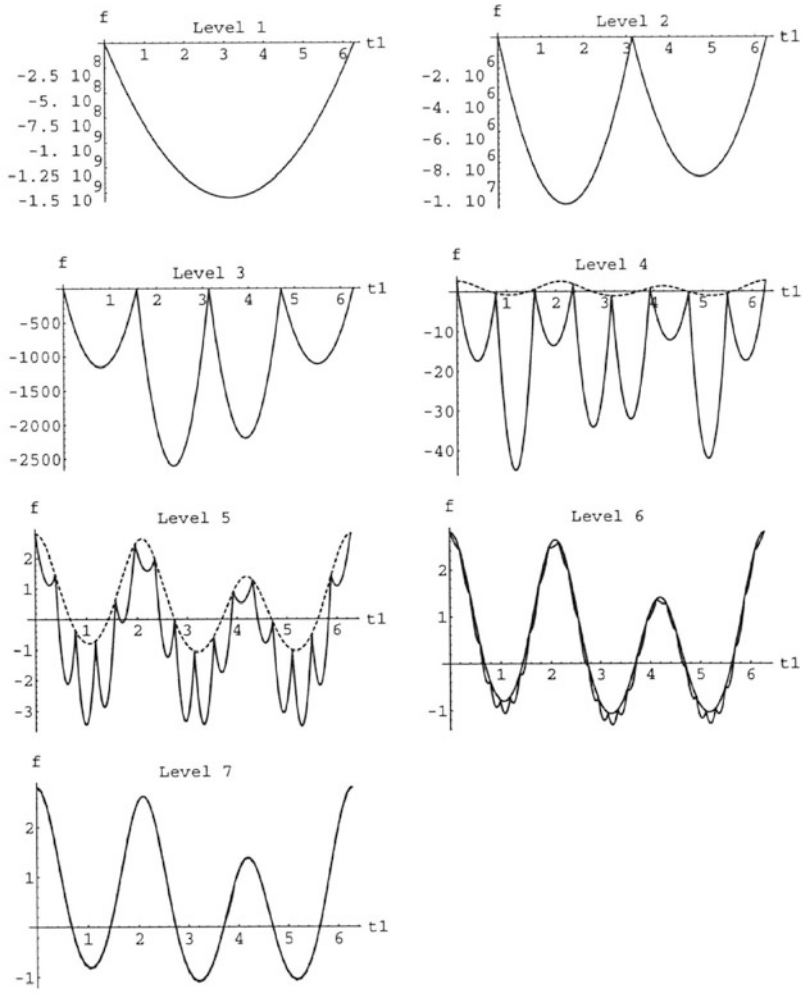
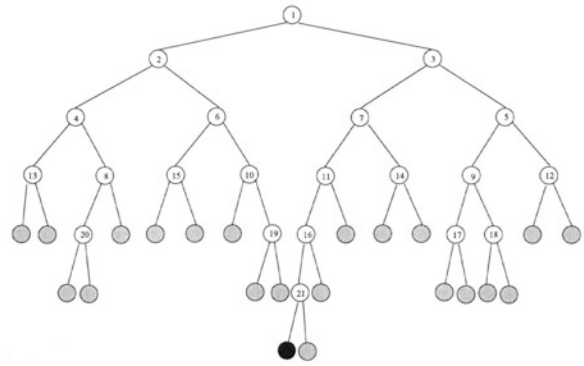


Tabelle 10.1 zeigt die exakten (Fall E) und die mit Hilfe der Intervall-Hesse-Matrix berechneten (Fall I) α-Werte für die ersten vier äquidistanten Zerlegungen des zulässigen Bereichs $[0, 2\pi]$ der Variablen t; das i-te Intervall im Level L lautet

$$\mathcal{I}_L^i := \left[\frac{2}{L}\pi(i-1), \frac{2}{L}\pi i\right] \quad , \quad 1 \le i \le L \quad .$$

Abbildung 11.4 Branch&Bound-Baum beim aBB-Verfahren (Fall E). Diese Abbildung wurde mit freundlicher Genehmigung von Claire Adjiman aus [9], Abb. 3.5, entnommen. Die Zahl in den weißen Knoten des Baumes bezeichnet die Iterationsnummer, verworfene Knoten sind in grau markiert, und der optimale Knoten ist in schwarz dargestellt.



Mit $d_{\max}$ ist die maximale Differenz der Funktion $f(t)$ und ihrer konvexen Unterschätzer gemäß (4.6.3) bezeichnet. Tabelle 10.2 zeigt die Ergebnisse für die nachfolgenden Zerlegungen $L = 5 \ldots 8$.

Level	Intervall/π	α	$d_{\max}$	α	$d_{\max}$
5	$[0.00, 2.00]$	0-10.7	0-0.413	10.2-23.5	0.393-0.906
6	$[0.00, 1.00]$	0-10.7	0-0.103	2.8-14.0	0.027-0.135
7	$[1.00, 2.00]$	0-10.7	0-0.026	0.0-10.7	0.000-0.026
8	$[0.00, 0.50]$	0-10.7	0-0.006	0.0-10.7	0.000-0.006

.

Zu beachten ist, dass mit kleineren Intervallgrößen bzw. wachsendem L die berechneten α-Werte sich den exakten Werten erwartungsgemäß annähern. Wie in den Abbildungen 11.2 und 11.3 ersichtlich wird, sind die Unterschätzer, die mit Hilfe exakter α-Werte berechnet wurden, viel genauer als die mit der Intervallmethode berechneten Werte. Die Abbildungen 11.4 und 11.5 zeigen die B&B-Bäume für die Fälle I und E. Beide beginnen mit einer Breitensuche, die dann in eine Tiefensuche übergeht. Bei einer Abbruchgenauigkeit von $\varepsilon = 10^{-6}$ wurden 21 Iterationen im Fall I benötigt, dagegen nur 10 im Fall E benötigt; die meisten Knoten wurden bereits im Level 5 bzw. 4 verworfen.

11.2 Ein Verschnittproblem aus der Papierindustrie

Hier stehen wir wie in Abschnitt 6.2 vor der Aufgabe, Papierprodukte verschiedener Größe aus einer großen Papierrollen zu schneiden, um damit Kundennachfragen zu befriedigen. Diesmal soll dabei aber nicht nur die Wahl der Schnittmuster und deren Multiplizität, sondern auch die Schnittmuster selbst Freiheitsgrad der Optimierung sein. Diese Fragestellung, die in Arbeiten von Westerlund *et al.* (1994,[290]) und Harjunkoski (1997,[128]) behandelt wird, führt auf ein nichtlineares, nichtkonvexes ganzzahliges Optimierungsproblem.

Abbildung 11.5 Branch&Bound-Baum beim aBB-Verfahren (Fall I). Diese Abbildung wurde mit freundlicher Genehmigung von Claire Adjiman aus [9], Abb. 3.6, entnommen.

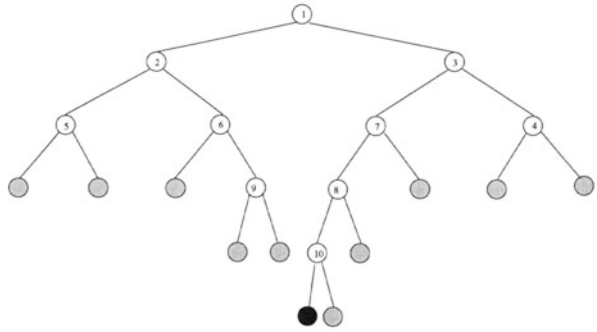
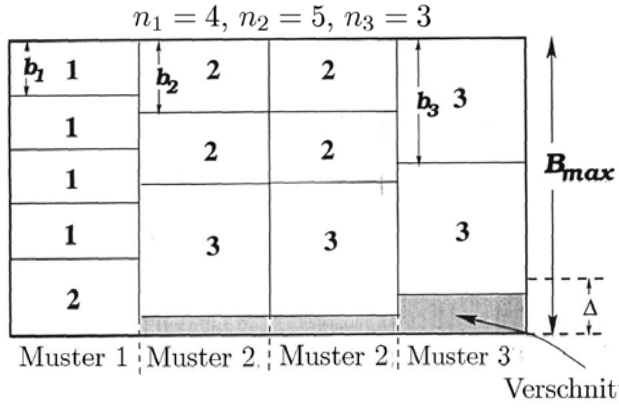


Abbildung 11.6 Verschnittproblem mit 3 Mustern. Diese Abbildung wurde mit freundlicher Genehmigung von Claire Adjiman aus [9], Abb. 9.7, entnommen.



Jede Produktpapierrolle ist durch ihre Weite b gekennzeichnet. Alle Produktrollen sollen von gleicher Länge sein. Die Rohpapierolle hat die Weite $B_{\max}$ und eine als unendlich lang angenommene Länge. Im allgemeinen ist es nicht möglich, einen gesamten Auftrag abfallfrei aus dem Rohpapier zu schneiden. Das optimale Verschnittmuster minimiert den Verschnitt. Um das beste Schema zu identifizieren, wird eine maximale Anzahl verschiedener Schnittmuster N^M vorgeschlagen, wobei ein Muster durch die Positionierung der Messer definiert ist. Jedes Schnittmuster enthält ein oder mehrere Teilstücke aus einem Sortiment von N verschiedenen Größen b_i und kann mehrmals im gesamten Sche-

ma wiederholt werden, um die Nachfrage nach n_i Stück der Produktgröße b_i zu erfüllen. Die Selektion eines jeden Musters wird durch eine Binärvariable δ_m, $m = 1, \ldots, N^M$, die Anzahl der Wiederholungen des Musters m durch die ganzzahlige Variable μ_m beschrieben. Die Anzahl der Produkte der Größe i im Muster m wird durch die ganzzahlige Variable ρ_{im} charakterisiert. Das in Abb. 11.6 gezeigte Muster ist durch

$$N = 3 \quad , \quad n = \begin{pmatrix} 4 \\ 5 \\ 4 \end{pmatrix} \quad , \quad \mu = \begin{pmatrix} 1 \\ 2 \\ 1 \end{pmatrix} \quad , \quad \rho = \begin{pmatrix} 4 & 1 & 0 \\ 0 & 2 & 1 \\ 0 & 0 & 2 \end{pmatrix}$$

parametrisiert; wir wollen die Muster jedoch später so anordnen, dass die häufiger verwendeten als erstes aufgeführt werden. Die Schnittmuster unterliegen noch einigen besonderen Nebenbedingungen. So muss z. B. jedes Muster eine minimale Breite von $B_{\max} - \Delta$ haben. Die Anzahl der im Einsatz befindlichen Messer ist durch N^K begrenzt, was bedeutet, dass nicht mehr als N^K Produkte in einem Muster sein können. Damit ergibt sich das folgende Minimierungsproblem:

$$\min_{m_m, \delta_m, r_{im}} \sum_{m=1}^{N^M} C_m^M \mu_m + C_m^Y \delta_m$$

unter den Nebenbedingungen: Erfüllung der Nachfrage

$$\sum_{m=1}^{N^M} \mu_m \rho_{im} \geq n_i \quad , \quad \forall i \quad ,$$

Berücksichtigung der unteren und oberen Weite bei Selektion der Muster

$$(B_{\max} - \Delta)\delta_m \leq \sum_{i=1}^{N} b_i \rho_{im} \leq B_{\max} \delta_m \quad , \quad m = 1, \ldots, N^M \quad ,$$

Beschränkung der Schnittmesser und damit der Anzahl der Produkte in einem Muster

$$\delta_m \leq \sum_{i=1}^{N} \rho_{im} \leq N^K \delta_m \quad , \quad m = 1, \ldots, N^M \quad ,$$

Zählung und Beschränkung der maximalen Wiederholung M eines Musters

$$\delta_m \leq \mu_m \leq M \delta_m \quad , \quad m = 1, \ldots, N^M \quad ,$$

eine spezielle Schnittbedingung, die sich aus der Nachfrage ableitet

$$\sum_{m=1}^{N^M} \mu_m \geq \max \left\{ \left\lceil \frac{\sum_{i=1}^{N} n_i}{N^K} \right\rceil , \left\lceil \frac{\sum_{i=1}^{N} b_i n_i}{B_{\max}} \right\rceil \right\} \quad ,$$

wird das Muster $m+1$ gewählt, so muss – um zu vermeiden, dass Muster definiert werden, die gar nicht verwendet werden – auch das Muster m gewählt werden

$$\delta_{m+1} \leq \delta_m \quad , \quad m = 1, \ldots, N^M - 1 \quad ,$$

das Muster m soll häufiger als Muster $m + 1$ verwendet werden

$$\mu_{m+1} \leq \mu_m \quad , \quad m = 1, \ldots, N^M - 1 \quad ,$$

und schließlich die Ganzzahligkeitsbedingungen

$$\delta_m \in \{0, 1\} \quad , \quad m = 1, \ldots, N^M \quad ,$$

$$\mu_m \in \{0, M_m\} \cap \mathbb{N}_0 \quad , \quad m = 1, \ldots, N^M$$

und

$$\rho_{im} \in \{0, N^K\} \cap \mathbb{N}_0 \quad , \quad i = 1, \ldots, N \quad , \quad m = 1, \ldots, N^M \quad .$$

Im realen Problem hatten die Kostenkoeffizienten die Werte

$$C_m^M = 1.0 \quad , \quad C_m^Y = 0.1 \quad , \quad m = 1, \ldots, N^M \quad .$$

Das Problem wurde in Adjiman (1999,[9]) für die in der nachfolgenden Tabelle genannten Fälle gelöst; $B_{\max}$, Δ und b sind in mm spezifiziert.

#	N P $B_{\max}$ Δ	n	b	M	$\log_{10} N^P$ z	μ	r	Δ^I
1	4 4 1850 100	15	290	30	16 19.6	3	2 0 0 0	0.50
		28	315	30		2	0 5 3 0	
		21	350	30		1	1 0 1 0	
		30	455	30		0	2 0 1 0	
2	4 4 1900 200	9	330	15	16 8.6	11	1 0 0 0	0.50
		7	360	12		0	1 0 0 0	
		12	385	9		0	2 0 0 0	
		11	415	6		0	1 0 0 0	
3	5 5 2000 200	12	330	15	23 10.3	15	1 0 0 0 0	0.20
		6	360	12		0	1 0 0 0 0	
		15	370	0		0	1 0 0 0 0	
		6	415	0		0	1 0 0 0 0	
		9	435	0		0	1 0 0 0 0	
4	6 6 2200 100	8	330	15	31 15.3	8	1 0 0 0 0 0	0.08
		16	360	12		7	2 0 0 0 0 0	
		12	380	8		0	0 2 0 0 0 0	
		7	430	7		0	0 1 0 0 0 0	
		14	490	4		0	0 2 0 0 0 0	
		16	530	2		0	1 0 0 0 0 0	

Für jedes Muster gab es meist mehrere globale Optima; die Tabelle zeigt nur eines. Die Ganzzahligkeitslücke ist bei diesem Problem erfreulich klein. Die in der Tabelle gezeigten Lösungen wurden mit dem GMIN-αBB Algorithmus (Adjiman, 1999, [9]) berechnet. Da die Nichtexistenz eines Musters m, d. h. $\delta_m = 0$, die korrespondierenden Variablen μ_m und ρ_{im} eliminiert, wird auf die Variablen $\delta_m = 0$ mit höchster Priorität verzweigt.

11.3 Zuschnitt- und Packprobleme mit Kreisen, Polygonen und Ellipsen

Die Böden großer Tanks, wie sie z. B. in Bierbrauereien verwendet werden, werden aus einer kreisförmigen Grundfläche gebogen und konvex verformt. Sie werden aus hochwertigem Edelstahl gefertigt, wobei die Kreise aus rechteckigen Metallplatten [engl.: *coils*]

geschnitten werden, wobei diese Rechtecke z. B. der Produktionsbeschränkung unterliegen, dass sie nicht länger als $L = S_1^{\mathrm{P}} = 8\mathrm{m}$ und nicht breiter als $B = S_2^{\mathrm{P}} = 4\mathrm{m}$ sein dürfen. Benötigte Kreise, die diese Maße überschreiten, werden in konvexe Polygone zerlegt und danach zusammengeschweißt. Da dieser Edelstahl sehr teuer ist, sollen zu einer Menge von Kreisen oder Polygonen Auftragsrechtecke minimaler Fläche produziert werden; spezielle Modelle und Lösungsansätze, die die Überschneidungsfreiheit konvexer Polygone mit Hilfe separierenden Hyperebenen garantieren, wurden hierzu von Kallrath (2009,[161]) entwickelt. Alternativ können auf bereits gefertigte Rechtecke auf Lager vorrätig sein. Dann besteht die Aufgabe darin, eine optimale Flächenausnutzung durch Zuordnung von Auftragsobjekten und Bestandsrechtecken zu erhalten; diese Aufgabe wird in Rebennack *et al.* (2009,[244]) durch einen Spaltenenumerierungsverfahren gelöst, bei dem nichtkonvexe NLP-Probleme global gelöst werden.

Farbe, die in größeren Mengen auf Wände aufgetragen werden soll, wird häufig in Eimern elliptischer Grundform aufbewahrt, damit die Anstrichrollen möglichst groß sein können und das Gewicht der vollen Farbeimer nicht zu groß wird. Zum Transport solcher Eimer kann es sinnvoll sein, große und kleine Eimer auf LKW-Ladeflächen zu kombinieren, um eine möglichst gut Flächenauslastung zu erzielen. Da sich Ellipsen auch gut eigenen, andere geometrische Figuren durch Überdeckung zu approximieren, sind Pack- oder Verschnittprobleme mit Ellipsen oder später in drei Dimensionen mit Ellipsoiden an sich schon sehr interessant. Kallrath & Rebennack (2013,[164]) entwickelten hierzu separierende Hyperebenen, die letztlich zu einer geschlossenen Formulierungs des Problems führten und es erlauben, eine gegebene Menge von Ellipsen auf flächenminimale Rechtecke zu platzieren.

Betrachten wir hier lediglich den Fall, dass alle zu platzierenden Objekte $i \in \mathcal{I}$ tatsächlich in ein Rechteck $\mathcal{R}$ der erlaubten Größe passen, d. h. $x_d^{\mathrm{P}} \leq S_d^{\mathrm{P}}$, $d \in \{1, 2\}$; die Fläche

$$a = x_1^{\mathrm{P}} x_2^{\mathrm{P}} \tag{11.3.1}$$

dieses Rechtecks soll minimal werden. Mit Hilfe geeigneter Ungleichungen müssen wir sicherstellen, dass alle Objekte die Grenzen von $\mathcal{R}$ nicht überschreiten und sich nicht überschneiden. Im Folgenden wollen wir diese Bedingungen für Kreise, Polygone und Ellipsen formulieren.

11.3.1 Modellierung der Zuschnittbedingungen

11.3.1.1 Zuschnittbedingungen für Kreise

Handelt es sich bei unseren Objekten nur um Kreise mit Radien R_i, so sind die Bedingungen recht einfach. Bezeichnet $\mathbf{x}_i$ das Zentrum des Kreises i, so wird die Überschneidungsfreiheit durch

$$(\mathbf{x}_i - \mathbf{x}_{i'})^2 \geq (R_i + R_{i'})^2 \quad , \quad \forall\{\ (i, i') \mid i < i'\} \tag{11.3.2}$$

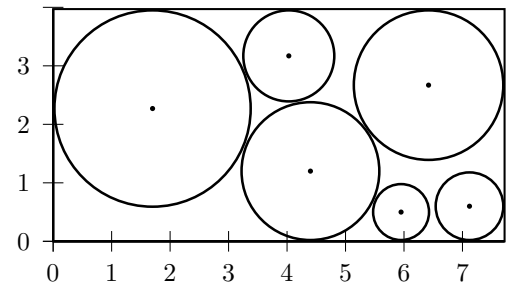
sichergestellt. Für n Kreise ergeben sich daher $n(n-1)/2$ Ungleichungen vom Typ (11.3.2).

Die Ungleichungen

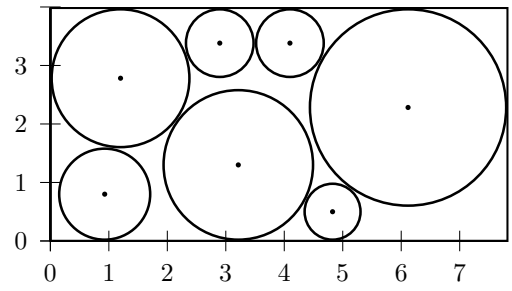
$$x_{id} \geq R_i \quad ; \quad \forall\{i, d\} \tag{11.3.3}$$

und

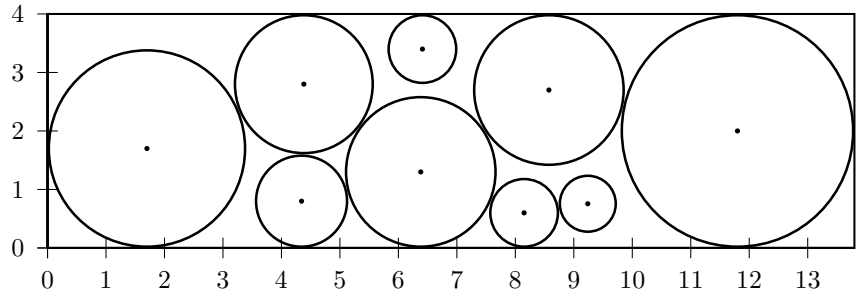
Abbildung 11.7 Zuschnittlösung für Kreise verschiedener Radien. Alle Lösungen konnten in kurzer Zeit berechnet werden; globale Optimalität wurde mit GAMS/Lindoglobal nachgewiesen.



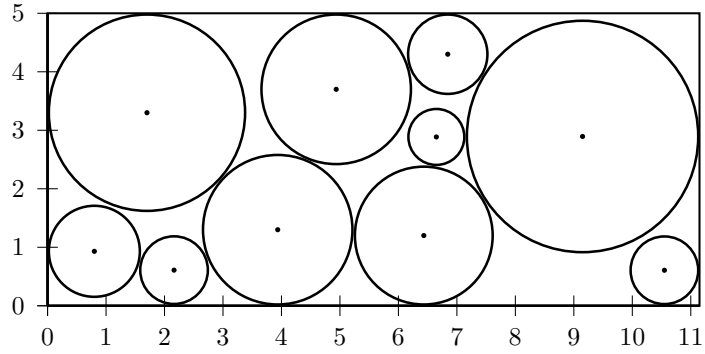
(a) c6



(b) c7



(c) c8



(d) c10

$$x_{id} + R_i \leq x_d^{\mathrm{P}} \leq S_d^{\mathrm{P}} \quad ; \quad \forall\{i,d\} \tag{11.3.4}$$

garantieren, dass kein Kreis die Grenzen des Rechtecks überschreitet.

11.3.1.2 Zuschnittbedingungen für Polygone

Ein Polygon p sei durch seine K_p Eckpunkte [engl.: *vertices*] $V_{p1}, \ldots, V_{pK_p}$, bzw. durch deren Koordinaten $\mathbf{X}_{pk}$, $k = 1, \ldots, K_p$, charakterisiert. Zwar gibt es interessante Anwendungen nichtkonvexer Polygone in der Textilindustrie, hier sollen aber nur konvexe Polygone betrachtet werden. Die Polygone sind implizit und vollständig durch ihre Zentren, die Richtung vom Zentrum zu den Eckpunkten und Abstände zu den Zentren sowie durch die Orientierung beschrieben; siehe Fig. 11.8. Das Zentrum $\mathbf{X}_p^0$ des ursprünglichen, unverschobenen und nichtrotierten Polygons ist durch

$$\mathbf{X}_p^0 = \frac{1}{K_p} \sum_{k=1}^{K_p} \mathbf{X}_{pk} \quad ; \quad \forall\{p\} \tag{11.3.5}$$

definiert. Die Polygone können nun in einem beliebigen, durch den Vektor $\mathbf{x}_q^0$

$$\mathbf{x}_p^0 = \frac{1}{K_p} \sum_{k=1}^{K_p} \mathbf{x}_{pk} \quad ; \quad \forall\{p\} \tag{11.3.6}$$

definierten Zentrum platziert werden. Die Erhaltung der Gestalt und Orientierung wird durch die Gleichung

$$\mathbf{x}_{pk} = \mathbf{x}_p^0 + \begin{pmatrix} \cos\alpha_p & \sin\alpha_p \\ -\sin\alpha_p & \cos\alpha_p \end{pmatrix} \left(\mathbf{X}_{pk} - \mathbf{X}_p^0\right) \quad ; \quad \forall\{p,k\} \tag{11.3.7}$$

beschrieben. Möchte man trigonometrische Terme vermeiden, so kann man anstelle des Rotationswinkels α als freie Variable auch $v_p := \cos\alpha_p$ und $w_p := \sin\alpha_p$ mit den Schranken $-1 \leq v_\alpha \leq +1$ and $-1 \leq w_\alpha \leq +1$ als Variablen verwenden, wobei diese weiter durch die trigonometrisch Identität

$$v_p^2 + w_p^2 = 1 \quad ; \quad \forall\{p\} \tag{11.3.8}$$

verknüpft sind. Für Polygone mit Symmetrieaxe brauchen wir nur Winkel im Intervall von 0° bis 180° zu berücksichtigen, d. h. $-1 \leq w_\alpha \leq 1$. Weitere Symmetrieeigenschaften können bei regulären Polygonen ausgenutzt werden.

Die Bedingung, dass ein konvexes Polygon vollständig in $\mathcal{R}$ liegt, reduziert sich auf die Bedingung, dass alle Eckpunkte im Innern von $\mathcal{R}$ liegen, d. h.

$$X_{pkd} \leq x_d^{\mathrm{P}} \leq S_{\max,d} \quad ; \quad \forall\{p,k,d\} \quad . \tag{11.3.9}$$

Um nun sicherzustellen, dass zwei Polygone p und q' sich nicht überlappen, müssen wir etwas anders vorgehen als bei Kreisen, bei denen die Überlappungsfreiheit lediglich durch eine Abstandsbedingung der Zentren beschrieben werden konnte. Bei beliebigen konvexen Objekten in der Ebene, dazu zählen hier die Polygone und später auch die Ellipsen, können wir uns zunutze machen, dass jede Gerade, die nicht durch das Innere des Objektes verläuft, die Eigenschaft besitzt, dass das Objekt auf einer Seite der Geraden

Abbildung 11.8 Darstellung eines Polygons mit Hilfe von Eckpunkten und Mittelpunkt.

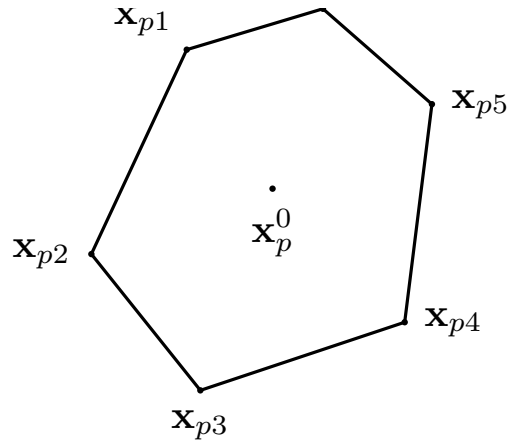
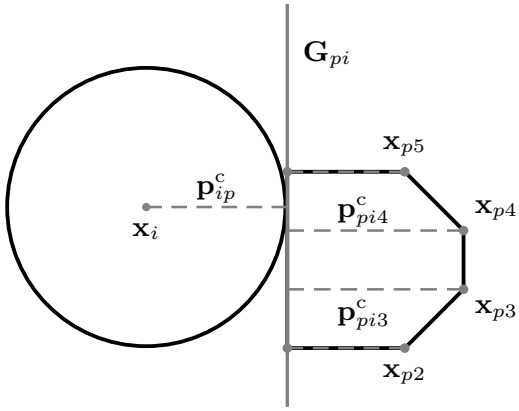
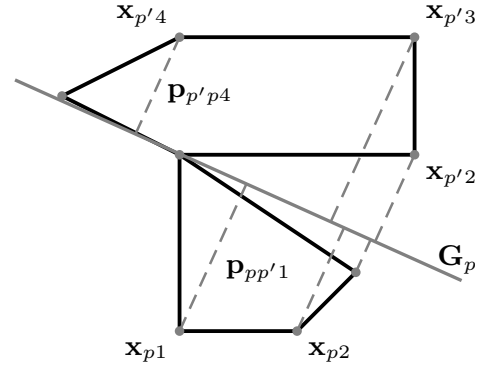


Abbildung 11.9 Trenngeraden bei Polygonen sowie zwischen Polygonen und Kreisen.



(a) Kreis-Polygon



(b) Polygon-Polygon

liegt. Für je zwei konvexe Objekte, die bis auf Berührungspunkte überlappungsfrei sind, kann daher mindestens eine Gerade (in höheren Dimensionen, mindestens eine trennende Hyperebenen) gefunden werden, bei denen beide Objekte auf verschiedenen Seiten der Gerade liegen.

Bei Polygonen folgt aus der Konvexität beider Polygone, dass alle Eckpunkte von q und q' der trennenden Gerade liegen; siehe Fig. 11.9. Seien p und p' Polygone mit Eckpunktmengen K_p und $K_{p'}$; die Mengen K_p und $K_{p'}$ können verschiedene Kardinalität haben. Die Geraden $\mathbf{G}_{pp'}$, die die Polygone p und p' trennen, beinhalten die Variablen $\mathbf{g}_{pp'}$, $\mathbf{m}_{pp'}$, und $\lambda_{pp'}$ je Polygonkombination pp' und haben die Gestalt

$$\mathbf{G}_{pp'} := G_{pp'}(\lambda) = \mathbf{g}_{pp'} + \mathbf{m}_{pp'}\lambda_{pp'} \quad ; \quad \forall\{p, p'|p' > p\} \quad , \tag{11.3.10}$$

wobei $\lambda \in \mathbb{R}$ die Gerade parametrisiert. Der Richtungsvektor $\mathbf{m}_{qq'}$ ist auf 1 normalisiert, d. h.

$$\mathbf{m}_{pp'}^2 = 1 \quad ; \quad \forall\{p, p'|p' > p\} \quad . \tag{11.3.11}$$

Diese Bedingung ist problematisch, da sie den nichtkonvexen Charakter des Problems verstärkt. Allerdings ist (11.3.11) hilfreich, da wir infolge der Normalisierung den Normalenvektor $\mathbf{n}_{pp}$ bezüglich $\mathbf{G}_{pp'}$ leicht durch

$$(n_{pp'1}, n_{pp'2})^{\mathrm{T}} = (m_{pp'2}, -m_{pp'1})^{\mathrm{T}} \quad ; \quad \forall\{p, p'|p' > p\} \tag{11.3.12}$$

berechnen können. Die K_p Verbindungsvektoren $\mathbf{p}_{pp'k}$ von $\mathbf{G}_{pp'}$ zum Eckpunkt V_{pk} von Polygon p sind durch

$$\mathbf{p}_{pp'k} = \mathbf{x}_{pk} - (\mathbf{g}_{pp'} + \mathbf{m}_{pp'd}\lambda_{pp'k}) \quad ; \quad \forall\{p, p', k|p' > p \wedge k \leq K_p\} \quad , \tag{11.3.13}$$

gegeben, während für die $K_{p'}$ Eckpunkte $V_{p'k}$ des Polygons p' die Verbindungsvektoren gemäß

$$\mathbf{p}_{p'pk} = \mathbf{x}_{p'k} - (\mathbf{g}_{pp'} + \mathbf{m}_{pp'd}\lambda_{p'pk}) \quad ; \quad \forall\{p, p', k|p' > p \wedge k \leq K_{p'}\}$$

berechnet werden. Die Hilfsvariablen $\lambda_{pp'k}$ und $\lambda_{p'pk}$ werden benötigt, um die Fußpunkte $\mathbf{p}_{pp'k}$ und $\mathbf{p}_{p'pk}$ zu berechnen. Die beiden Polygone p und p' werden getrennt durch die Bedingungen von Parallelität

$$\mathbf{p}_{pp'k} = \Delta_{pp'k}\mathbf{n}_{pp'} \quad ; \quad \forall\{p, p', k|p' > p \wedge k \leq K_p\} \quad , \tag{11.3.14}$$

und Anti-Parallelität

$$\mathbf{p}_{p'pk} = -\Delta_{p'pk}\mathbf{n}_{pp'} \quad ; \quad \forall\{p, p', k|p' > p \wedge k \leq K_{p'}\} \quad , \tag{11.3.15}$$

wobei die Variablen $\Delta_{pp'k}$ und $\Delta_{p'pk}$ die Abstände der Eckpunkte von der Trenngeraden messen.

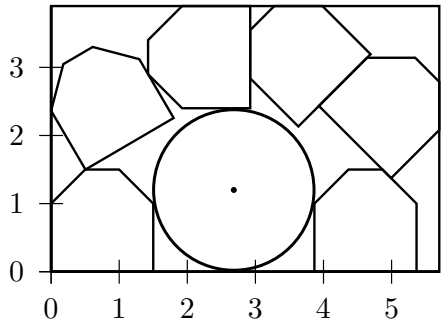
Um sicherzustellen, dass Polygone sich nicht mit Kreisen schneiden, ersetzen wir in (11.3.11) bis (11.3.15) Polygon p' durch den Kreis i und nehmen die folgenden Änderungen vor. Die Variablen $\Delta_{p'pk}$ in (11.3.15) werden auf den Radien R_i fixiert, d. h. die Trenngerade ist Tangente an den Kreis. $\Delta^{\mathrm{c}}_{pik}$ entspricht $\Delta_{pp'k}$ in (11.3.14), die nun die Gestalt

$$\mathbf{p}^{\mathrm{c}}_{pik} = \Delta^{\mathrm{c}}_{pik}\mathbf{n}_{pi} \quad ; \quad \forall\{p, i, k|k \leq K_p\} \quad , \tag{11.3.16}$$

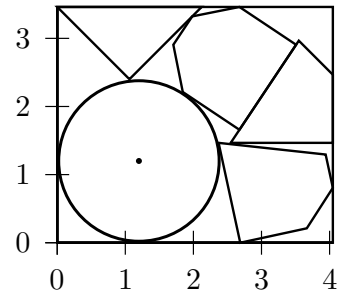
annimmt; (11.3.15) wird zu

$$\mathbf{p}^{\mathrm{c}}_{ip} = -R(i)\mathbf{n}_{ip} \quad ; \quad \forall\{i, p\} \quad . \tag{11.3.17}$$

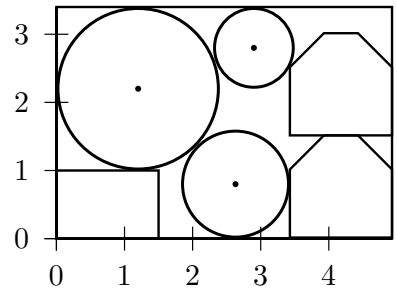
Abbildung 11.10 Zuschnittlösung für Kreise und Polygone. Alle Lösungen konnten zwar in kurzer Zeit berechnet werden, aber für keine gelang der Nachweis der Optimalität.



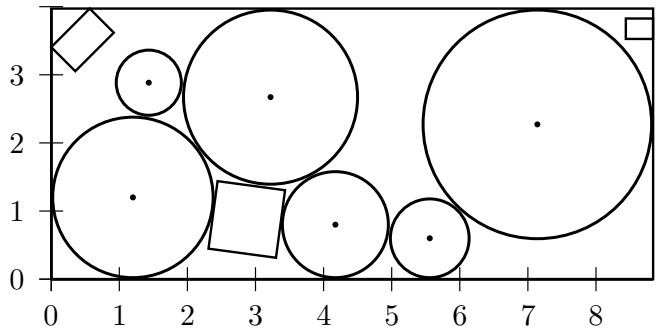
(a) c1p6-3



(b) c1p4M1



(c) c3p3



(d) c6p3

11.3.1.3 Zuschnittbedingungen für Ellipsen

Eine Ellipse i ist durch ihre große und kleine Halbachse a_i und b_i charakterisiert; sie ist vollständig durch ihr Zentrum $\mathbf{x}_i^0$ und ihre Orientierung θ_i, dem Winkel zwischen großer Halbachse und der x-Achse, beschrieben. Für $\theta_i = 0$ ist die Ellipse (der Index i wurde nicht mitgeführt) durch die quadratische Gleichung

$$\frac{(x-x_0)^2}{a^2} + \frac{(y-y_0)^2}{b^2} = 1 \tag{11.3.18}$$

definiert. Um zu einer allgemeinen Gleichungsdarstellung zu gelangen, notieren wir

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \mathsf{R}_\theta \begin{pmatrix} x-x_0 \\ y-y_0 \end{pmatrix} \quad , \quad \mathsf{R}_\theta := \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$$

mit Rotationsmatrix R_θ. Die Terme x' und y' der Komponentenschreibweise

$$\begin{aligned} x' &= (\cos\alpha)(x-x_0) + (-\sin\alpha)(y-y_0) \\ y' &= (\sin\alpha)(x-x_0) + (\cos\alpha)(y-y_0) \end{aligned}$$

setzen wir in die Ellipsengleichung (11.3.18) ein. Dies führt uns auf die allgemeine quadratische Form

$$(\mathbf{x}-\mathbf{x}_0)^{\mathrm{T}} \mathsf{A} (\mathbf{x}-\mathbf{x}_0) = 1 \quad , \tag{11.3.19}$$

einer rotierten Ellipse mit Zentrum $\mathbf{x}_0$ beliebiger Dimensions, wobei A eine positivdefinite Matrix ist und $\mathbf{x}$ bzw. $\mathbf{x}_0$ Vektoren sind.

Die Eigenvektoren von A sind die Hauptrichtungen der Ellipse und die Eigenwerte von A sind die reziproken, quadrierten Hauptachsen: a^{-2} und b^{-2}. Dabei kann A bzw. A_θ aus Rotationsmatrix und Eigenwerte wie folgt berechnet werden:

$$\mathsf{A}_\theta := \mathsf{R}_\theta \mathsf{D} \mathsf{R}_\theta^{\mathrm{T}} \tag{11.3.20}$$

mit

$$\mathsf{D} := \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix} = \begin{pmatrix} a^{-2} & 0 \\ 0 & b^{-2} \end{pmatrix} \quad , \tag{11.3.21}$$

und daher

$$\begin{aligned} \mathsf{A}_\theta &= \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix} \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix} \\ &= \begin{pmatrix} \cos^2\theta\lambda_1 + \sin^2\theta\lambda_2 & \cos\theta\sin\theta\lambda_1 - \cos\theta\sin\theta\lambda_2 \\ \cos\theta\sin\theta\lambda_1 - \cos\theta\sin\theta\lambda_2 & \cos^2\theta\lambda_2 + \sin^2\theta\lambda_1 \end{pmatrix} \quad . \end{aligned}$$

Möchte man wie bei den Polygonen trigonometrische Terme vermeiden, so kann man anstelle des Rotationswinkels θ als freie Variable auch $v := \cos\theta$ und $w := \sin\theta$ mit den Schranken $-1 \le v \le +1$ und $0 \le w \le +1$ als Variablen verwenden, wobei diese weiter durch die trigonometrisch Identität

$$v_p^2 + w_p^2 = 1 \quad ; \quad \forall\{p\} \tag{11.3.22}$$

verknüpft sind. Für Ellipsen mit der großen Halbachse als Symmetrieachse brauchen wir nur Winkel im Intervall von $0°$ bis $180°$ zu berücksichtigen, d. h. $0 \le w \le 1$.

Eine Bedingung abzuleiten, die erzwingt, eine Ellipse vollständig innerhalb des Rechtecks $\mathcal{R}$ zu halten, ist wesentlich schwieriger als bei Kreisen und Polygonen. Bezeichnen x_{id}^- und x_{id}^+ die minimale und maximale Ausdehnung von Ellipse i in Dimension d, so lauten die Ungleichungen zwar einfach

$$0 \leq x_{id}^- \leq x_{id}^+ \leq S_d^{\mathrm{P}} \quad ; \quad \forall\{i,d\} \quad . \tag{11.3.23}$$

Die Schwierigkeit liegt aber darin, einen geschlossenen Ausdruck für x_{id}^- und x_{id}^+ abzuleiten.

Wir berechnen nun x_{id}^+ und x_{id}^- (den Index i wollen wir der Einfachheit wegen dabei weglassen) als Maximal- und Minimalwert der Zielfunktion $c^{\mathrm{T}}x$

$$x_d^+ := \max c^{\mathrm{T}}x = \max x_d \quad ,$$

bzw.

$$x_d^- := \min c^{\mathrm{T}}x = \min x_d \quad ,$$

mit der Ellipsengleichung (11.3.19) als Nebenbedingung. Für den Dimensionsindex $d = 1$ wählen wir $c^{\mathrm{T}} := \{1, 0\}$, während für $d = 2$ $c^{\mathrm{T}} := \{0, 1\}$ der passende Vektor ist. Anstelle von (11.3.19) lösen wir allerdings das etwas einfachere Problem

$$x_d^+ := \max \mathbf{c}^{\mathrm{T}}(\mathbf{x} + \mathbf{x}_0) = x_{0d} + \max x_d \quad , \quad \forall d \in \mathcal{D} := \{1, 2\}$$

bzw.

$$x_d^- := \min \mathbf{c}^{\mathrm{T}}(\mathbf{x} + \mathbf{x}_0) = x_{0d} + \min x_d \quad , \qquad \forall d \in \mathcal{D} := \{1, 2\}$$

mit der vereinfachten Ellipsengleichung

$$\mathbf{x}^{\mathrm{T}}\mathsf{A}\mathbf{x} = 1 \quad , \tag{11.3.24}$$

die eine rotierte Ellipse im Ursprung beschreibt.

Wir lösen dieses beschränkte Optimierungsproblem nun, in dem wir die Lagrangefunktion

$$L(\mathbf{x}, \lambda) = \mathbf{c}^{\mathrm{T}}(\mathbf{x} + \mathbf{x}_0) + \lambda \left(\mathbf{x}^{\mathrm{T}}\mathsf{A}\mathbf{x} - 1\right) \tag{11.3.25}$$

mit Lagrangemultiplikator λ aufstellen. Die notwendigen Bedingungen sind die KKT-Bedingungen

$$\mathbf{c} + 2\lambda\mathsf{A}^{\mathrm{T}}\mathbf{x} = 0 \tag{11.3.26}$$

und

$$\mathbf{x}^{\mathrm{T}}\mathsf{A}\mathbf{x} = 1 \quad . \tag{11.3.27}$$

Multiplizieren wir (11.3.26) mit $\mathbf{x}^{\mathrm{T}}$ (diese Operation ist erlaubt, da das Zentrum der Ellipse nie der Maximal- oder Minimalpunkt sein kann), so erhalten wir mit (11.3.27)

$$x_d + 2\lambda = 0 \quad ,$$

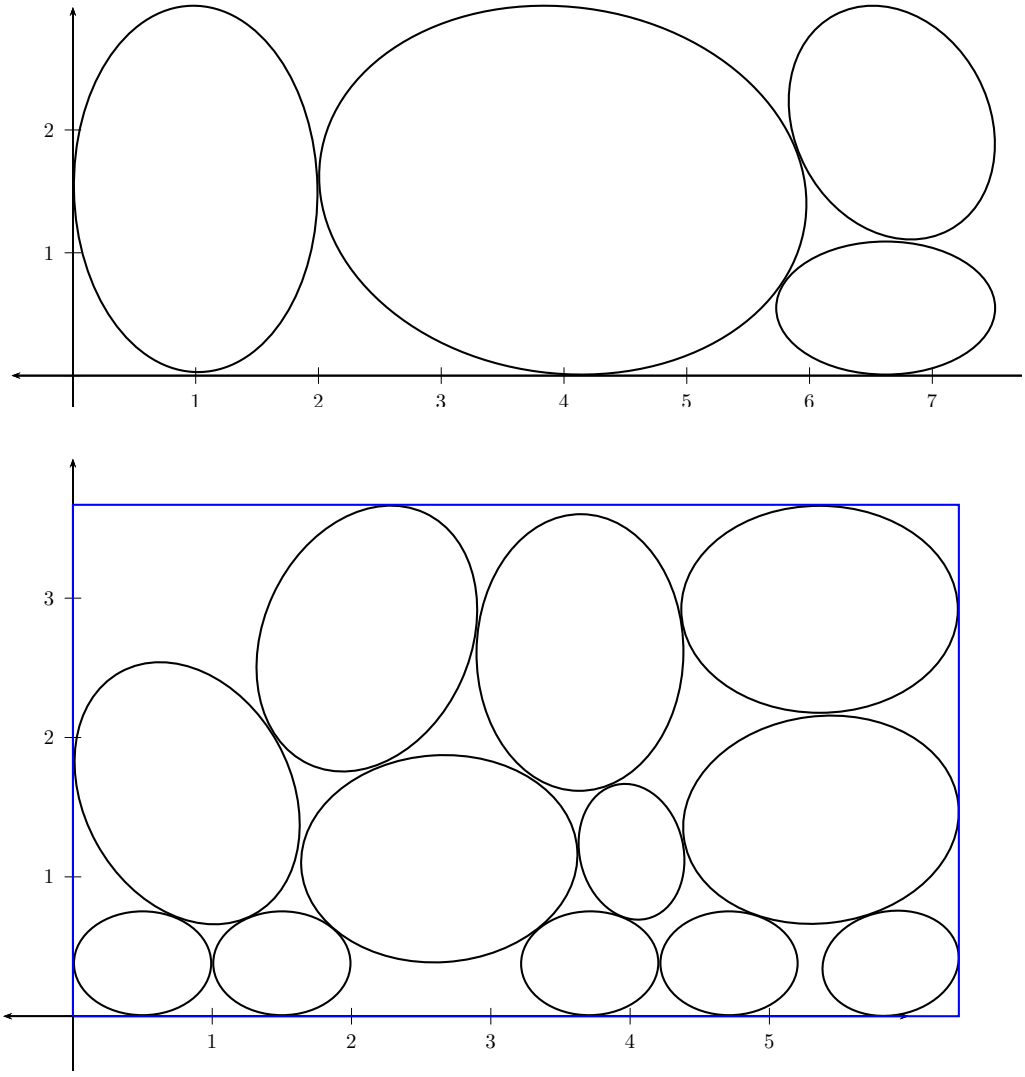
was uns ermöglicht, λ aus (11.3.26) zu eliminieren, was schließlich auf

$$\mathbf{c} - x_d\mathsf{A}^{\mathrm{T}}\mathbf{x} = 0 \tag{11.3.28}$$

führt. Zunächst führen wir eine detaillierte Analyse für $d = 1$ durch und erhalten

$$\begin{aligned} 1 - x_1\left(A_{11}x_1 + A_{21}x_2\right) &= 0 \\ -x_1\left(A_{12}x_1 + A_{22}x_2\right) &= 0 \quad . \end{aligned}$$

Abbildung 11.11 Zuschnittlösungen für 4 und 12 Ellipsen. Das Bild oben zeigt die optimale Lösung füer 4 Ellipsen. Im Bild unten ist eine zulässige Lsung für 12 Ellipsen gezeigt; der Optimalitätsnachweis steht noch aus.



Wegen $x_1 \neq 0$ (das Zentrum der Ellipse kann nie ein stationärer Punkt der KKT-Bedingungen sein), dividieren wir die zweite Gleichung durch x_1 und folgern

$$x_2 = -\frac{A_{12}}{A_{22}} x_1 \quad .$$

Hieraus folgt weiter

$$x_1^2 = \left(A_{11} - A_{21} \frac{A_{12}}{A_{22}} \right)^{-1} = \frac{A_{22}}{\lambda_1 \lambda_2} = A_{22} a^2 b^2 \quad ,$$

wobei wir die aus der Eigenwertzerlegung resultierende Eigenschaft

$$\det \mathsf{A} = A_{11} A_{22} - A_{12} A_{21} = \lambda_1 \lambda_2$$

und schließlich

$$\begin{aligned} x_1^+ &= \max \mathbf{c}^{\mathrm{T}} (\mathbf{x} + \mathbf{x}_0) = x_{01} + \sqrt{x_1^2} = x_{01} + ab\sqrt{A_{22}} \\ &= x_{01} + \sqrt{a^2 \cos^2(\theta_j - \omega) + b^2 \sin^2(\theta_j - \omega)} \end{aligned}$$

bzw.

$$\begin{aligned} x_1^- &= \min \mathbf{c}^{\mathrm{T}} (\mathbf{x} + \mathbf{x}_0) = x_{01} - \sqrt{x_1^2} = x_{01} - ab\sqrt{A_{22}} \\ &= x_{01} - \sqrt{a^2 \cos^2(\theta_j - \omega) + b^2 \sin^2(\theta_j - \omega)} \end{aligned}$$

erhalten. In ähnlicher Weise folgt für $d = 2$

$$\begin{aligned} -x_2 \left(A_{11} x_1 + A_{21} x_2 \right) &= 0 \\ 1 - x_2 \left(A_{12} x_1 + A_{22} x_2 \right) &= 0 \end{aligned}$$

und daraus

$$x_2^2 = \left(A_{22} - A_{12} \frac{A_{21}}{A_{11}} \right)^{-1} = \frac{A_{11}}{\lambda_1 \lambda_2} = A_{11} a^2 b^2 \quad ,$$

und schließlich

$$x_2^+ = \max \mathbf{c}^{\mathrm{T}} (\mathbf{x} + \mathbf{x}_0) = x_{02} + \sqrt{x_2^2} = x_{02} + ab\sqrt{A_{11}} \tag{11.3.29}$$

$$= x_{02} + \sqrt{b^2 \cos^2(\theta_j - \omega) + a^2 \sin^2(\theta_j - \omega)} \tag{11.3.30}$$

bzw.

$$x_2^- = \min \mathbf{c}^{\mathrm{T}} (\mathbf{x} + \mathbf{x}_0) = x_{02} - \sqrt{x_2^2} \tag{11.3.31}$$

$$= x_{02} - \sqrt{b^2 \cos^2(\theta_j - \omega) + a^2 \sin^2(\theta_j - \omega)} \quad . \tag{11.3.32}$$

Mit diesen Formeln für die minimale und maximale Ellipsenausdehnungen x_d^- und x_d^+ erhalten wir die korrekte Einbettung einer Ellipse in das Rechteck $\mathcal{R}$. Wir verwenden diese Resultate im Folgenden, um die Überlappungsfreiheit zweier Ellipsen mit Hilfe trennender Geraden sicher zustellen. Betrachten wir dazu also wieder eine um den Winkel θ rotierte Ellipse sowie eine durch

$$\mathbf{G}(t) := \mathbf{g}_0 + \mathbf{g}t$$

parametrisierte Gerade, die durch den Punkt $\mathbf{g}_0$ führt und den normalisierten Richtungsvektor $\mathbf{g}$ besitzt mit

$$|\mathbf{g}| = 1 \quad . \tag{11.3.33}$$

Können wir eine notwendige Bedingung dafür angeben, dass die Ellipse vollständig auf einer Seite von $\mathbf{G}(t)$ liegt oder $\mathbf{G}(t)$ gerade berührt? Wir werden sehen, dass die möglich ist. Beginnen wir dabei mit $\mathbf{G}(t)$ und berechnen den Neigungswinkel ω

$$\cos\omega := (1,0)\mathbf{g} = g_1 \tag{11.3.34}$$

mit der x-Achse. Falls $\mathbf{g}$ nicht parallel zur x-Achse ist, existiert ein Schnittpunkt x_h; andernfalls brauchen wir die nachfolgenden Überlegungen nicht zu bemühen, da bei Parallelität $\omega = 0$ und somit gar keine Neigung vorliegt.

Wir wollen uns gedanklich nun vorstellen, dass wir das Koordinatensystem so transformieren (verschieben und drehen), dass $(x_h, 0)$ der Ursprung eines neuen Koordinatensystems wird, in dem $\mathbf{G}(t)$ nun identisch mit der neuen x-Achse ist. Wir brauchen nun lediglich die Ellipse in dem neuen Koordinatensystem darzustellen und die obigen Formeln für die minimale Ausdehnung $x_{id}^{-\prime}$ der Ellipse im neuen Koordinatensystem anzuwenden. Falls $x_{i2}^{-\prime} \geq 0$, liegt Ellipse i oberhalb der Geraden (oder berührt sie). Damit lautet die notwendige Bedingung für *oberhalb* also $x_{i2}^{-\prime} \geq 0$.

Zur Ableitung der Transformation bezeichne t_0 den Wert von t, für den $\mathbf{G}(t)$ die x-Achse schneidet, also $G_2(t_0) = 0$. In den neuen, verschobenen und rotierten Koordinatensystem sind die Zentrumskoordinaten daher durch den Vektor

$$\mathbf{v}_0 := \begin{pmatrix} x_{01} - G_1(t_0) \\ d_0 \end{pmatrix}$$

gegeben, wobei d_0 bzw. im konkreten Fall d_{0ij}, den Abstand von Ellipse i zur $\mathbf{G}(t)$ bzw. $\mathbf{G}_{ij}(t)$, die Ellipse i und Ellipse j trennen soll, bezeichnet, d. h.

$$d_{0ij} := \frac{(-g_{2ij}, g_{1ij})(\mathbf{x}_{0i} - \mathbf{g}_{0ij})}{|\mathbf{g}_{ij}|} = (-g_{2ij}, g_{1ij})(\mathbf{x}_{0i} - \mathbf{g}_{0ij}) \quad . \tag{11.3.35}$$

Hierbei haben wir mit $(-g_2, g_1)^{\mathrm{T}}$ einen zu $\mathbf{g}$ orthonormalen Vektor konstruiert. Im neuen Koordinatensystem können wir den Umfang der Ellipse durch

$$\mathbf{v}(\varphi) = \mathbf{v}_0 + R_{\theta-\omega} \begin{pmatrix} A\cos\varphi \\ B\sin\varphi \end{pmatrix} \quad , \quad 0 \leq \varphi \leq 2\pi \quad ,$$

erzeugen; Punkte des Umfangs genügen der Ellipsengleichung

$$(\mathbf{v} - \mathbf{v}_0)\mathsf{E}(\mathbf{v} - \mathbf{v}_0)^{\mathrm{T}} = 1 \quad , \quad \mathsf{E} := \mathsf{R}_{\theta-\omega}\mathsf{D}\mathsf{R}_{\theta-\omega}^{\mathrm{T}} \quad . \tag{11.3.36}$$

Der Winkel $\theta - \omega$ zwischen der rotierten x_1-Achses und der Geraden ist durch

$$\cos(\theta - \omega) = (\cos\theta, \sin\theta)\mathbf{g} \tag{11.3.37}$$

gegeben. Mit Hilfe von Formel (11.3.31) erhalten wir

$$v_2^- = v_{02} - \sqrt{v_2^2} \quad , \quad v_2^2 = \left(E_{22} - E_{12}\frac{E_{21}}{E_{11}}\right)^{-1} = \frac{E_{11}}{\lambda_1\lambda_2} = E_{11}a^2b^2 \quad , \tag{11.3.38}$$

und

$$v_2^- = v_{02} - \sqrt{v_2^2} = v_{02} - ab\sqrt{E_{11}} = v_{02} - ab\sqrt{\cos^2(\theta - \omega_{ij})\lambda_1 + \sin^2(\theta - \omega_{ij})\lambda_2} \quad . \tag{11.3.39}$$

Die Transformation führt zwar zu sehr nichtlinearen Ausdrücken, aber die Bedingung dafür, dass sich Ellipse i oberhalb der Trenngeraden befindet ist einfach

$$v_{02i} - a_i b_i \sqrt{\cos^2(\theta_i - \omega_{ij})\lambda_{1i} + \sin^2(\theta_i - \omega_{ij})\lambda_{2i}} \geq 0 \quad ,$$

bzw.

$$v_{02i}^2 - \left[b_i^2 \cos^2(\theta_i - \omega_{ij}) + a_i^2 \sin^2(\theta_i - \omega_{ij})\right] \geq 0 \quad \wedge \quad v_{02i} \geq 0 \quad . \tag{11.3.40}$$

Die Bedingung dafür, dass Ellipse j unterhalb der Trenngeraden liegt, lautet

$$v_{02j} + a_j b_j \sqrt{\cos^2(\theta_j - \omega_{ij})\lambda_{1j} + \sin^2(\theta_j - \omega_{ij})\lambda_{2j}} \leq 0 \quad ,$$

bzw.

$$-v_{02j}^2 + \left[b_j^2 \cos^2(\theta_j - \omega_{ij}) + a_j^2 \sin^2(\theta_j - \omega_{ij})\right] \leq 0 \quad \wedge \quad v_{02j} \leq 0 \quad . \tag{11.3.41}$$

Für n Ellipsen haben wir also $n(n-1)/2$ Ungleichungen vom Typ (11.3.40) und (11.3.41). Die Modellformulierung wird vervollständigt durch

$$v_{02i} = d_{0i} \quad , \quad v_{02j} = d_{0j}$$

für jede Ellipse i und j sowie der Berechnung von ω_{ij} gemäß (11.3.34).

Zu beachten ist, dass v_{02} nur von d_0 abhängt, nicht aber vom Schnittpunkt mit der x-Achse. Daher gelten die notwendigen und hinreichenden Bedingungen dafür, dass eine Ellipse oberhalb oder unterhalb der Trenngeraden liegt auch für den Fall, dass die Trenngerade parallel zur x-Achse liegt.

11.3.2 Problemstruktur und Symmetrie

Die oben formulierten Zuschnittprobleme stellen sich als NLP-Probleme mit den folgenden nichtkonvexen Aspekten dar:

1. Zunächst die bilineare Zielfunktion (11.3.1). Bei Fixierung von Länge oder Breite von $\mathcal{R}$ reduziert sich dies auf eine lineare Zielfunktion.
2. Die Bedingung der Überlappungsfreiheit führt auf eine geometrische Situation, aus der offensichtlich ein nichtkonvexer Bereich resultiert. Um dies zu verstehen, stelle man sich $\mathcal{R}$ mit einem bereits fixierten Objekt i_f vor. Der zulässige Bereich der übrigen Objekte $i \in I \backslash \{i_f\}$ bezüglich i_f ist $\mathcal{R}$ ohne den von i_f überdeckten Bereich. Handelt es sich bei den Objekten ausschließlich um Kreise, so ist (11.3.2) die entscheidende Ungleichung mit quadratischen und bilinearen Termen. Auch bei Polygonen und Ellipsen lässt sich das Problem komplett mit Hilfe quadratischer Gleichungen oder Ungleichungen formulieren. Die Normalisierungsgleichungen (11.3.11) und (11.3.33) sind dabei besonders problematisch.

Da sämtliche nichtkonvexen Terme quadratischer Natur sind, ist es nicht verwunderlich, dass der Solver GLOMIQO besonders gut bei dieser Problemklasse abschneidet und bei den meisten Problemen der einzige Globalsolver ist, der in der Lage ist die Lücke zwischen oberer und unterer Schranke zu schließen, d. h. wirklich das globale Optimum zu berechnen.

Neben der Nichtkonvexität sind die vorgestellten Probleme aber noch wegen ihres verdeckten kombinatorischen Charakters schwierig. Kombinatorischen Problemen sind wir bisher nur im Zusammenhang mit gemischt-ganzzahligen Problemen begegnet. Beim Problem des Handlungsreisenden resultiert die kombinatorische Struktur aus den möglichen Permutationen für die Reihenfolge der zu besuchenden Städte. Die Situation ist hier ähnlich, was sich wie folgt veranschaulichen lässt. Angenommen, man allokiert jedes Objekt i mit den Perlen einer Perlenkette, die dann durch das Rechteck gelegt wird bzw. sich durch $\mathcal{R}$ *schlängelt*. Durch den Weg bzw. dieser *Schlängelung*, den die Perlenkette durch $\mathcal{R}$ nimmt, können die Objekte viele relative Positionen zueinander einnehmen, d. h. Objekt j kann links, unterhalb, rechts oder oberhalb von Objekt i positioniert sein. Hinzu kommt bei Polygonen und Ellipsen noch die Orientierung. So betrachtet bestehen die Freiheitsgrade in der Reihenfolge der Objekte entlang der Kette, die Positionen auf der Kette und in dem Weg, den die Kette durch $\mathcal{R}$ nimmt. Die Reihenfolge könnte zwar prinzipiell durch vollständige Enumerierung behandelt werden, dies wird aber praktisch wohl nur für kleine Instanzen möglich sein. Die Positionen auf der Kette, dargestellt durch die Zentren der Objekte, sowie die Orientierung wird im NLP-Modell abgebildet. Der Weg der Kette durch $\mathcal{R}$ stellt ein schwieriges Problem an sich dar.

Ein Teil der Schwierigkeiten des Problems resultiert auch aus der Symmetrie. Die Symmetrie kann ein wenig reduziert werden, indem man das Zentrum *eines* beliebigen Objektes in den ersten Quadranten von $\mathcal{R}$ platziert. Wählen wir beispielsweise den Kreis i_*, so bedeutet dies

$$x_{i_* d} \leq \frac{1}{2} x_d^{\mathrm{P}} \quad ; \quad \forall \{d\} \quad . \tag{11.3.42}$$

Bei einem Polygon p_*, wirkt sich die Ungleichung

$$x_{p_* d}^0 \leq \frac{1}{2} x_d^{\mathrm{P}} \quad ; \quad \forall \{d\} \tag{11.3.43}$$

sehr positiv auf die Rechenzeit aus.

Symmetrieentartung durch die Anwesenheit kongruenter Objekte, also solcher, die in Form und Fläche identisch sind, kann reduziert werden, in dem man diese derselben Kongruenzklasse zuordnet I^{co}. Für Objekte i und i' in derselben Kongruenzklasse wenden wir die ordnenden Ungleichungen

$$x_{i1} + 5x_{i2} \leq x_{i'1} + 5x_{i'2} \quad ; \quad \forall \{(i,i') | i < i' \wedge I_i^{\mathrm{co}} = I_{i'}^{\mathrm{co}}\} \tag{11.3.44}$$

an; die Wahl der Koeffizienten ist dabei recht willkürlich.

Ein weiterer Entartungsgrad, der Probleme für einen Globalsolver verursachen kann, ist mit *freien Objekten* verbunden. Das sind solche, meist kleinere Objekte, die sich berührungsfrei zwischen den größeren *bewegen* können ohne dabei den Wert der Zielfunktion, d. h. die Fläche a des Rechtecks, zu ändern. Bei Zuschnittproblemen ist dies zwar unproblematisch; bei Packungsproblemen aber eher unerwünscht. Eine Möglichkeit, mit diesem Problem umzugehen, besteht darin, die Zentrumskoordinaten mit in die zu minimierende Zielfunktion aufzunehmen; dies führt dazu, dass solche Objekte möglichst weit links unten an den Koordinatenursprung platziert werden.

11.3.3 Einige Ergebnisse

In Kallrath (2009,[161]) sind Ergebnisse für Kreise und Polygone enthalten. Für bis zu 10 Kreisen konnte Optimalität bewiesen, für kleine Anzahlen von Polygonen ebenfalls. Die

von Misener & Floudas (2012,[213]) mit GLOMIQO erzielten Ergebnisse berechnen optimale Lösungen in wesentlich kürzerer Zeit. Die zeigt sich auch bei kleineren Problemen mit Ellipsen.

11.4 Optimale Verkaufsstrategien

Bei diesem Problem handelt es sich um ein nichtkonvexes, nichtlineares Minimierungsproblem mit stückweise-linearer Zielfunktion. Daher kann das globale Optimum mit einer MILP-Formulierung bestimmt werden. Das Ziel besteht darin, von drei Lieferanten, die ihre Güter zu verschiedenen, von der abgenommen Menge abhängigen Preisen anbieten, kostenminimal die benötigten Güter zu beziehen. Hierbei soll allerdings zur Vermeidung von Abhängigkeiten von jedem Lieferanten nicht mehr als eine zulässige Höchstmenge gekauft werden. Vom Lieferanten Wagner sollen z. B. nicht mehr als 40% bezogen werden; er bietet die folgende Preisstruktur mit Stufenrabatt [engl.: *all-units discounts*] an:

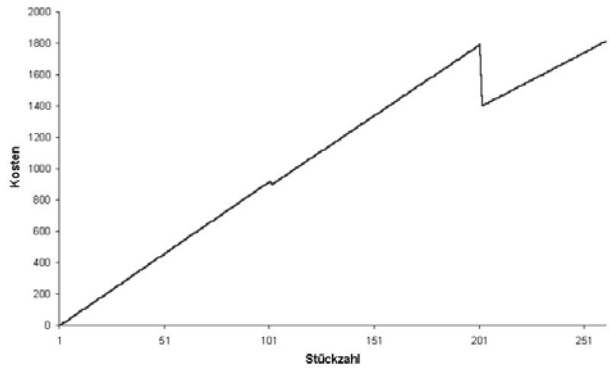
Spanne	0 - 100	101 - 200	201 - 1000
Preis	9.2	9	7
Sprungstelle	100	200	1000

$$C_w(x) := \begin{cases} 9.2x\ , & 0 \le x \le 100 \\ 9.0x\ , & 100 \le x \le 200 \\ 7.0x\ , & 200 \le x \le 1000 \end{cases}$$

Dies bedeutet, dass die gesamten Kosten für 150 Stück, die wir von Wagner beziehen, $150 \cdot 9 = 1350$ betragen. Die Abbildung 11.12 zeigen die Kosten $C_w(x)$ als Funktion der Stückzahl x bis zu einer Stückzahl von 260. Der Rabatt der erreichten Rabattstufe wird für die gesamte Liefermenge gewährt und die Gesamtkosten ergeben sich also als Produkt aus der gelieferten Menge und dem Preis der erreichten Rabattstufe.

Nachfolgend soll nun der etwas allgemeinere Fall mit N^S durch s indizierte Lieferanten und N^B Sprungstellen [engl.: *breakpoints*] b, d. h. Punkten, an denen der Einheitspreis sich ändert, betrachtet werden. Damit ergeben sich gestufte Preisabschläge, also Rabattstufen. Dieses Problem lässt sich mit den beiden nachfolgend vorgestellten Formulierungen beschreiben. Die erste Variante verwendet SOS-2-Mengen, die zweite, wesentlich elegantere und effizientere Formulierung dagegen SOS-1-Mengen.

Abbildung 11.12 Kosten gegen Stückzahl bis zu einer Stückzahl von 260.



Die auf SOS-2-Mengen aufbauende Formulierung benötigt neben den eigentlichen N^B Sprungstellen wie weiter unten aufgeführt aus numerischen Gründen weitere $N = 2N^B$ numerische Sprungstellen. Mit den Indizes

$$\begin{array}{lcl} b \in \mathcal{B} = \{1, \ldots, N^B\} & : & \text{die Menge der Sprungstellen} \\ k \in \mathcal{K} = \{1, \ldots, N\} & : & \text{die Menge der erweiterten Sprungstellen} \\ s \in \mathcal{S} = \{1, \ldots, N^S\} & : & \text{die Menge der Lieferanten} \end{array}$$

lassen sich die relevanten Daten darstellen als

$$\begin{array}{lcl} B_{sb} & , & \text{die mit Sprungstelle } b \text{ assoziierte Produktmenge von Lieferant } s \\ C_{sk} & , & \text{gesamte, an Lieferant } s \text{ zu zahlende Kosten an Sprungstelle } k \\ \Delta & , & \text{Faktor, der die Gitterspreizung beschreibt} \\ \Delta_k & , & \text{rekursiv definierter Gitterspreizungsfaktor an Sprungstelle } k \\ P_{sb} & , & \text{Preis vom Lieferanten } s \text{ zwischen den Sprungstellen } b-1 \text{ und } b \\ P^U_{sb} & , & \text{max. prozentualer Bezug von Lieferant } s \text{ zwischen } b-1 \text{ und } b \\ R & , & \text{die gesamt benötigte Stückzahl} \\ X_{sk} & , & \text{erweiterte Gitterpunkte an Sprungstelle } k \text{ von Lieferant } s \end{array} .$$

Infolge der Stufenrabatte verhalten sich Kosten hinsichtlich der gekauften Menge unstetig, da die Einheitspreise nur abschnittsweise konstant sind. Damit liegt eine besondere Art der Nichtlinearität vor. In Abschnitt 5.7.2 wurden SOS-2-Mengen zur Modellierung nichtlinearer Funktionen eingeführt; hier wird deutlich, wie dies in der Praxis Anwendung findet. Zur Behandlung der Unstetigkeit führen wir den Parameter Δ ein, der den Abstand zu einer Sprungstelle angibt und erlaubt, weitere Daten

$$\Delta_1 = 0 \quad , \quad \Delta_2 = -\Delta \quad , \quad \Delta_k = -\Delta_{k-1} \quad , \quad k \in \{3, \ldots, N\}$$

rekursiv abzuleiten. Typischerweise kann $\Delta = 1$ sein, was rekursiv zu der Δ_k-Folge $0, -1, 1, -1, \ldots, -1$ führt. Darauf aufbauend und mit Hilfe der Definitionen[1]

$$k_1 = \left\lfloor \frac{k+1}{2} \right\rfloor \quad , \quad k_2 = \left\lfloor \frac{k}{2} \right\rfloor$$

lassen sich die zusätzlichen *numerischen Sprungstellen* des erweiterten Gitters

$$X_{s1} = 0 \quad , \quad X_{sk} = B_{sk_2} + \Delta_k \quad , \quad k \in \{2, \ldots, N\}$$

rekursiv berechnen. Im obigen Beispiel ergeben sich die Zahlen

X_{s1}	X_{s2}	X_{s3}	X_{s4}	X_{s5}	X_{s6}
0	99	101	199	201	999

.

Anschließend berechnen wir die gesamten Kosten an den ursprünglichen Sprungstellen bei *Annährung von links und von rechts*

$$C_{s1} = 0 \quad , \quad C_{sk} = P_{sk_1} B_{sk_2} \quad , \quad \forall k \in \{2, \ldots, N\} \quad .$$

[1] Der Ausdruck $\lfloor x \rfloor$ bezeichnet die größte ganzzahlige Zahl, die x nicht überschreitet. Ist x eine positive Zahl, so gibt $\lfloor x \rfloor$ gerade den ganzzahligen Teil von x an, z. B. $\lfloor 4.7 \rfloor = 4$.

Hierzu ein Beispiel: C_{s2} gibt unter Annahme des Einheitspreises 9.2 die Kosten von 920 an der Sprungstelle $B_{s1} = 100$ wenn wir uns von links nähern; C_{s3} ergibt den rechtsseitigen Grenzwert, der sich aus dem Einheitspreis von 9.0 ergibt. Zu beachten ist, dass unsere Eingangsdaten die Dimension *Geld/Stück* haben, C_{sk} dagegen die gesamten Kosten repräsentiert und daher die Dimension *Geld* hat. Die Auswertung der Preisdaten des Lieferanten Wagner ergibt die folgenden Gesamtkosten an den Sprungstellen:

C_{s1}	C_{s2}	C_{s3}	C_{s4}	C_{s5}	C_{s6}
0	920	900	1800	1400	7000

.

Das Modell beinhaltet nun die kontinuierlichen Variablen $x_s \geq 0$, die die vom Lieferanten s bezogene Menge beschreiben, sowie die SOS-2-Variablen $\lambda_{ks} \geq 0$, die der Konvexitätsbedingung

$$\sum_{k=1}^{N} \lambda_{ks} = 1 \quad , \quad \forall s \in \mathcal{S} \tag{11.4.45}$$

unterliegen und mit den Variablen x_s durch

$$x_s = \sum_{k=1}^{N} X_{sk} \lambda_{ks} \quad , \quad \forall s \in \mathcal{S} \tag{11.4.46}$$

verknüpft sind. Die Gleichung (11.4.46) ist sehr gut als Referenzbedingung geeignet, da die Sprungstellen X_{sk} eine sinnvolle Monotoniebeziehung im Problem etablieren.

Mit diesen Variablen lässt sich nun zunächst die Zielfunktion

$$\min \quad \sum_{s=1}^{N^S} \sum_{k=1}^{N} C_{sk} \lambda_{ks} \tag{11.4.47}$$

formulieren. Weiter muss der Bedarf R befriedigt werden, d. h.

$$\sum_{s=1}^{N^S} x_s \geq R \tag{11.4.48}$$

und schließlich müssen die maximalen prozentualen Bezugsmengen beachtet werden, d. h.

$$x_s \leq P_s^U \frac{R}{100} \quad , \quad \forall s \in \mathcal{S} \quad . \tag{11.4.49}$$

Damit ist die Beschreibung des Modells abgeschlossen.

Die auf SOS-1-Mengen basierende Formulierung hat dagegen eine völlig andere Struktur. Hierin selektieren die durch

$$\delta_{sb} = \begin{cases} 1 \text{ , wenn } B_{sb-1} \leq y_{sb} \leq B_{sb} \\ 0 \text{ , sonst} \end{cases} \quad , \quad \begin{matrix} \forall s \in \mathcal{S} \\ \forall b \in \mathcal{B} \end{matrix}$$

definierten Binärvariablen δ_{sb}, in welcher Rabattstufe wir uns aufhalten. Diese Definition führt im Falle $B_{s0} = 0$ zu der folgenden Verknüpfung

$$x_s = \sum_{b=1}^{N^B} y_{sb} \quad , \quad \forall s \in \mathcal{S} \tag{11.4.50}$$

zwischen den nichtnegativen kontinuierlichen Variablen y_{sb} und x_s. Die Binärvariablen δ_b bilden eine SOS-1-Menge mit der Referenzbeziehung

$$\sum_{b=1}^{N^B} B_{sb}\delta_{sb} \quad , \quad \forall s \in \mathcal{S}$$

und unterliegen der Konvexitätsbedingung

$$\sum_{b=1}^{N^B} \delta_{sb} = 1 \quad , \quad \forall s \in \mathcal{S} \quad .$$

Diese Binärvariablen stellen sicher, dass für jeden Lieferanten s höchstens eine der N^B Variablen y_{sb} von Null verschieden ist. Damit wird ersichtlich, warum (11.4.50) zulässig ist und durch die beiden Ungleichungen

$$B_{sb-1}\delta_{sb} \leq y_{sb} \leq B_{sb}\delta_{sb} \quad , \quad \forall s \in \mathcal{S} \quad , \quad \forall b \in \mathcal{B}$$

erzwungen wird. Vervollständigend treten (11.4.48)-(11.4.49) hinzu sowie die Zielfunktion

$$\min \quad \sum_{s=1}^{N^S} \sum_{b=1}^{N^B} P_{sb}x_{sb} \quad .$$

Bewertend sei festgehalten, dass die zweite Modellformulierung nicht nur eleganter und einfacher verständlich, sondern auch aus mathematischen Gründen vorzuziehen ist und das eigentliche Problem exakt repräsentiert, wogegen die Einführung des Parameters Δ in der ersten Variante eine Ungenauigkeit in das Modell einbringt. Es zeigt aber wiederum einmal mehr, dass die Modellformulierung nicht immer eindeutig ist und je nach Ausgangspunkt sogar zu sehr unterschiedlichen Formulierungen führen kann.

12 Schlussbetrachtungen und Ausblick

In diesem Kapitel schließen wir mit einigen Schlussbetrachtungen zu dem in diesem Buch vorgestellten Material, einer Diskussion über die Möglichkeiten der parallelen Optimierung und einem Ausblick über die Zukunft der gemischt-ganzzahligen Optimierung.

12.1 Lernenswertes aus den Fallstudien

Die in diesem Buch vorgestellten Modellformulierungen bieten dem Modellierer Bausteine für die Entwicklung umfassender Modelle z. B. im Supply Chain Management. In der Lage zu sein, vollständige, genaue und optimale Lösungen großer Planungsprobleme zu berechnen, bietet das Potential, enorme Kosten einzusparen, die Effizienz zu erhöhen, und sorgfältig mit limitierten Ressourcen umzugehen. Erfahrungen mit dem Einsatz der mathematischen Optimierung führen zu vorsichtigen Ersparnisschätzungen von etwa 3-4%. Die Fallstudien können dieses Potential nur andeuten, da es sich aus didaktischen Gründen oder aus Gründen der Vertraulichkeit meist um etwas vereinfachte und verkleinerte Modelle handelt. Dennoch sollte der Leser von der Lektüre des Buches in folgender Weise profitiert haben: die Fallstudien

- bieten ein breites Spektrum von realen Problemen aus verschiedenen Anwendungsbereichen und Industriezweigen und haben hoffentlich einen Eindruck vermittelt, welche Probleme sich mit Methoden der Optimierung behandeln lassen;
- enthalten einige besondere Formulierungen und Feinheiten, welche die Modellformulierung erheblich verbessern und die in der Trickkiste des Modellierers vorhanden sein sollten;
- zeigen den Bedarf an sorgfältigen, manchmal sehr speziellen, auf das jeweilige Problem angepassten Lösungstechniken;
- enthalten für andere Probleme relevante Strukturen und Unterstrukturen;
- können Bausteine oder Ausgangspunkte für komplexere Modelle sein.

Die Beispiele sollten verdeutlichen, dass Modellierung nicht unbedingt ein geradlinig einfacher, sondern eher ein iterativer Prozess mit einigen Reformulierungen ist. Manchmal stellt sich in der Praxis die Frage, ob man ein Modell mit der vom Kunden gewünschten Abbildungstreue entwickeln soll und dann eventuell Probleme bei dessen Lösung bekommt, oder ob man schon während der Modellierung mit berücksichtigen soll, was man für lösbar hält und dementsprechend sogleich die Abbildungstreue reduziert. Hier möchte ich mich ganz klar positionieren: Abbildungstreue ist wegen der Akzeptanz und Nähe zur Realität unverzichtbar und sollte höchste Priorität haben. Im Kapitel 8 wurden Wege aufgezeigt, auch sehr schwierige und große Probleme zu lösen. Sollten diese Möglichkeiten immer noch nicht reichen, um das Problem zu lösen, darf über eine Reduzierung der Abbildungstreue nachgedacht werden – vorher rate ich davon ab.

J. Kallrath, *Gemischt-ganzzahlige Optimierung: Modellierung in der Praxis*,
DOI 10.1007/978-3-658-00690-7_12, © Springer Fachmedien Wiesbaden 2013

12.2 Parallele Optimierung

In einigen Anwendungsfeldern ist die Rechengeschwindigkeit wesentlich für die Akzeptanz der Technologie und der berechneten Lösungen. In der Strömungsmechanik, Astrophysik oder Quantenchemie lässt sich durch Parallelisierung eine erhebliche Laufzeitverbesserung erzielen. Kann dies nicht auch im Umfeld der Optimierung eine Rolle spielen? Die Antwort ist sicherlich *ja*, wobei hier zwischen Parallelisierungsbemühungen des kombinatorischen Teils der Algorithmen (hier sind insbesondere nichtdeterministische Aspekte zu erwarten) und des Kernoptimierungsproblems (LP, NLP) zu unterscheiden ist.

Für gemischt-ganzzahlige *lineare* Optimierungsprobleme bieten kommerzielle Softwarepakete inzwischen standardmässig parallele Versionen für PC-Netzwerke oder Multiprozessorsysteme mit nahezu linearem Verhalten in der Anzahl der Prozessoren an. Während parallele Algorithmen und deren Implementierung vor etwa 10 Jahren noch eher ein Thema für Spezialisten war, gehören diese inzwischen zum *normalen* Alltag. Im Umfeld der nichtlinearen Optimierung mag dies im Augenblick noch ein wenig anders sein. Dies hat aber eher damit zu tun, dass die Algorithmen in diesem Bereich jünger und weniger ausgereift sind und noch nicht anhand vieler wirklich großer Probleme getestet wurden. Gerade bei der gemischt-ganzzahligen Optimierung sollte aber stets bedacht werden, dass mit parallelen Architekturen selbst bei linearem Verhalten das exponentielle Wachstum der B&B-Verfahren nicht in den Griff zu bekommen ist. In diesem Sinne hilft parallele Optimierungssoftware bei gutartigen Problemen, die Grenze dessen, was noch lösbar ist, weiter hinauszuschieben; sie ist aber nur bedingt ein Mittel, strukturell schwierige Probleme, z. B. Schedulingprobleme, zu lösen. In der gemischt-ganzzahligen *nichtlinearen* Optimierung, wenn man nicht schon Methoden der globalen Optimierung verwendet, kann man von verschiedenen Startpunkten ausgehen.

12.3 Zukünftige Entwicklungen

In vielen beschriebenen Problemen und ihren Modellformulierungen wird ersichtlich, dass sich inzwischen sehr große lineare Probleme lösen lassen. Beispiele dafür finden wir in der Prozessindustrie, in Raffinerien und bei Fluggesellschaften, also meist grossen Organisationen, aber auch in mittleren und kleineren Unternehmungen gibt es ein lohnendes Potential zur Kostenreduzierung und Effizienzerhöhung durch den Einsatz gemischtganzzahliger Optimierung. Infolge schnellerer Prozessoren und verbesserter Algorithmen lassen sich Planungsprobleme, die vor 10 Jahren vielleicht eine Nacht auf einem Firmenrechner an Rechenzeit erforderten, nun in weniger als einer Stunde, vielleicht auch in wenigen Minuten lösen, sodass ein interaktives Arbeiten möglich wird. Es ist zu erwarten, dass in einigen Jahren MINLP und Globale Optimierungstechniken für nichtkonvexe, nichtlineare Optimierungsprobleme eine ähnliche Rolle spielt, wie heutzutage MILP.

Mit Standard-Software wie z. B. `SAP APO` – siehe Abschnitt 3.5.1 – stehen dem Anwender leistungsstarke Werkzeuge für fest umrissene Problemstellungen zur Verfügung. Noch ist es nötig, dass diese Programme von Personal mit Kenntnissen in Optimierung und Modellierung benutzt werden, aber es ist denkbar, dass die Modelle und Algorithmen derartig robust werden, dass für bestimmte Teilaufgaben automatisierte Lösungen möglich werden. In der Erstellung von Modellen und Lösungsverfahren in dieser Qualität wird viel Erfahrung benötigt, ebenso wie im Umgang mit neueren Lösungstechniken wie

B&C zur Lösung von MILP-Problemen, Verfahren zur Lösung von MINLP-Problemen, oder der Möglichkeit, mehrere Lösungsverfahren in Kombination zu benutzen.

Neben der Robustheit ist zu erwarten, dass die verbesserten Möglichkeiten dazu führen, dass Modelle detailreicher und realistischer werden; auch dies ist ein weites Feld für den Modellierer, dessen Bedeutung infolge des erweiterten Methodenspektrums eher noch zunehmen wird. Ihm fällt die Aufgabe zu, zu unterscheiden zwischen nützlichen Details und übertriebener Präzision, die nicht im Einklang mit der Qualität der Eingangsdaten steht. Oft wäre es sinnvoll, bei strategischen Entscheidungen operative Aspekte zu berücksichtigen, z. B. bei der Verfahrensauslegung oder dem Kauf von Anlagen.

Verbesserte Lösungsverfahren sollten auch dahingehend eingesetzt werden, die Modelle realistischer hinsichtlich der Datenqualität zu gestalten. In diesem Buch wurden durchgehend deterministische Modelle diskutiert. Da die Daten jedoch Unsicherheiten unterworfen sind, über die möglicherweise Wahrscheinlichkeitsaussagen vorliegen, lassen sich bei LP-Problemen mit unsicheren Daten robuste Lösungen zu finden [33] oder die stochastische Optimierung als das Mittel der Wahl einsetzen. Selbst für MILP-Probleme existieren Lösungsverfahren zur Berücksichtigung von Daten mit stochastischen Unsicherheiten [siehe z. B. Schultz (1995,[263]), Carøe & Schultz (1999,[49])]. Wenngleich diese Ansätze noch Einschränkungen unterliegen und meist nur in Form universitärer Software implementiert wurden, liegen erfolgreiche Anwendungen dieser Techniken z. B. in der Energiewirtschaft [Gollmer *et al.* (2000,[110])] und der chemischen Industrie [Sand *et al.* (2000,[253]) und Engell *et al.* (2001,[80])] vor.

In unseren Betrachtungen haben wir wenig Rücksicht auf die Komplexität der betrachteten Probleme im Sinne der Komplexitätstheorie genommen. Diese Theorie, siehe z. B. Garey & Johnson (2000,[98]), hat sich als sehr nützlich erwiesen bei der Einteilung der Probleme in zwei Klassen: die der leichten und schweren Probleme[1]. Probleme, die in polynomialer Laufzeit gelöst werden, gehören der leichten Klasse an, die mit $\mathcal{P}$ bezeichnet wird. Der Aufwand, eine Matrix mit n Zeilen und Spalten zu invertieren, wächst mit n^3 an[2], also gehört dieses Problem der Klasse $\mathcal{P}$ an. Der Nachweis der Zugehörigkeit eines Problems zur Klasse $\mathcal{P}$ ist erbracht, wenn es gelingt, einen Algorithmus zu konstruieren, der dieses Problem in polynomialer Laufzeit löst. Bei Problemen, die nicht in polynomialer Zeit gelöst werden, steigt der Rechenaufwand meist exponentiell, z. B. mit 2^n, oder mit $n!$ an. Klee & Minty (1972,[178]) haben gezeigt, dass es LP-Probleme gibt, für die das Simplexverfahren mindestens exponentiell viele Iterationsschritte benötigt; die meisten praktischen LP-Problemen können dagegen mit dem Simplexverfahren in polynomialer Zeit gelöst werden. Der Nachweis, dass LP-Probleme dagegen der Klasse $\mathcal{P}$ angehören, gelang Khachian (1979,[176]) indem er zeigen konnte, dass einige Innere-Punkte-Methoden LP-Probleme in polynomialer Laufzeit lösen[3]; für die am meisten verwendeten Verfahren

[1] Eine Einteilung in die Klasse der *hoffnungsvoll schweren* und *hoffnungslos schweren* wäre vielleicht angemessener, denn die Rechenzeit bei den angeblich leichten, in polynomialer Zeit lösbaren Problemen, kann je nach den in diesem polynomialen Zusammenhang auftretenden Koeffizienten und Potenzen auch recht groß werden. Zudem sollte man bedenken, dass die Komplexitätstheorie sich mit der Bestimmung des Laufzeitverhaltens in Abhängigkeit von der Problemgröße beschäftigt. Es gibt aber durchaus viele Probleme, darunter die als $\mathcal{NP}$-vollständig oder $\mathcal{NP}$-schwer klassifizierten, die aber in der Praxis in kleinen Instanzen auftreten und mühelos mit B&B- oder B&C-Verfahren gelöst werden können. Bei hoffnungslos schweren Problem ist die Lage also je nach Problemgröße doch nicht hoffnungslos.

[2] Vorfaktoren oder Polynome niedrigerer Ordnung werden hierbei nicht weiter betrachtet.

[3] Hier ist aber wieder zu bemerken, dass die Komplexitätstheorie Aussagen über die Skalierbarkeit von Problemen hinsichtlich ihres Lösungsverhaltens macht. In der Praxis schneidet

der Praxis steht ein entsprechender theoretischer Nachweis allerdings noch aus. Dies zeigt, dass es nicht so einfach ist, nachzuweisen, dass ein Problem nicht zu $\mathcal{P}$ gehört. Ist nämlich für ein Problem bisher kein Algorithmus konstruiert worden, der dieses Problem in polynomialer Zeit löst, so heißt das ja nicht, dass dies nicht noch in Zukunft möglich sein könnte. In der Klasse $\mathcal{NP}$ – diese Abkürzung steht für nichtdeterministisch polynomial – finden wir Probleme, die bisher mit keinem verfügbaren (deterministischen) Algorithmus in polynomialer Laufzeit gelöst werden konnten. Ist für ein Problem der Nachweis der Zugehörigkeit zu $\mathcal{P}$ noch ausstehend und kann gleichzeitig in polynomialer Laufzeit gezeigt und festgestellt werden, dass eine vorgeschlagene Lösung tatsächlich eine Lösung des Problems ist, so zählt es zur Klasse $\mathcal{NP}$. Nützlich ist, dass $\mathcal{NP}$-Probleme gewisse Verwandtschaftsbeziehungen hinsichtlich des erforderlichen Rechenaufwandes aufweisen: sie lassen sich bis auf Konstante in gleicher Zeit bzw. mit gleichem Laufzeitverhalten lösen. Dies führt zur Definition der $\mathcal{NP}$-vollständigen Probleme. Ein Problem heißt $\mathcal{NP}$-vollständig[4], wenn die Existenz eines polynomialen Algorithmus zur Lösung dieses Problems impliziert, dass alle Probleme in $\mathcal{NP}$ in polynomialer Zeit gelöst werden können. Beispiele hierfür sind das Rucksackproblem aus Abschnitt 2.2.7, das Problem des Handlungsreisenden oder das in Abschnitt 5.1.5 beschriebene Erfüllbarkeitsproblem, für das als erstes die $\mathcal{NP}$-Vollständigkeit bewiesen wurde. Auch die gemischt-ganzzahlige Optimierung in all ihren Ausprägungen – MILP, MINLP und auch gemischt-binäre Optimierung – fällt in die Klasse $\mathcal{NP}$ (Karp 1972,[172]); oft wird ein exponentielles Wachstum mit 2^n beobachtet. Dies hat fatale Folgen. Angenommen, ein Problem der Größe $n = 1000$ lässt sich in einer Stunde lösen. Das gleiche Problem mit einem Datensatz, der auf die Größe $n = 1005$ führt, kann dann in kaum unter 32 Stunden gelöst werden, d. h. ein Problem, dass nur um ein halbes Prozent vergrössert wurde, benötigt mehr als ein Tag Rechenzeit. Man mag argumentieren, dass die Klassifikation eines Problems hinsichtlich seiner Zugehörigkeit zu einer Komplexitätsklasse eher von theoretischem Interesse ist. Hat man ein bestimmtes Problem mit Dateninstanzen sich nicht ändernder Größe und kann es in einer zufriedenstellenden Rechenzeit tatsächlich lösen, so könnte man dieser Argumentation zustimmen. Möchte man jedoch etwas über das Skalierungsverhalten des Problems wissen oder kann man es einfach nicht in der gewünschten Rechenzeit lösen, so kann die Klassifikation oder der Nachweis, dass ein Problem $\mathcal{NP}$-vollständig oder $\mathcal{NP}$-schwer ist, helfen, bestimmte Irrwege zu vermeiden und die richtigen Mittel zu finden, sich dem Problem wenigstens mit effizienten Heuristiken und Suchstrategien zu nähern. Gerade im Zusammenhang mit Scheduling wird deutlich, dass die gemischt-ganzzahlige Optimierung wegen der Komplexitätsproblematik auf unüberwindbare Schranken trifft.

Was sind wahrscheinliche Szenarien für Planung und Scheduling in der näheren Zukunft? Bereits jetzt ist erkennbar, dass es eine wachsende Anzahl von Softwarepaketen gibt, die Planungsprobleme mit exakten Verfahren lösen. Gemischt-ganzzahlige Optimierung entwickelt sich zunehmend zu einem akzeptierten Standard für Planung und Design; dies ist sehr offensichtlich in der Prozessindustrie [157]. Für Schedulingprobleme ist dies noch nicht so und meist findet man hier heuristische Verfahren. Aber es ist ein Trend erkennbar, der die mathematische Optimierungszunft und die Gemeinschaft

das Simplexverfahren häufig besser als die Innere-Punkte-Methoden ab.

[4] Weiterhin gibt es noch die $\mathcal{NP}$-schweren Probleme. Jedes Problem, auf das wir ein $\mathcal{NP}$-vollständiges Problem in polynomialer Zeit abbilden bzw. transformieren können und das unabhängig davon, ob dieses Problem zur Klasse $\mathcal{NP}$ gehört, die Eigenschaft hat, dass es nicht in polynomialer Zeit gelöst werden kann, es sei denn, es gilt $\mathcal{P} = \mathcal{NP}$, heißt $\mathcal{NP}$-schwer. Die $\mathcal{NP}$-schweren Probleme sind also mindestens genau so schwer wie die $\mathcal{NP}$-vollständigen.

derer, die auf Constraint Programming setzen, näher zusammenwachsen lässt. Es werden Hybridverfahren [siehe z. B. Harjunkoski *et al.* (2000,[129]), Hooker (2000,[135]) oder Jain & Grossmann (2001,[146])] entwickelt, die Sprachelemente und algorithmische Komponenten beider Welten vereinigen. Es ist zu erwarten, dass dies z. B. auf Supply Chain Optimierung und Schedulingprobleme große Auswirkungen haben wird. Die Europäische Union entschied 1999, das Projekt `LISCOS` (*Large Integrated Supply Chain Optimization Software*) mit mehreren Millionen Euro zu unterstützen. Dieses Projekt (http://www.liscos.fc.ul.pt), das von der Gruppe *Scientific Computing* der BASF Aktiengesellschaft initiiert und gemeinsam mit 8 Konsortiumspartnern durchgeführt wird, zielt auf die Entwicklung von MIP-CP Hybridverfahren. Timpe (2002,[271]) beschreibt eine erfolgreiche Anwendung dieses Ansatzes zur Lösung eines Planungs- und Schedulingsproblems in der chemischen Industrie. Etwa gleichzeitig wurde von der EU das Projekt `COCONUT` [40] zur Lösung nichtkonvexer Optimierungsprobleme mit Methoden aus Globalen Optimierung und Constraint Programming [40] unterstützt. Ein weiterer Ansatz sind zeitkontinuierliche Modellformulierungen; hier sind besonders die Arbeiten [143], [142], [144], [127], [194] und [196] von Floudas und Mitarbeitern sehr vielversprechend.

12.3.1 Simultane operative Planung und Design-Optimierung

In der chemischen Industrie geschieht es häufig, dass ein Kunde ein Planungswerkzeug oder eine Schedulingsoftware für ein Produktionsnetz wünscht, das gerade aufgebaut oder erweitert wurde. Besonders in den Schedulingproblemen stellt sich dann häufig heraus, dass es für bestimme Produkte oder Situationen Engpässe gibt. Die Situation könnte erheblich verbessert werden, wenn während der Designphase die Planungs- und Schedulingaspekte mit einbezogen würden. Zwar mag dieses Problem mathematisch sehr aufwendig erscheinen, da Schedulingprobleme für sich genommen schon sehr schwierig sind. Allerdings sind Schedulingprobleme häufig nur dann sehr schwierig, wenn einige kritische Ressourcen (z. B. Rohmaterial, Maschinenverfügbarkeit oder Personal) besonders knapp werden. Gerade dieses könnte aber durch die simultane Analyse von Design- und Planungsproblem offenbar und vermieden werden. Die simultane Analyse erfordert, dass einigermaßen realistische und detaillierte Nachfrageprognosen vorliegen und die betroffenen operativen und in der Planung tätigen Abteilungen kooperieren; das ist zwar leider oft ein Problem. Die in den Abschnitten 10.1 und 10.2 beschriebene Standortanalyse und Auslegung der Topologie eines Systems von Reaktoren sind erfolgreiche Beispiele.

12.3.2 Simultane operative Planung und strategische Optimierung

Das in Kallrath (2001b,[155]; 2002c,[156]) beschriebene Problem demonstriert den Nutzen eines Modell, dass gleichzeitig operative und strategisch-taktische Aspekte enthält. Die Firma möchte zusätzliche Standorte oder Anlagen erwerben, neue mit verbesserter Technologie installieren, oder ältere Anlagen schließen. In der chemischen Industrie können hierbei logische Verknüpfung zwischen verschiedenen Anlagen gegeben sein, d. h. wird eine Anlage abgestellt, muss möglicherweise eine andere ebenfalls abgeschaltet werden. Die strategischen Aspekte des Problems sind durch die Kosten für den Kauf eines Standorts bzw. für die Inbetriebnahme oder Abschaltung einer Anlage gegeben, wobei die Investitionskosten sich über einen Mehrjahreszeitraum als sinnvoll erweisen sollen. Für die in Frage kommenden Anlagetypen müssen Daten bereitgestellt werden,

wie sie für bereits in der operativen Planung verwendete Anlagen erforderlich sind. Zu den Entscheidungsfreiheitsgraden gehören die Zeitpunkte der Inbetriebnahme bzw. der Abschaltung. Da diese Entscheidungen langfristige Konsequenzen für ein Produktionsnetzwerk haben, ist es wichtig, dass die Optimalität der Lösung bewiesen, zumindest aber die Qualität der Lösung im Sinne oberer und unterer Schranken bewertet werden kann und dann die Lösung gegenüber Marktschwankungen robust sind – dies kann z. B. mit Hilfe der stochastischen Optimierung erreicht werden. In Kallrath (2009, [160]) werden diese Aspekte ausführlich diskutiert.

12.3.3 Polylithische Modellierungs- und Lösungsansätze

Die in Kapitel 8 beschriebenen polylithischen Modellierungs- und Lösungsansätze erweitern die Menge der praktischen Optimierungsprobleme, die sich in vernünftiger Zeit noch lösen lassen. In den Jahren 2006, 2008, 2010 und 2012 wurden regelmässig eine Teilmenge derartiger Techniken, in denen Metaheuristiken wie genetische Algorithmen, Simulated Annealing oder Tabu-Suche und mathematische Optimierungsalgorithmen kombiniert werden – auch *Matheuristics* genannt, auf den *International Workshops on Model-Based Metaheuristics* vorgestellt und diskutiert.

Durch eine zunehmende Verbesserung der algebraischen Modellierungssprachen werden polylithische Verfahren in der Zukunft leichter zu implementieren sein, wodurch ihr Anwendungsumfang und ihre Bedeutung zunehmen wird.

12.3.4 Globale Optimierung

Globale Optimierungstechniken für nichtkonvexe NLP- oder MINLP-Probleme finden Eingang in einer wachsenden Anzahl kommerziell verfügbarer, deterministischer Algorithmen oder Programme. Hierzu zählen BARON, COUENNE, LINDOGLOBAL und neuerlich auch GLOMIQO für MINLP-Probleme, die quadratisch in Zielfunktion und Constraints sind. Eine zunehmende Menge von Problemen, die bisher nur lokal gelöst werden konnten, lassen sich nun schon global lösen – und diese Menge wird zunehmen.

12.4 Mathematische Optimierung für eine bessere Welt

In diesem Buch wurde mathematische Optimierung als ein Modellierungsansatz und eine Lösungstechnik für Probleme vorgestellt, die in der realen Welt – vornehmlich in der Industrie, aber in der Tat überall dort, wo Entscheidungen zu treffen sind – auftreten und einer Quantifizierung in einem deterministischen oder probabilistischen Sinne zugänglich sind. Lassen wir für einen Augenblick einmal Lebewesen auf anderen Planeten und fremden Galaxien außer Acht, so werden auf der Erde Entscheidungen von Menschen für Menschen und andere Lebewesen mit Konsequenzen für alle Lebensarten auf diesem Planeten getroffen. Einige der nachfolgend angesprochenen Aspekte sind daher nicht nur auf Projekte, die mathematische Optimierungsmethoden einsetzen, beschränkt, sondern gelten viel umfassender dort, wo in Entscheidungsprozessen widersprüchliche Interessen aufeinander treffen. Allerdings bietet die mathematische Optimierung gerade unter den Aspekten Sicherheit und Robustheit von Entscheidungen quantitative Unterstützung

bzw. bringt Unstimmigkeiten in Annahmen und Voraussetzungen eines Modells eher an den Tag als dies z. B. mit punktuellen Simulationen möglich ist.

Die Historie auf diesem Planeten hat gezeigt, dass Entscheidungen in sehr verschiedener Vorgehensweise (diktatorisch, pseudo-demokratisch, demokratisch, pseudo-sachlich, strikt sachlich) getroffen werden. Alle Entscheidungen scheinen jedoch die Komponenten Ziel oder *Ziele*, *Randbedingungen*, die man bereit ist zu akzeptieren oder die man aus physikalischen Gründen einhalten muss, und mögliche *Entscheidungsfreiheitsgrade* zu enthalten.

Diktatorische Entscheidungsprozesse sind zwar meist, wenn sie nicht gerade von einem Diktator mittleren oder niederen IQs getroffen werden, konsistent, da der Diktator wahrscheinlich schon genau seine Ziele kennt. Die Entscheidung wird jedoch nicht unbedingt sachlich sein, und sie hat den Nachteil, dass sie nicht alle Betroffenen der Entscheidung einbezieht. Damit ist der Bestand dieser Entscheidung aber fraglich; sobald sich Machtverhältnisse ändern, wird wahrscheinlich auch die getroffene Entscheidung wieder rückgängig gemacht. A priori sollte man auf Basis demokratischer Wurzeln erwarten dürfen, dass die von einer Entscheidung Betroffenen in den Entscheidungsprozess mit einbezogen werden.[5] Beispiele, wie es nicht sein sollte, liefert die Realität leider genügend; Entscheidungsprozesse, dazu gehören auch Meinungsbildungsprozesse, die z. B. den Umgang mit behinderten Menschen betreffen, finden häufig zu Unzeiten – 23^{30} Uhr oder später – und ohne entsprechende Repräsentation behinderter Menschen und sogar ohne diejenigen, die helfend tätig sind (Krankenschwestern, Pflegepersonal), statt. Entscheidungen würden sicherlich besser akzeptiert, wenn z. B. bei Produktionsplanungssystemen und Personaleinsatzplänen, insbesondere solchen, die mühelos vom Ziel Urlaubsplanung zum Ziel Personalreduzierung wechseln können, die Betroffenen in den Modellierungsprozess einbezogen würden. Demokratische und kollektive Entscheidungsprozesse haben aber auch ihre Tücken. Sie können, wie in der mathematischen Spieltheorie bewiesen und in Eichner *et al.* (1996,[78]) diskutiert wird, zu recht seltsamen Paradoxa führen. Wenn mehr als drei mit Prioritäten versehene Alternativen zur Verfügung stehen, ist es möglich, dass sich Entscheidungen ergeben, die eigentlich von niemandem gewollt sind. Pseudo-demokratische Entscheidungsprozesse, das sollen hier solche sein, die nicht wirklich demokratisch sind, sei es z. B., dass nicht alle Informationen zur Verfügung stehen bzw. Informationen von einigen Beteiligten zurück gehalten werden, oder, dass nicht anonym abgestimmt wird und einige Beteiligte sich nicht trauen, ihre Meinung zu sagen, werden ebenfalls fragwürdige Entscheidungen produzieren. In sachlichen Entscheidungsprozessen wird versucht, die Fakten so sauber und absichtlich neutral wie möglich zu präsentieren; insbesondere werden keine Fakten bewusst zurück gehalten oder verzerrt dargestellt. Es wird auch jedem zugestanden, diese Fakten vor seinem eigenen Hintergrund zu verstehen, zu bewerten und zu interpretieren, d. h. daraus sein eigenes Bild der Realität zu konstruieren. Wichtig ist aber, dass ein Konsens darüber herrscht, welche Argumentations- und Begründungsstrukturen anerkannt und zugelassen sind und dass eine Orientierung an der Realität vorgenommen wird. Diejenigen, die der philosophischen Richtung des Konstruktivismus nahe stehen, mögen verzeihen, dass hier sehr vereinfacht von Realität gesprochen wird. Was hier gemeint ist, soll an folgendem Beispiel verdeutlich werden. Noch im 18. Jahrhundert hielt die Pariser Akademie der Wissenschaft an der traditionellen Lehre fest, dass nichts vom Himmel auf die Erde fallen kann; dies trotz

[5] Vielleicht aber auch gerade nicht, denn den alten Griechen wäre in ihrer Demokratie wohl auch nicht eingefallen, Sklaven in den Entscheidungsprozess mit einzubeziehen.

einiger Meteoritenfunde französischer Bauern - man weigerte sich schlicht, genau hinzuschauen oder zuzuhören. Den Sonnenflecken im Jahre 1609 und 1610 erging es ähnlich; es gab Leute, die wollten sie einfach nicht anschauen.

Auftraggeber, Einzelpersonen oder Teams, die einen Entscheidungsprozess mit welchem Ziel auch immer so gut wie möglich abschließen möchten, und auch die Modellierer, die das Problem einer mathematischen Optimierung zugänglich machen sollen, müssen daher folgendes beachten:

1. Soll die Entscheidung so gut wie möglich sein, so müssen alle möglichen Konsequenzen und Aspekte untersucht werden; d. h. noch nicht, dass sie in das zu entwickelnde Modell einfließen müssen – es ist denkbar, dass sie durch sichere Fehlerabschätzung aus der weiteren Betrachtung entfallen können. So ist z. B. für den Bau eines Aufzugs unter irdischen Gravitationsverhältnissen innerhalb der erforderlichen bautechnischen Ungenauigkeit die Newtonsche Gravitationstheorie ausreichend, und es braucht nicht die Allgemeine Relativitätstheorie bemüht zu werden. Bestimmte Aspekte nicht zu betrachten, kann aber ein schlimmer Fehler sein und zu Akzeptanzproblemen führen (s. S. 8 und die betroffenen Lastwagenfahrer).

2. Ist der Verantwortliche für den Entscheidungsprozess ein klassischer Diktator, so sollte auch er besser die Möglichkeit von Aufständen als Reaktion auf bestimmte Entscheidungen in das Modell mit einfließen lassen – meistens lieben Diktatoren ein langes Leben als Diktator und ihr Interesse an Revolutionen ist gering. Der Modellierer sollte an dieser Stelle besonders vorsichtig sein, denn ein mathematisches Modell legt sämtliche systemimmanenten Zusammenhänge klar offen; ist dies dem Diktator klar, so lebt der Modellierer in ständiger Lebensgefahr, da er zu viel weiß.

3. Demokratische Entscheidungsprozesse führen allerdings auch nicht immer zu den besten aller möglichen Welten und können den tollsten Unsinn produzieren; die Sache wird ja auch nicht besser dadurch, dass man zu einer Gruppe von 600 unqualifizierten Entscheidern noch weitere 100 hinzugibt. In Amerika stand 1897 ein Gesetzesentwurf – House Bill. No. 246 of the 1897 Indiana State Legislature – zur demokratischen Abstimmung an, den Wert der Kreiszahl $\pi = 3.141592653\ldots$ vereinfachend auf z. B. 3.2 zu setzen, was aber wohl gerade noch von einem Mathematiker verhindert wurde [74]. Bei allen demokratischen oder durch Konsens herbeigeführten Entscheidungen sollte deshalb darauf geachtet werden, dass nicht gegen wahre Sachverhalte oder gar gegen Naturgesetze verstoßen wird – die Natur versteht nämlich nicht allzu viel Spaß. Physiker und Ingenieure werden dies bestätigen.

4. Der Modellierer ist gut beraten, zu versuchen, herauszufinden, was die wahren Motive seines Auftraggebers sind. Gerade in grossen Konzernen kommt es gelegentlich vor, dass Auftraggeber alles Mögliche im Sinne haben, zuletzt aber das Wohl der Firma; siehe hierzu auch De Beuckelaer (2001,[66]). Derartige Auftraggeber sollten sich der mathematischen Optimierung mit großer Vorsicht nähern, denn ihre Haltung könnte leicht aufgedeckt werden.

Die mathematische Modellierung und Optimierung sind bestens gerüstet, Entscheidungsprozesse quantitativ zu unterstützen, die strikt sachlich durchgeführt werden sollen – es sei dahin gestellt, ob dies in Wirklichkeit oft vorkommt; wenn nicht, sollte man versuchen, diesem Ideal so nahe wie möglich zu kommen.

Neben dem bereits oben Gesagten sollte man bei strikt sachlichen Entscheidungsprozessen beachten, dass hierbei und im Modellierungsprozess jeder, der sachlich kompetent dazu beitragen kann, dies ohne Berücksichtigung seines Status und ohne Konsequenzen für seine Karriere auch darf, – einige Arbeiter während der Design- oder Modellierungsphase hinzuzuziehen, könnte z. B. von Nutzen sein. Während der Modellierungsphase sollte also jeder Beteiligte zunächst einmal völlig unabhängig sein. Dies würde die Unvorhergenommenheit fördern und der Sache gut tun. Die besten Projekte, die mir bekannt sind, waren solche mit neuen Mitarbeitern, die dem Konzern erst kurz angehörten, oder solche, die kurz, d. h. nicht mehr als 2 Jahre, vor ihrer Pensionierung standen; beide Gruppen waren an Motivation, Offenheit, Sachlichkeit und Engagement kaum zu überbieten.

Nehmen wir einmal an, dass ein Auftraggeber einen strikt, oder doch zumindest einen sehr weitgehend sachlichen Entscheidungsprozess wünscht, d. h. er wird von störenden firmeninternen Prozessen, Gruppen-, Abteilungs- und Bereichsegoismen oder Überlegungen, was seiner Karriere nützlich oder schädlich ist, absehen. Damit sollte seine Akzeptanz einhergehen, alle Mitarbeiter, deren Einbeziehung in den Modellierungsprozess sinnvoll ist, entsprechend mit einzubinden. Dies setzt voraus, dass auch diese Mitarbeiter das Problem sachlich angehen. In Unternehmen mit guter Führungsstruktur sollte dies möglich sein; in Firmen, in denen die Umstrukturierungen zu Beginn des 21. Jahrhunderts völlig unnötigerweise das Vertrauen der Mitarbeiter in die Unternehmensführung zerstört haben, ist das schon problematischer und manchmal wohl eine große Führungsaufgabe.

Kann ein Entscheidungsproblem ganz oder teilweise als mathematisches Optimierungsproblem formuliert und soll es als solches gelöst werden, so benötigt man sowohl auf Seiten des Auftraggebers als auch auf Seiten der Modellierung Menschen, die das Problem *so gut wie möglich lösen* wollen und nicht durch zuviele andere Dinge gedanklich und in ihrer Entscheidungsfreiheit gefesselt sind. Der Grundgedanke der Optimierung, wenn er richtig und nachhaltig verinnerlicht wurde, lässt einen nämlich nicht ruhen, bevor man nicht wirklich die optimale Lösung oder doch wenigstens eine sichere Aussage hat ableiten können. Optimierung ist nicht nur eine Lösungstechnik, sondern im guten Sinne eher eine Lebenseinstellung. Ein guter Arzt, der noch dem *Eid des Hippokrates*[6] folgt, wird bei einem schwerkranken Patienten auch nicht eher ruhen[7], bis er ihn entweder gerettet hat oder sicher sein kann, alles versucht zu haben. Ein Feuerwehrmann, der nach Lebenden in einem brennenden Gebäude sucht, wird eine ähnliche Einstellung haben – warum sollte er sonst sein Leben einsetzen? Die 80%-Mentalität und Argumentationsweise, *80% sind gut genug, alles andere ist kostenmäßig nicht zu vertreten*, ist dagegen ein ganz anderer Ansatz, der, wenn er gut begründet ist, auch seine Berechtigung hat – nur, meist ist die Begründung eben nicht gut, weil a) oft das Gesamtpotential gar nicht bekannt ist: man kennt es häufig erst nach einer rigorosen mathematischen Untersuchung und b) weil das zusätzliche Potential nicht ins Verhältnis zu den Kosten gesetzt wird. Die oben erwähnte Lebenseinstellung hat weitestgehend auch damit zu tun, was

[6] Arzt des Altertums, der etwa von 460 bis 377 v. Chr. gelebt hat und die griechische Heilkunde begründete. Er formulierte den Hippokratischen Eid, der für einen wahren Arzt auch heute noch gültige sittliche Gebote enthält. Für Naturwissenschaftler würde die Entsprechung des Hippokratischen Eides wohl „Messwerte fälsche man nicht!" lauten; im Falle von Personen mit Entscheidungsverantwortung, insbesondere auch Manager und Politiker, wäre vielleicht das sittliche Gebot „Nenne offen und vollständig alle Ziele und Absichten" sinnvoll.

[7] Im wahrsten Sinne des Wortes wird er sich nämlich auch nicht um Feierabend, Wochenende oder andere Dinge, die gerne in betrieblichen Vereinbarungen oder im deutschen Arbeitsrecht verankert sind, kümmern.

als Argument und Begründungsstruktur akzeptiert wird.

Soweit es den Modellierer betrifft, hängt der Projekterfolg von seinen technischen Kenntnissen der mathematischen Optimierung ab, mehr aber vielleicht noch von seiner Motivation – sportive oder altruistische Elemente können zu großen Leistungen führen. Problematisch ist, wenn der Modellierer Konzernstrukturen oder wirtschaftlichen Zwängen ausgesetzt ist bzw. wie ganze Beratungsbranchen unter dem Druck steht, sich über Projekte *verrechnen* zu müssen – wie kann man da sicher sein, dass es dann nicht manchmal zu seltsamen Verzerrungen kommt? Wird dann wirklich diejenige Person oder dasjenige Team mit bester Eignung das Projekt durchführen, oder kann es vorkommen, dass weniger geeignetes Personal und manchmal sogar unpassende Projektleiter mit Arbeit versorgt werden, damit sie sich *verrechnen* können?

Ein gut durchgeführtes Projekt, das mathematische Optimierungsaspekte enthält, wird, falls die Komplexität des zugrunde liegenden Problems dies zulässt – dies zeigt sich allerdings aus mathematischen Gründen oft erst im Projektverlauf –, mit hoher Wahrscheinlichkeit zu einem Erfolg und zu einer Situation führen, in der alle Beteiligten gewinnen – und zwar in ihrer eigenen, selbst gewählten von allen Beteiligten akzeptierten Skala. Es benötigt dazu aber die Beteiligung aller und deren Motivation, sowie eine Atmosphäre, die erlaubt, über Ziele sachlich, offen und vertrauensvoll zu reden und zu einem wirklichen Konsens zu kommen. Frei nach dem Motto „*Nenne mir Deine Ziele und ich sage Dir, wie Du Dich verhalten wirst*", dürfen gerade die Ziele für die Mitglieder des Projektteams, aber auch die des Optimierungsproblems selbst, nicht zu eng gefasst sein, sondern müssen so formuliert sein, dass ein konsistentes Verhalten entsteht. Ein erstes Beispiel für eine inkonsistente Zielvereinbarung ist z. B. die Kopplung, dass es bei mehr Leistung mehr Bruttoeinkommen gibt; das kann in Deutschland infolge von Steuerfreigrenzen für bestimmte Abschreibungen dazu führen, dass es bei mehr Leistung weniger Nettoeinkommen gibt - die einzig richtige und sinnvolle Kopplung wäre, Leistung und Nettoeinkommen zu koppeln. Ein weiteres Beispiel besteht in der Kopplung, dass Vertriebsleute, möglichweise sogar noch produktspezifisch, nach verkaufter Menge (in Tonnen) entlohnt werden, ohne jedoch die Kosten zu berücksichtigen, die sie z. B. durch Vergabe von Rabatten oder (vielleicht sogar bewusst) zu hoher Prognosen in Produktion und Logistik verursachen; es könnte besser sein, etwas weniger von einem bestimmten Produkt zu verkaufen und dafür mehr Gewinn mit einem anderen Produkt zu erzielen. Entkoppelte Marketing- und Produktionsabteilungen, die z. B. den Zielvorgaben „versuche so viel wie möglich zu verkaufen" und „produziere möglichst und stets nahe der Vollauslastung" folgen, können erhebliche Kosten produzieren. Bei länderübergreifenden Produktionsplanungssystemen in großen Konzernen sollten nationale Steueraspekte und Freigrenzen berücksichtigt werden und die Zielfunktion „Nachsteuereinkommen" maximieren; die Modelle werden dadurch zwar komplizierter, aber die Ergebnisse, nämlich die nach Steuer, auch wesentlich sinnvoller und ergiebiger. Ein weiteres Beispiel, in dem die Kopplung zu seltsamen Verhaltensweisen führen kann, sind Transferpreise in Unternehmungen. Man sollte erwarten, dass diese wenigstens von übergeordneten Unternehmensinstanzen fixiert werden, um sicherzustellen, dass die Transferpreise so gestaltet werden, dass es für das Unternehmen optimal ist, aber das ist leider nicht immer der Fall. Firmenpolitische Transferpreise, die unter Umständen sogar Marktrealität vermissen lassen, können aber zu völlig unsinnigen Materialflüssen innerhalb eines Unternehmens führen. Wesentlich besser wäre es, faire und konsistente Preise mit Hilfe der Spieltheorie und der mathematischen Optimierung zu bestimmen. Bei all den Beispielen entstehen die Schwierigkeiten und Inkonsistenzen dadurch, dass unzulässige Dekompositionen durchgeführt

werden, eine enge lokale Sichtweise vorherrscht und die Zielfunktion nicht ganzheitlich genug formuliert wird. Herrscht aber ein konstruktiver, sachlich fundierter Konsens[8] aller Beteiligten hinsichtlich der Ziele, Erlöse und Kosten und ist die Bereitschaft da, die sachliche Ebene nicht zu verlassen und der Sache den Vorrang vor persönlichen, firmenpolitischen, gruppen- oder abteilungsegoistischen Denkweisen zu geben, so kann man sich vermutlich auch auf die übrigen Aspekte der mathematischen Optimierung (Nebenbedingungen und Freiheitsgrade) gut einigen.

Nach der Diskussion einiger Punkte, die den Erfolg von Optimierungsprojekten gefährden, ist es nun bestimmt auch lohnend, zu beleuchten, welche Eigenschaften erfolgreichen Projekten zu eigen waren:

1. Die Projektergebnisse bzw. die auf mathematischer Optimierung beruhenden Planungswerkzeuge wurden wirklich von allen Beteiligten gewollt. Und es wurde niemand ausgeschlossen, der gerne beteiligt sein wollte.
2. Die Ergebnisse – und das war meist früh absehbar – führten zu einer Situation, in der alle einen – mit ihrem jeweils eigenen Maßstab gemessenen – Gewinn hatten. Hier wird aber wieder klar ersichtlich, wie wichtig Offenheit, Vertrauen und eine tolerante, konstruktive Fehlerkultur ist; andernfalls wird nicht unbedingt jeder seine Ziele und Maßstäbe offen nennen.
3. Die Projekte hatten meist starke Persönlichkeiten als Projektführer, die im Sinne des Teams handelten und entschieden, aber klar die Verantwortung als einzeln handelnde Personen hatten. Es ist wahrscheinlich eine der fragwürdigsten Ideen in Politik, Wirtschaft und Verwaltung der vergangenen 20 Jahre, Entscheidungskompetenz auf Teams, Komitees oder ähnliches zu verteilen – man stelle sich dies einmal in der Natur vor, eine Schar von Jägern z. B., die während der Jagd erst einmal groß debattieren; von den Debatten nach der Jagd, wenn es gilt, die Jagdbeute zu verteilen, so man denn überhaupt mit diesem System zu einem derartigen Erfolg kommt, einmal ganz abgesehen. Die Analyse erfolgreicher Projekte zeigt, dass von Einzelpersonen geführte Vorhaben wesentlich besser abschneiden als andere.
4. Die Projektteams wurden gezielt und unter Berücksichtigung der Aufgabenstellung zusammengestellt. Insbesondere wurde bei der Zusammenstellung Wert auf die *innere Chemie* der Teams gelegt.
5. Die Anzahl der in den erfolgreichen Projekten beteiligten Personen war stets recht klein, meist nicht mehr als fünf. Das muss nicht bedeuten, dass Projekte mit mehr Teilnehmern nur Unglück bringen. Aber es sind dann sicherlich noch höhere Anforderungen an Sachlichkeit, Vertrauen und Offenheit zu erfüllen; der Projektführer muss dann wirklich eine Führerpersönlichkeit im guten Sinne sein, der sein Team motiviert, auf die Eigenheiten der Teammitglieder eingeht und ihr Wissen und ihre Fähigkeiten zu nutzen versteht und von allen als Projektführer anerkannt ist.

[8] An der Stelle ist Vorsicht geboten! Sachlich stellt hier einen gewissen Gegensatz zu personenbezogen dar. Ein Argument zählt, unabhängig davon, wer das Argument vorbringt. Ein Konsens unter Experten, die nicht bereit sind, Ihre Entscheidungen und deren Gründe nach außen offen zu legen bzw. die es ablehnen, Fachleute von außerhalb ihres Kreises mit einzubeziehen, ist verdächtig. Deshalb wird hier auch betont *alle Beteiligten*. Es kann auch nicht schaden, von Zeit zu Zeit einmal externen, neutralen Rat einzuholen oder die internen Experten von externen, qualifizierten Institutionen evaluieren zu lassen. Besondere Beachtung sollte hier der Begriff *qualifiziert* finden; Berater von namhaften Beraterfirmen sind nicht notwendigerweise als qualifiziert anzusehen.

6. Die besten Auftraggeber bzw. Kunden waren Mitarbeiter, die der Firma erst wenige Jahre angehörten, oder solche, die kurz, d. h. nicht mehr als 2 Jahre, vor ihrer Pensionierung standen.

Mathematische Optimierung ist sehr eng verknüpft mit Lebensauffassung – sie erzieht zu mehr Präzision und Sachlichkeit, denn Unstimmigkeiten werfen stets neue Fragen auf und verlangen, beseitigt zu werden –, aber auch mit Führung und Führungsstil. Herrschen in der Unternehmenskultur Offenheit, Ehrlichkeit und hat Sachkompetenz Vorrang vor der Politik, gibt es dort eine gute Mischung von Menschen mit verschiedenem Hintergrund, die sich mit ihrem jeweiligen fachlichen Hintergrund akzeptieren, respektieren und kooperieren, so sollte es naheliegend und selbstverständlich sein, dass Entscheidungsprozesse durch mathematische Optimierung unterstützt werden. Es muss zwar nicht immer mathematische Optimierung sein, aber häufig ist sie das einzig adäquate Mittel, Entscheidungen in komplexen Systemen mit vielen Freiheitsgraden oder vielen Restriktionen sicher und quantitativ zu untermauern, vorhandene Ressourcen effizienter einzusetzen, oder robuste strategische Entscheidungen bei unsicheren Eingangsdaten zu treffen. Gerade in Zeiten, wo die Bereitschaft, Risiken einzugehen, eher bescheiden ist – manche Soziologen sprechen vielleicht nicht unberechtigt davon, dass wir in einer Risikovermeidungsgesellschaft leben – , sollte man erwarten, dass die mathematische Optimierung, die es erlaubt, sichere Aussagen bezüglich der Zulässigkeit und Güte von Entscheidungen zu treffen, hoch im Kurs steht. Es werden nicht immer spektakuläre, karrierebeschleunigende Ergebnisse sein, eher kann die mathematische Optimierung gerade in großen Unternehmungen in kleinen Schritten dazu beitragen – ein Physiker würde wohl sagen: in *adiabatischer* Weise – die technische, prozedurale, diskutiv-argumentative Basis eines Unternehmens zu verbessern, sofern es denn diesen Prozess wünscht und zulässt.

Große Unternehmen mit ihrer hohen Mitarbeiterzahl sind aus statistischen Gründen ein kleineres Abbild der gesamten Gesellschaft, eigentlich des ganzen Staates. Da deren Mitarbeiter aber auch Mitglieder von Staat und Gesellschaft sind und die verinnerlichte Firmenkultur teilweise mit in die Gesellschaftskultur einbringen, wird ersichtlich, dass die mathematische Optimierung nicht nur die Entscheidungsprozesse und wirtschaftliche Situation eines Unternehmens oder dessen innere Kultur verbessert, sondern weit darüber hinaus beitragen kann zu einer für alle besseren Welt.

A Mathematische Beschreibung der Optimierungsverfahren

Der Anhang beschreibt – einige elementare Kenntnisse in der Analysis und der linearen Algebra voraussetzend – die verwendeten Optimierungsverfahren eingehender und soll tiefere Kenntnisse vermitteln, die in vielen Fällen bei der Modellierung sehr relevant sind.

A.1 Eine tiefere Betrachtung des Simplexverfahrens

A.1.1 Das Simplexverfahren — Eine detaillierte Beschreibung

Betrachtet sei das lineare Optimierungsproblem in der Standardform

$$\mathbf{LP}: \max \quad \left\{ \mathbf{c}^{\mathrm{T}}\mathbf{x} \mid \mathsf{A}\mathbf{x} = \mathbf{b}, \quad \mathbf{x} \geq 0, \quad \mathbf{x} \in \mathbb{R}^n, \quad \mathbf{b} \in \mathbb{R}^m \right\} \quad , \tag{A.1.1}$$

wobei $\mathbb{R}^n$ den Vektorraum der reellen Vektoren mit n Komponenten bezeichnet und in der Regel $n \geq m$ gilt.

In Abschnitt 4.2.4 wurde das Konzept der Basisvariablen eingeführt, die in dem Vektor $\mathbf{x}_B$ zusammengefasst sind. Die algebraische Grundlage des Simplexverfahrens ist das Konzept der Basis $\mathcal{B}$ von A, einer Matrix, die aus einer Menge linear unabhängiger Spalten $\mathcal{B}$=$\{\mathbf{A}_{j_1},...,\mathbf{A}_{j_m}\}$ von A gebildet wird. Manchmal bezeichnet man als Basis auch die Indexmenge $\mathcal{J} = \{j_1, ..., j_m\}$, die die linear unabhängigen Spalten von A indiziert. Die inverse Matrix $\mathcal{B}^{-1}$ liefert eine Basislösung $\bar{\mathbf{x}} \in \mathbb{R}^n$

$$\bar{\mathbf{x}}^{\mathrm{T}} = \left(\mathbf{x}_B^{\mathrm{T}}, \mathbf{x}_N^{\mathrm{T}}\right) \quad ,$$

wobei $\mathbf{x}_B$ der Vektor der Basisvariablen ist und sich gemäß

$$\mathbf{x}_B = \mathcal{B}^{-1}\mathbf{b}$$

berechnet, während $\mathbf{x}_N$ ein $(n-m)$-dimensionaler Vektor

$$\mathbf{x}_N = \mathbf{0} \quad , \quad \mathbf{x}_N \in \mathbb{R}^{n-m}$$

aus Nichtbasisvariablen ist. Da die Matrix A nicht mehr als m linear unabhängige Spalten haben kann, kann die Basis auch nicht mehr als m Variablen enthalten, d. h. nicht mehr als m Variablen können von Null verschiedene Werte annehmen. Die übrigen $n-m$ Variablen, die Nichtbasisvariablen, müssen die Werte Null annehmen. Genügt $\bar{\mathbf{x}}$ neben obigen algebraischen Bedingungen noch den Ungleichungsbedingungen $\bar{\mathbf{x}} \geq 0$, d. h. ist $\bar{\mathbf{x}}$ ein zulässiger Punkt in $S = \{\mathbf{x} : \mathsf{A}\mathbf{x} = \mathbf{b}, \mathbf{x} \geq 0\}$, so wird $\bar{\mathbf{x}}$ *zulässige Basislösung* [engl.: *basic feasible solution*] oder *zulässiger Basispunkt* [engl.: *basic feasible point*] genannt. Gelten die folgenden Bedingungen

1. die Matrix A hat m linear unabhängige Spalten $\mathbf{A}_j$,

J. Kallrath, *Gemischt-ganzzahlige Optimierung: Modellierung in der Praxis*,
DOI 10.1007/978-3-658-00690-7_13, © Springer Fachmedien Wiesbaden 2013

2. der zulässige Bereich S ist nicht leer und

3. die Menge der Zielfunktionswerte $\{\mathbf{c}^{\mathrm{T}}\mathbf{x} : \mathbf{x} \in S\}$ besitzt eine obere Schranke,

so entspricht S den Punkten eines konvexen Polyeders P ([224], Satz 1.1.4, S.47) und jeder zulässige Basispunkt korrespondiert zu einer Ecke von P ([224], Satz 1.1.7, S.52). Gemeinsam garantieren (2) und (3), dass das LP-Problem weder unzulässig noch unbeschränkt ist, somit ein Optimum annimmt.

Mit Hilfe des *Eckensatzes* der linearen Programmierung, siehe z. B. ([224], Satz 1.1.5, S. 48), lässt sich zeigen, dass zur Bestimmung der optimalen Lösung lediglich alle Basispunkte hinsichtlich ihrer Zulässigkeit hin untersucht werden müssen, von den zulässigen die Zielfunktion ausgewertet wird und dann schließlich so der beste Punkt gewählt wird. In diesem Sinne ist die Bestimmung des Optimums eines LP-Problems ein kombinatorisches Problem. Im ungünstigsten Fall gibt es

$$f_1(n,m) := \left(\begin{array}{c} n \\ m \end{array} \right) = \frac{n!}{m!(n-m)!} \tag{A.1.2}$$

Basislösungen. Sofern das Problem nicht entartet ist [was das bedeutet, wird weiter unten erklärt], findet sich die optimale Lösung in dieser endlichen Menge von Basispunkten. McMullen (1970,[208]) hat gezeigt, dass höchstens[1]

$$f_2(n,m) := \left(\begin{array}{c} n - \left\lfloor \frac{m+1}{2} \right\rfloor \\ n-m \end{array} \right) + \left(\begin{array}{c} n - \left\lfloor \frac{m+2}{2} \right\rfloor \\ n-m \end{array} \right) \tag{A.1.3}$$

zulässige Basislösungen geben kann. In einem Beispiel mit $n = 16$ und $m = 8$ ist $f_1(16,8) = 12870$ und $f_2(16,8) = 660$. Zwar ist f_2 wesentlich kleiner als f_1, aber wir bekommen doch ein Gefühl dafür, dass ein rein kombinatorischer Zugang in der Praxis wohl nicht attraktiv ist.

Geometrisch interpretiert, folgt das Simplexverfahren den Ecken eines Polyeders. In jedem Iterationsschritt wird der Wert der Zielfunktion entweder verbessert oder bleibt gleich. Klee & Minty (1972,[178]) haben ein Beispiel eines LP-Problems mit n Variablen konstruiert, in dem 2^n Ecken untersucht werden müssen, d. h. das Simplexverfahren kann bis zur Beendung exponentiell viele Schritte benötigen. Algebraisch wird in jeder Iteration eine Spalte der gegenwärtigen Basismatrix A geändert, d. h. in einem Basisaustauschschritt werden aus A und den Vektoren $\mathbf{b}$ und $\mathbf{c}$ die neuen Daten A', $\mathbf{b}'$ und $\mathbf{c}'$ berechnet; dieser Schritt wird auch *Pivot*, *Pivotoperation* oder *Pivotschritt* genannt.

Jetzt lässt sich auch besser erläutern, was wir unter einem entarteten LP-Problem verstehen wollen. Algebraisch interpretiert, existieren in der Lösung eines entarteten LP-Problems Basisvariablen, die den Wert einer ihrer Schranken annehmen, sich also wie Nichtbasisvariablen verhalten. Kombinieren wir die algebraische und geometrische Sichtweise, so kann die Ecke mit optimalen Zielfunktionwert algebraisch durch verschiedene Basen dargestellt werden. Man könnte dies auch so interpretieren, dass die Ecken, die diesen Basen formal entsprechen, durch Kanten der Länge Null verbunden sind.

Anstatt nun die gesamte Systemmatrix A im Speicher zu halten bzw. bei jedem Iterationsschritt zu transformieren, arbeitet das *Revidierte Simplexverfahren* [engl.: *revised simplex algorithm*] lediglich mit den Eingangsdaten $\mathsf{A}, \mathbf{b}$ und $\mathbf{c}^{\mathrm{T}}$ sowie der aktuellen Basisinverse $\mathcal{B}^{-1}$. Die Rechenschritte dieses Verfahrens seien wie folgt zusammengefasst.

[1] Dieses Result ist nur dann richtig, wenn das Problem keine oberen Schranken [siehe Abschnitt A.1.3] enthält; diese Voraussetzung ist in der Standardform gefüllt.

Zunächst wird, wie in Abschnitt A.1.2 beschrieben, eine zulässige Basis $\mathcal{B}$ bestimmt. Dieses Problem zu lösen, ist strukturell genau so schwierig wie das Optimierungsproblem selbst zu lösen.

Ist die Basis bekannt, so kann daraus im Prinzip[2] die Basisinverse $\mathcal{B}^{-1}$ der Basismatrix berechnet werden. Hieraus folgen die Werte der Basisvariablen

$$\mathbf{x}_B = \mathcal{B}^{-1}\mathbf{b} \quad , \tag{A.1.4}$$

sowie die Werte der dualen Variablen $\boldsymbol{\pi}^{\mathrm{T}}$

$$\boldsymbol{\pi}^{\mathrm{T}} := \mathbf{c}_B^{\mathrm{T}}\mathcal{B}^{-1} \quad . \tag{A.1.5}$$

Hierbei ist zu beachten, dass es sich bei $\boldsymbol{\pi}^{\mathrm{T}}$ um einen Zeilenvektor handelt. Nun sind wir in der Lage, für alle Nichtbasisvariablen die reduzierten Kosten d_j

$$d_j = c_j - \boldsymbol{\pi}^{\mathrm{T}}\mathbf{A}_j \tag{A.1.6}$$

zu berechnen [die reduzierten Kosten aller Basisvariablen sind identisch Null $\mathbf{d}(\mathbf{x}_B) = \mathbf{0}$]. Bemerkenswert ist, dass die Formel (A.1.6) zur Berechnung der reduzierten Kosten nur die ursprünglichen[3] Daten c_j, $\mathbf{b}$ und $\mathbf{A}_j$, sowie die aktuelle Basis $\mathcal{B}$ verwendet. Ist irgendein Wert d_j positiv, so kann die Zielfunktion durch entsprechende Erhöhung von x_j verbessert werden; die bisher bestimmte zulässige Basis entspricht noch nicht der optimalen Lösung. Die Auswahl, welches der möglichen d_j man wählen soll, erfolgt teilweise heuristisch. In Lehrbüchern findet man meist den Hinweis, die Nichtbasisvariablen mit größtem d_j zu wählen, aber kommerzielle Algorithmen verwenden eher das sogenannte *Partielle Pricing* [engl.: *partial pricing*] oder auch *Devex Pricing* [130]. Der Begriff *partiell* deutet hierbei an, dass die reduzierten Kosten nicht für alle Nichtbasisvariablen bestimmt werden. In einem Maximierungsproblem wird manchmal einfach die Nichtbasisvariable gewählt, für die zum ersten Mal ein positives d_j gefunden wird; andere Heuristiken wählen die neue Basisvariable zufällig aus einer Menge von Kandidaten. Schließlich wird eine Heuristik benötigt, die steuert, wann von partiellem zu komplettem *Pricing* gewechselt werden soll. Komplettes Pricing wird benötigt, da nur dieses den Optimalitätsnachweis ermöglicht. Ein hinreichendes Optimalitätskriterium in einem Maximierungsproblem lautet

$$d_j = c_j - \boldsymbol{\pi}^{\mathrm{T}}\mathbf{A}_j \leq 0 \quad , \quad \forall j \tag{A.1.7}$$

bzw. in einem Minimierungsproblem

$$d_j = c_j - \boldsymbol{\pi}^{\mathrm{T}}\mathbf{A}_j \geq 0 \quad , \quad \forall j \quad . \tag{A.1.8}$$

Zu beachten ist, dass es sich hierbei um hinreichende Kriterien handelt. In entarteten[4] Problemen können algebraisch verschiedene Basen dieselbe zulässige Basislösung,

[2] Wie auf Seite 316 betont, wird die Basisinverse $\mathcal{B}^{-1}$ in kommerziellen Implementationen fast nie explizit berechnet. Ein großer Aufwand wird betrieben, um die Basismatrix ausgefeilt zu faktorisieren und damit das Gleichungssystem im (revidierten) Simplexverfahrens effizient zu lösen; siehe z. B. Chvátal (1983,[51]).

[3] Von nun an bezeichne $\mathbf{A}_j$ die Spalten der ursprünglichen Matrix A, die der Variablen x_j entsprechen.

[4] Auf die Problematik der Zyklen beim Simplexverfahren im Falle entarteter Basislösungen wollen wir hier nicht näher eingehen, da dies bei praktischen Problemen und Verwendung guter kommerzieller Algorithmen kaum bzw. nie vorkommt. Hierzu tragen spezielle Zusatzregeln bei, von denen eine in Werner (1992,[288]) beschrieben ist.

sprich Ecke, repräsentieren; einige davon erfüllen möglicherweise die Kriterien (A.1.7) oder (A.1.8) nicht. Liegt diese Situation der primalen Entartung nicht vor und existieren dennoch alternative Lösungen – dieser Fall wird *duale Entartung* genannt –, dann müssen die reduzierten Kosten für einige der Nichtbasisvariablen notwendigerweise den Wert Null annehmen. Gilt in einem Maximierungsproblem $d_j < 0$ für alle Nichtbasisvariablen, so ist die optimale Lösung eindeutig; bei größeren Problemen in der Praxis ist dies allerdings nur selten der Fall.

Solange das Optimum noch nicht erreicht ist, wird auch getestet, ob das Problem möglicherweise unbeschränkt ist. Ist das Problem beschränkt, so wird die Regel des minimalen Verhältnisses verwendet, um eine Basisvariable zu eliminieren. Beide Schritte finden etwa gleichzeitig statt: die Regel des minimalen Verhältnisses versagt nämlich, wenn der Eingangsvektor zu einer unendlichen Verbesserung führt. Die Daten, die zur Anwendung dieser Regel nötig sind, folgen ebenfalls direkt aus $\mathcal{B}^{-1}$

$$\mathbf{A}'_j = \mathcal{B}^{-1}\mathbf{A}_j \quad .$$

Nach Anwendung dieser Regel liegt eine neue Basis, d. h. eine neue Indexmenge linear unabhängiger Spalten vor.

Bleibt noch zu klären, wie die aktuelle Basisinverse $\mathcal{B}^{-1}$ berechnet wird. Hierzu gibt es verschiedene Verfahren, die aber allesamt äquivalent sind und vermeiden, $\mathcal{B}^{-1}$ explizit zu berechnen. Elementare Zeilenoperationen überführen die existierende Basisinverse in die aktuelle Basisinverse der nächsten Iteration.[5] Aus numerischen Gründen wird jedoch z. B. alle 100 Iterationen die Basisinverse erneuert, indem die ursprüngliche Matrix invertiert wird. Dadurch wird verhindert, dass sich Rundungsfehler akkumulieren. Ein weiterer Vorteil des revidierten Simplexverfahrens besteht in der besseren Ausnutzung der Dünnbesetztheit. In den meisten praktischen Fällen ist A nur sehr dünn besetzt; nach einigen Iterationen geht diese Struktur meist verloren und es ist eine zunehmende Befüllung der transformierten Matrix A' erkennbar. Der Rückgriff auf die ursprüngliche Matrix wirkt hier sehr beschleunigend.

Das Verfahren berechnet nun in den nächsten Schritten die Werte der Variablen, die dualen Werte usw., bis schließlich die Optimalität nachgewiesen ist.

Die Berechnung der Basisinversen ist offenbar ein Kernelement des Simplex-Verfahrens und ist auch der Rechenschritt, der die meiste Zeit erfordert. Daher ist es lohnend, diesen Schritt noch einmal genauer ins Auge zu fassen. Moderne Softwareimplementierungen des revidierten Simplexverfahrens berechnen $\mathcal{B}^{-1}$ nicht explizit, sondern verwenden die Produktform

[5] Das Bootsverleihproblem zeigt diese Eigenschaft sehr eindrucksvoll. Hierzu sei das System linearer Gleichungen in jeder Iteration betrachtet. Die mit den Variablen $s_1, \ldots, s_4$ assoziierten Spalten der ursprünglichen Basisvariablen entsprechen der Basisinversen der aktuellen Basis. Dies hat seine Ursache darin, dass die elementaren Zeilenumformungen der Multiplikation mit einer Matrix äquivalent sind, die die Gleichungen durch eine andere Matrix, z. B. M darstellen. Betrachten wir die erste Iteration, so können wir verstehen, wie die Methode arbeitet. Die Anfangsmatrix und insbesondere die Spalten, die zu den neuen Basisvariablen korrespondieren, werden mit M multipliziert und ergeben offensichtlich $\mathcal{B}\cdot\mathsf{M} = \mathbb{1}$, wobei $\mathbb{1}$ wieder die Einheitsmatrix bezeichnet. Ist $\mathcal{B}$ regulär, so muss $\mathsf{M} = \mathcal{B}^{-1}$ gelten. Da alle Spalten von A mit M multipliziert wurden, und speziell also auch die mit den ursprünglichen Basisvariablen assoziierte Einheitsmatrix, ergeben die Spalten gerade die Basisinverse $\mathcal{B}^{-1}$. In jeder Iteration k wird die ursprüngliche Matrix mit einer solchen Matrix M_k multipliziert, sodass die ursprünglichen Basisspalten das Produkt aller Matrizen M_k und somit die Basisinverse der aktuellen Matrix darstellen.

$$\mathcal{B}_k = \mathcal{B}_0\eta_1\eta_2\ldots\eta_k$$

der Basis, um die Basis nach k Iterationen als Funktion der ursprünglichen Basis $\mathcal{B}_0$ (gewöhnlich die Einheitsmatrix) und der sogenannten *Eta-Matrizen* oder *Eta-Faktoren* η_i. Die η_i-Matrizen sind $m \times m$ Matrizen der Gestalt

$$\eta = \mathbb{1} + \mathbf{u}\mathbf{v}^{\mathrm{T}}$$

und leiten sich aus dem dyadischen Produkt $\mathbf{u}\mathbf{v}^{\mathrm{T}}$ der beiden Vektoren $\mathbf{u}$ und $\mathbf{v}$ ab. Dadurch ergibt sich eine sehr einfache Struktur („1" auf der Diagonalen und ansonsten nur von Null verschiedene Einträge in einer einzigen Spalte). Um die η-Matrizen zu speichern, genügt es, die erzeugenden η-Vektoren $\mathbf{u}$ und $\mathbf{v}$ zu speichern. Die Lösung $\mathbf{x}_B = \mathcal{B}^{-1}\mathbf{b}$ der Gleichung $\mathcal{B}\mathbf{x}_B = \mathbf{b}$ lässt sich dann durch

$$\mathbf{x}_B = \eta_k^{-1}\eta_{k-1}^{-1}\ldots\eta_1^{-1}\mathbf{b}$$

darstellen. Die Inversen der η-Matrizen lassen sich unter geeigneten Voraussetzungen leicht mit Hilfe der Formel

$$\eta^{-1} = \mathbb{1} - \frac{1}{1+\mathbf{v}^{\mathrm{T}}\mathbf{u}}\mathbf{u}\mathbf{v}^{\mathrm{T}}$$

berechnen. Da alle η_i-Vektoren gespeichert werden müssen, steigt der Speicherbedarf mit fortlaufenden Iterationen an. Die komplette Neuberechnung der Inversen ist also nicht nur aus numerischen Gründen, sondern auch wegen des limitierten Speichers erforderlich. Lesern, die an diesen Aspekten der linearen Algebra, LU-Zerlegung, η-Vektoren und Erhaltung der Dünnbesetztheit interessiert sind, sei Gill *et al.* (1981,[104], S.192), Padberg (1996, Abschnitt 5.4) und Vanderbei (1996,[282]) zur weiteren Lektüre empfohlen.

Im folgenden Abschnitt wird das Verfahren nun zur Verdeutlichung auf das Bootsverleihproblem angewendet; hierbei ist es notwendig, sorgfältig die Indizes zu verfolgen. Die Basis der Lösung auf Seite 82 ist (x_2, x_1, s_3, s_4), und daher gilt

$$\mathbf{c}_B^{\mathrm{T}} = (600, 800, 0, 0) \quad , \quad \mathcal{B} = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & -1 & 1 & 0 \\ 3 & 4 & 0 & 1 \end{pmatrix} \quad .$$

Zu beachten ist, dass die Spalten der Basismatrix $\mathcal{B}$ den Basisvariablen entsprechen, dabei aber konsistent der Bezug zur ursprünglichen Matrix bzw. den ursprünglichen Gleichungen (4.2.4) bewahrt bleibt. Mit Hilfe der in der linearen Algebra bekannten Formeln folgt die Basisinverse zu

$$\mathcal{B}^{-1} = \begin{pmatrix} 1 & -1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -1 & 2 & 1 & 0 \\ -3 & -1 & 0 & 1 \end{pmatrix} \quad ,$$

woraus wiederum die Schattenpreise

$$\boldsymbol{\pi}^{\mathrm{T}} = (600, 200, 0, 0)$$

folgen. Betrachtet man wieder Seite 82, so stellt man fest, dass sich die Basisinverse der optimalen Lösung direkt aus dem System der linearen Gleichungen nach allen Transformationen ablesen lässt: die Spalten, die der ersten zulässigen Basislösung entsprechen, also die mit den Schlupfvariablen s_1 bis s_4 assoziierten Spalten 3 bis 6, enthalten gerade die Basisinverse der optimalen Lösung.

In Abschnitt A.2 wird das Konzept des zum primalen Problem korrespondierenden dualen Problems eingeführt. Die optimalen Werte der Variablen des dualen Problems entsprechen den reduzierten Kosten und Schattenpreisen des primalen Problems. Daher kann man das auf das primale Problem angewandte Simplexverfahren auch so verstehen, dass hier die Variablen des dualen Problems (bzw. die reduzierten Kosten und Schattenpreise des primalen Problems) iteriert werden und sich das Simplexverfahren implizit zwischen dem primalen und dualen Problem bewegt, wobei die primalen Lösungswerte als auch die reduzierten Kosten ständig verbessert werden.

Mit dem Verständnis der dualen Werte und der Schattenpreise ist es auch möglich, eine weitere Interpretation der reduzierten Kosten als Funktion der Schattenpreise zu geben. Während die dualen Werte bzw. die Lagrange-Multiplikatoren die Kosten für aktive Nebenbedingungen des primalen Problems darstellen, sind die reduzierten Kosten einer Nichtbasisvariablen die Schattenpreise für die Ablösung einer Nichtbasisvariablen von einer ihrer Schranken. Dies erklärt, warum die reduzierten Kosten der Basisvariable den Wert Null haben: sie sind nicht an eine ihrer Schranken fixiert und haben diesbezüglich keine Schattenpreise.

A.1.2 Die Berechnung von Startlösungen

In der bisherigen Erklärung des Simplexverfahrens sind wir stets von der Existenz einer ersten zulässigen Basislösung ausgegangen, die dann fortlaufend bis zum Nachweis der Optimalität iteriert wurde. Hier soll nun gezeigt werden, wie man zu dieser Startlösung kommt. Es gibt verschiedene Verfahren, aber die bekanntesten sind das *Big-M-Verfahren* und Verfahren, die eine *Phase I* und *Phase II* haben. Um die beiden Verfahren zu diskutieren, sei das folgende, in Standardform vorliegende Minimierungsproblem

$$\min \quad \mathbf{c}^{\mathrm{T}}\mathbf{x}$$

mit n Variablen unter den m Nebenbedingungen

$$\mathsf{A}\mathbf{x} = \mathbf{b} \quad , \quad \mathbf{x} \geq \mathbf{0}$$

betrachtet. Dabei kann im Folgenden stets $\mathbf{b} \geq \mathbf{0}$ vorausgesetzt werden, denn andernfalls würde man die entsprechenden Zeilen der Gleichungen $\mathsf{A}\mathbf{x} = \mathbf{b}$ einfach mit -1 multiplizieren. Damit ist es möglich, nichtnegative Variablen $\mathbf{v} = (v_1, \ldots, v_m)$ einzuführen, die die Verletzung einer Gleichung messen, und das Hilfsproblem

$$\min \quad \mathbf{c}^{\mathrm{T}}\mathbf{x} + M \sum_{j=1}^{m} v_j$$

unter den Nebenbedingungen

$$\mathsf{A}\mathbf{x} + \mathbf{v} = \mathbf{b} \quad , \quad \mathbf{x} \geq \mathbf{0} \quad , \quad \mathbf{v} \geq \mathbf{0}$$

zu formulieren, das im Vergleich zum ursprünglichen Problem einen zusätzlichen Strafterm in der Zielfunktion und entsprechend modifizierte Nebenbedingungen enthält. Hierbei ist M eine passend zu wählende und vom Problem abhängige „große" Zahl, z. B. 10^5. Dieses Hilfsproblem hat den Vorteil, dass es durch bloßes Hinschauen erlaubt, eine Startlösung zu bestimmen. Wie durch Einsetzen einfach überprüft werden kann, ist

$$\mathbf{v} = \mathbf{b} \quad , \quad \mathbf{x} = \mathbf{0}$$

nämlich eine zulässige Startlösung. Damit kann das Simplexverfahren auf dieses Hilfsproblem angewendet werden. Wird M hinreichend groß gewählt, so darf gehofft werden, dass für alle Hilfsvariablen $\mathbf{v} = \mathbf{0}$ gilt. Enthält die Lösung dagegen positive Werte v_j, so war M entweder nicht groß genug, oder das ursprüngliche Problem ist unzulässig. Wie kann man nun die Größe des Parameters M passend wählen? Man könnte mit kleinen Werten beginnen und überprüfen, ob von Null verschiedene Hilfsvariablen existieren. Falls nicht, hat man eine Startlösung für das ursprüngliche Problem gefunden, andernfalls sollte man M erhöhen. Hilft auch dies nicht, so ist es ratsam, die nachfolgend beschriebene Zwei-Phasen-Methode zu verwenden, die den Vorteil bietet, dass sie den Parameter M nicht enthält und somit skalenunabhängig ist.

Die Zwei-Phasen-Methode verwendet die Zielfunktion

$$\min \quad \sum_{j=1}^{m} v_j \quad ,$$

also lediglich die Summe der Hilfsvariablen. Ist der Zielfunktionswert der optimalen Lösung von Null verschieden, so ist nachgewiesen, dass das ursprüngliche Problem unzulässig ist. Überprüft man, welche der Hilfsvariablen v_j positiv ist, so kann man möglicherweise, wie in Abschnitt 6.6.1 weiter ausgeführt, bereits einen Hinweis darauf erhalten, was die Unzulässigkeit verursacht.

Warum werden nun aber zwei Verfahren vorgestellt? Würde eines nicht ausreichend sein? Betrachtet man beide Methoden sorgfältig, so stellt man fest, dass beide Vor- und Nachteile haben. Im Grenzwert $M \to \infty$ geht das *Big-M*-Verfahren in die Zwei-Phasen-Methode über. Das *Big-M*-Verfahren ist nicht skaleninvariant, bietet aber die Möglichkeit der adaptiven Anpassung der Parameters M und hat den Vorteil, dass im Erfolgsfall eine zulässige Startlösung bestimmt wird, die der optimalen Lösung des ursprünglichen Problems wahrscheinlich schon näher kommt, da die Variablen $\mathbf{x}$ ebenfalls in der Zielfunktion vertreten sind. In Gebrauch sind auch Mischtypen, die zwischen Zwei-Phasen-Methode und *Big-M*-Verfahren schalten.

Weniger bekannt sind heuristische und der Bezeichnung *Crash-Verfahren* geführte und in kommerzieller LP-Software implementierte Ansätze; siehe hierzu auch Abschnitt 5.11. Diese heuristischen Verfahren bestimmen bereits sehr gute Startlösungen, die nahe an der optimalen Lösung liegen. Im Zusammenhang mit der Bestimmung einer zulässigen Startlösung ist auch erwähnenswert, dass eine zulässige Basislösung, die von einem früheren Rechenlauf des Problems vorliegt, möglichweise nützlich sein kann, wenn die Problemdaten sich nur wenig geändert haben oder nur wenige Variablen oder Nebenbedingungen hinzugetreten sind; hier ist das duale Simplexverfahren sehr geeignet.

Schließlich werden auch *Hybridverfahren*, die mit Hilfe von IPM und anschließendem Cross-Over zum Simplexverfahren wechseln, zur Bestimmung zulässiger Startlösungen eingesetzt.

A.1.3 LP-Probleme mit oberen Schranken

Meist wurden in diesem Buch nur LP-Probleme mit n Variablen und m Nebenbedingungen betrachtet, die als Maximierungs- oder Minimierungsproblem in Standardform

$$\min \quad \mathbf{c}^{\mathrm{T}}\mathbf{x}$$

unter den Nebenbedingungen

$$\mathsf{A}\mathbf{x} = \mathbf{b} \quad , \quad \mathbf{x} \geq \mathbf{0}$$

vorlagen. In vielen praktischen Fragestellungen liegen die Probleme jedoch häufig mit oberen und unteren Schranken vor, d. h.

$$\min \quad \mathbf{c}^{\mathrm{T}}\mathbf{x}'$$

unter den Nebenbedingungen

$$\mathsf{A}\mathbf{x}' = \mathbf{b}' \quad , \quad \mathbf{l}' \leq \mathbf{x}' \leq \mathbf{u}' \quad .$$

Da es stets möglich ist, die Variablensubstitution $\mathbf{x} = \mathbf{x}' - \mathbf{l}'$ durchzuführen, wobei die neuen Variablen $\mathbf{x}$ dann den Schranken $\mathbf{0} \leq \mathbf{x} \leq \mathbf{u}' - \mathbf{l}' = \mathbf{u}$ unterliegen, genügt es, das Problem

$$\min \quad \mathbf{c}^{\mathrm{T}}\mathbf{x}$$

unter den Nebenbedingungen

$$\mathsf{A}\mathbf{x} = \mathbf{b} \quad , \quad \mathbf{0} \leq \mathbf{x} \leq \mathbf{u}$$

zu betrachten. Zwar könnte man mit Hilfe zusätzlicher Schlupfvariablen $\mathbf{s} \geq \mathbf{0}$ das Problem in die Standardform

$$\min \quad \mathbf{c}^{\mathrm{T}}\mathbf{x}$$

unter den Nebenbedingungen

$$\begin{matrix} \mathsf{A}\mathbf{x} = \mathbf{b} \\ \mathbf{x} + \mathbf{s} = \mathbf{u} \end{matrix} \quad , \quad \mathbf{x} \geq \mathbf{0} \quad , \quad \mathbf{s} \geq \mathbf{0}$$

überführen, aber in vielen praktischen Problemen ist $n \gg m$ (häufig hat man mehr als 3 oder 10 mal so viele Variablen wie Nebenbedingungen), sodass die direkte Anwendung des Simplexverfahrens zu einem unnötig großen Problem mit $m + n$ Nebenbedingungen bzw. einer $(m+n) \times (m+n)$ großen Basismatrix führen würde. Numerisch ist es wesentlich günstiger, das Simplexverfahren zu modifizieren und mit einer $m \times m$ Basismatrix zu arbeiten. Die Idee hierbei ist, zwischen Nichtbasisvariablen x_j, $j \in \mathcal{J}_0$, die an ihrer unteren Schranke (also Null) und solchen x_j, $j \in \mathcal{J}_u$, die an ihrer oberen Schranke fixiert sind, zu unterscheiden. Im Folgenden wird ausgeführt, wie in einem modifizierten Simplexverfahren der Basisaustausch funktioniert. Mit Hilfe der reduzierten Kosten können wir nun feststellen, dass die aktuelle Basis noch nicht optimal ist, wenn eine der beiden Bedingungen vorliegt:

$$\begin{matrix} \text{1) es existieren Indizes } j \in \mathcal{J}_0 \text{ mit } d_j < 0 \\ \text{2) es existieren Indizes } j \in \mathcal{J}_u \text{ mit } d_j > 0 \end{matrix} \quad .$$

Im ersten Fall könnten wir nämlich die entsprechende Nichtbasisvariable erhöhen, im zweiten Fall könnten wir sie verkleinern; in beiden Fällen würde der Zielfunktionswert weiter abnehmen, was bei einem Minimierungsproblem wünschenswert ist. Neu bei der Existenz oberer Schranken ist nun aber, dass bei der Änderung einer Nichtbasisvariablen zu überprüfen ist, ob sie bei diesem Prozess die obere bzw. untere Schranke erreicht.

Im Abschnitt 4.2.4 kontrollierte die *Regel des minimalen Verhältnisses*, wie weit eine Nichtbasisvariable verändert werden konnte. Die Änderung war dadurch limitiert, dass eine existierende Basisvariable ihre untere Schranke erreichte, also Null wurde. Bei Existenz einer oberen Schranke sind die Verhältnisse etwas komplizierter, aber folgen grundsätzlich ähnlichen Regeln.

Der Fall 1) führt auf zwei Unterfälle:

- 1a) die Nichtbasisvariable x_j kann bis zur oberen Schranke erhöht werden, wobei keine existierende Basisvariable Null oder ihre obere Schranke, oder
- 1b) während die Nichtbasisvariable x_j erhöht wird, erreicht eine Basisvariable den Wert Null oder ihre obere Schranke.

Der Fall 1a), auch Flip-Fall genannt, ist einfach zu behandeln: es ist lediglich der Index j in die neue Indexmenge $\mathcal{J}_u$ zu übertragen. Wird im Fall 1b) die Null bzw. die obere Schranke erreicht, so verfährt man wie im Standardsimplexverfahren: die Variable x_j wird der Basis hinzugefügt und der Index der Variable, die die Basis verlässt, wird Element der Indexmenge $\mathcal{J}_0$. bzw. $\mathcal{J}_u$.

Der Fall 2) kann analysiert werden, indem man $s_j = u_j - x_j$ betrachtet. Ist die Nichtbasisvariable x_j an ihrer oberen Schranke, so ist s_j an der unteren Schranke, also Null. Die Größe s_j spielt dieselbe Rolle (zu beachten ist, dass u_j ihre obere Grenze ist) wie die Nichtbasisvariable x_j, die in den Fällen 1a) und 1b) betrachtet wurden; daher gelten hier sinngemäß dieselben Argumente.

Die Operationen der linearen Algebra sind gegenüber dem Standardverfahren kaum verändert und erfordern keine nennenswerten Zusatzberechnungen; für weitere Details sei dem interessierten Leser Padberg (1996,[228], S.75-80) empfohlen. Festzuhalten ist, dass zwar einige zusätzliche Tests durchzuführen sind, aber keine signifikanten Zusatzrechnungen. Daher ist dieses modifizierte Simplexverfahren, in dem obere Schranken explizit behandelt werden, wesentlich schneller als die entsprechende Variante, in der obere Schranken als zusätzliche lineare Nebenbedingungen behandelt werden. Ist dieses Simplexverfahren in ein B&B-Verfahren eingebettet, so ist die Zeitersparnis erheblich. Hinzu kommt, dass beim B&B-Verfahren ja lediglich Schranken hinzugefügt werden, die wie nun gezeigt, besonders behandelt werden können.

Allerdings ist noch auf Folgendes hinzuweisen. Da die Schranken explizit und nicht als lineare Nebenbedingungen behandelt werden, werden für die Schranken keine Schattenpreise berechnet. Diese können allerdings aus den reduzierten Kosten derjenigen Variablen berechnet werden, die an ihren oberen Schranken fixiert werden. Dieser Idee liegt zugrunde, dass die Schattenpreise die differentielle Änderung der Zielfunktion bezüglich einer Einheitsänderung der rechten Seite der Nebenbedingungen repräsentieren. Für eine gegebene Nebenbedingung $x + s = u$ entspricht eine Änderung der rechten Seite u einer Änderung der Schlupfvariablen s (oder, x) um denselben Betrag; daher gilt $d_j = \pi$. Dies wird im folgenden Beispiel deutlicher:

Problem (Nebenbedingung)		Standardformulierung
$\begin{array}{rl} \min & -x-y \\ u.d.N. & 2x+y \le 3 \\ & x+2y \le 3 \\ & x \le 0.5 \\ & x \ge 0 \quad , \quad y \ge 0 \end{array}$	$\Longleftrightarrow$	$\begin{array}{rl} \min & -x-y \\ u.d.N. & 2x+y+s_1 = 3 \\ & x+2y+s_2 = 3 \\ & x+s_3 = 0.5 \\ & x \ge 0 \quad , \quad y \ge 0 \end{array}$

. (A.1.9)

Dieses Problem hat die Lösung

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 0.5 \\ 1.25 \end{pmatrix} \quad , \quad \begin{pmatrix} s_1 \\ s_2 \\ s_3 \end{pmatrix} = \begin{pmatrix} 0.75 \\ 0 \\ 0 \end{pmatrix} \quad ,$$

und[6]

$$\begin{pmatrix} d_x \\ d_y \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad , \quad \begin{pmatrix} \pi_1 \\ \pi_2 \\ \pi_3 \end{pmatrix} = \begin{pmatrix} 0 \\ -0.5 \\ -0.5 \end{pmatrix} \quad .$$

Die Variablen x, y und s_1 sind Basisvariablen; die zweite und dritte Ungleichung sind aktiv. Die dritte Nebenbedingung im Problem (A.1.9) kann aber auch als Schranke berücksichtigt werden

$$\begin{array}{rcl} \text{Problem (Schranke)} & & \text{Standardformulierung} \\ \\ \begin{array}{rl} \min & -x-y \\ u.d.N. & 2x+y \le 3 \\ & x+2y \le 3 \\ 0 \le x \le 0.5 & , \quad y \ge 0 \end{array} & \Longleftrightarrow & \begin{array}{rl} \min & -x-y \\ u.d.N. & 2x+y+s_1 = 3 \\ & x+2y+s_2 = 3 \\ 0 \le x \le 0.5 & , \quad y \ge 0 \end{array} \end{array} \quad . \tag{A.1.10}$$

Natürlich erhalten wir bezüglich x und y dieselbe Lösung, aber nun ist nur y eine Basisvariable, x ist eine Nichtbasisvariable an ihrer oberen Schranke und die reduzierten Kosten und die Schattenpreise (von denen gibt es jetzt formal nur zwei) sind verschieden (mit B sind die Werte gekennzeichnet, die sich ergeben, wenn man das Problem unter expliziter Berücksichtigung der Schranke angeht)

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 0.5 \\ 1.25 \end{pmatrix}, \quad \begin{pmatrix} d_x^B \\ d_y^B \end{pmatrix} = \begin{pmatrix} -0.5 \\ 0 \end{pmatrix}, \quad \begin{pmatrix} \pi_1^B \\ \pi_2^B \end{pmatrix} = \begin{pmatrix} 0 \\ -0.5 \end{pmatrix} \quad .$$

Im Beispiel gilt offenbar $\pi_3 = d_x^B$. Lassen sich derartige Beziehungen verallgemeinern, so steht uns ein Weg zur Verfügung, gegebene LP-Probleme mit oberen Schranken durch explizite Berücksichtigung dieser Schranken zu lösen und dennoch Schattenpreise für diese Schranken zu erhalten. Mit einigen algebraischen Manipulationen kann in der Tat gezeigt werden, dass diese Beziehungen gelten. Nachfolgend wird dasselbe Problem mit m Nebenbedingungen (hierbei sind die Schranken nicht gezählt) und n Variablen auf zwei verschiedene Weisen behandelt: einmal wird die Schranke als lineare Nebenbedingung formuliert, einmal dagegen explizit berücksichtigt:

$$\begin{array}{rcl} \text{Problem (Nebenbedingung)} & & \text{Problem (Schranke)} \\ \\ \begin{array}{rl} \min & \mathbf{c}^{\mathrm{T}}\mathbf{x} \\ u.d.N. & \mathsf{A}\mathbf{x}+\mathbf{s}=\mathbf{b} \\ & \mathbf{x}+\mathbf{s}=\mathbf{u} \\ & \mathbf{x} \ge 0 \end{array} & \Longleftrightarrow & \begin{array}{rl} \min & \mathbf{c}^{\mathrm{T}}\mathbf{x} \\ u.d.N. & \mathsf{A}\mathbf{x}+\mathbf{s}=\mathbf{b} \\ & \\ & 0 \le \mathbf{x} \le \mathbf{u} \end{array} \end{array} \quad .$$

Angenommen, in der optimalen Lösungen seien die ersten n_u Variablen an ihrer oberen Grenze (durch passende Spaltenvertauschungen kann dies stets erreicht werden). Im ersten Fall (Darstellung der Schrankenbedingungen als lineare Nebenbedingung) kann die Basismatrix als Blockmatrix

$$\mathsf{B} = \begin{pmatrix} \mathsf{U} & \mathcal{B} \\ \mathbb{1} & 0 \end{pmatrix}$$

[6] Etwas Vorsicht ist geboten bei einigen Softwarepaketen hinsichtlich der Vorzeichenkonvention. `Xpress-MP` verwendet z. B. eine andere Vorzeichenkonvention für die Schattenpreise und gibt als Lösung $\pi_2 = \pi_3 = +0.5$ aus.

dargestellt werden, wobei B eine $(n_u + m)(n_u + m)$ Matrix, U eine $m \times n_u$ Matrix, $\mathbb{1}$ die $n_u \times n_u$ Einheitsmatrix, $\mathsf{0}$ eine $n_u \times m$ Matrix von Nullen und $\mathcal{B}$ (unter geeigneten Voraussetzungen) eine reguläre $m \times m$ Matrix ist, die gleichzeitig auch die Basismatrix der optimalen Lösung des Problems ist, wenn die Schranken explizit berücksichtigt werden. In unserem kleinen Beispiel ist $m = 2$, $n_u = 1$ und

$$\mathsf{B} = \begin{pmatrix} 2 & 1 & 1 \\ 1 & 2 & 0 \\ 1 & 0 & 0 \end{pmatrix} \quad , \quad \mathsf{U} = \begin{pmatrix} 2 \\ 1 \end{pmatrix} \quad , \quad \mathbb{1} = (1) \quad , \quad \mathcal{B} = \begin{pmatrix} 1 & 1 \\ 2 & 0 \end{pmatrix} \quad .$$

Um zu zeigen, dass $\boldsymbol{\pi}_U^{\mathrm{T}}$, die zu der als lineare Nebenbedingung formulierten Schrankenbedingung gehörenden Schattenpreise, die gleichen Werte haben wie die reduzierten Kosten $\mathbf{d}_U^{\mathrm{T}}$ der an die oberen Schranken fixierten Nichtbasisvariablen, bestimmen wir einerseits die Matrix B^{-1} um $\boldsymbol{\pi}_U^{\mathrm{T}}$ zu ermitteln und berechnen andererseits $\mathbf{d}_U^{\mathrm{T}}$ basierend auf $\mathcal{B}$.

Die Basisinverse B^{-1} kann explizit als

$$\mathsf{B}^{-1} = \begin{pmatrix} \mathsf{0} & \mathbb{1} \\ \mathcal{B}^{-1}\mathbb{1} & -\mathcal{B}^{-1}\mathsf{U} \end{pmatrix}$$

geschrieben werden. Die Schattenpreise $\boldsymbol{\pi}_U^{\mathrm{T}}$ folgen aus

$$\boldsymbol{\pi}^{\mathrm{T}} = (\mathbf{c}_U^{\mathrm{T}}, \mathbf{c}_B^{\mathrm{T}}) \begin{pmatrix} \mathsf{0} & \mathbb{1} \\ \mathcal{B}^{-1}\mathbb{1} & -\mathcal{B}^{-1}\mathsf{U} \end{pmatrix} = (\mathbf{c}_B^{\mathrm{T}}\mathcal{B}^{-1}\mathbb{1}, \mathbf{c}_U^{\mathrm{T}} - \mathbf{c}_B^{\mathrm{T}}\mathcal{B}^{-1}\mathsf{U})$$

und ergeben

$$\boldsymbol{\pi}_U^{\mathrm{T}} = \mathbf{c}_U^{\mathrm{T}} - \mathbf{c}_B^{\mathrm{T}}\mathcal{B}^{-1}\mathsf{U} \quad .$$

Die reduzierten Kosten der Nichtbasisvariablen $\mathbf{x}_U$, die an ihren oberen, explizit berücksichtigten Schranken sind, sind nach Formel (A.1.6) aber gerade

$$\mathbf{d}_U^{\mathrm{T}} = \mathbf{c}_U^{\mathrm{T}} - \mathbf{c}_B^{\mathrm{T}}\mathcal{B}^{-1}\mathsf{U} \quad ,$$

womit die Gleichheit $\boldsymbol{\pi}_U^{\mathrm{T}} = \mathbf{d}_U^{\mathrm{T}}$ bewiesen ist.

A.1.4 Das duale Simplexverfahren

Neben dem (*primalen*) *Simplexverfahren*, dass in jedem Iterationsschritt versucht, den Zielfunktionswert durch Übergang von einer gegebenen zulässigen zu einer anderen zulässigen Basislösung zu verbessern, gibt es das *duale Simplexverfahren*. Dieses löst ebenfalls das primale Problem, geht dabei aber von einer dual-optimalen Lösungen aus, d. h. einer Basislösung, die zwar nicht unbedingt primal zulässig ist, deren reduzierte Kosten in (A.1.6) aber das richtige Vorzeichen haben, und iteriert unter Erhaltung der dualen Optimalität so lange, bis die Basislösung auch primal zulässig ist. Beide Verfahren sind im Sinne des Abschnittes A.2 dual zueinander. Während das primale Simplexverfahren zunächst eine neue Basisvariable wählt und dann entscheidet, welche existierende Basisvariable eliminiert wird, eliminiert das duale Verfahren zunächst eine existierende Basisvariable und entscheidet sich dann für eine neu aufzunehmende Variable.

Innerhalb des B&B-Verfahrens wird aus dem folgenden Grund häufig das duale Simplexverfahren verwendet. Im B&B-Verfahren unterscheidet sich ein Unterproblem von dem Ausgangsproblem lediglich durch die Existenz zusätzlicher Schranken auf der Variablen, auf die verzweigt wird. Im Anhang A.1.3 wurde gezeigt, wie innerhalb eines leicht modifizierten Simplexverfahrens Schranken numerisch effizient behandelt werden können. Im Sinne des Unterproblems und des dualen Simplexverfahrens kann die Lösung des Ausgangsproblems als dual optimal und primal unzulässig angesehen werden. In vielen Fällen ist es daher möglich, dass das duale Simplexverfahren mit einer einzigen oder aber doch wenigstens[7] mit nur einigen wenigen Iterationen die Zulässigkeit wieder herstellt. Es erlaubt uns daher in vielen Fällen, existierende Lösungen und Informationen effizient auszunutzen. Verwendet man das primale Simplexverfahren in B&B-Verfahren, so stellt man fest, dass damit wesentlich mehr Rechenzeit erforderlich ist.

A.2 Dualitätstheorie

Die in der LP auftretenden Schattenpreise und reduzierten Kosten zeigen, dass wir bei der Lösung eines LP-Problems nicht nur an den Werten der Variablen interessiert sind. Die Schattenpreise lassen sich einfacher verstehen, wenn wir das mit dem ursprünglichen Problem verknüpfte *duale* LP-Problem betrachten. Damit ergibt sich auch noch ein tieferes Verständnis des Konzeptes der Optimalität. Die formale mathematische Basis bildet das Konzept der Dualität, das sich als sehr fruchtbar erweist bei theoretischen Betrachtungen, aber auch bei der Verbesserung der numerischen Eigenschaften von Algorithmen zur Lösung nichtlinearer, linearer und gemischt-ganzzahliger Optimierungsprobleme. Effiziente IPM [siehe z. B. Anhang A.3] könnten z. B. nicht ohne das Konzept der Dualität verstanden werden. Schließlich erlaubt die Dualitätstheorie der Durchführung von *Sensitivitätsanalysen*, d. h. eine Abschätzung der Auswirkung kleiner Änderungen in den Eingangsdaten des Problems auf unsere Lösung.

A.2.1 Konstruktion des dualen Problems in der linearen Programmierung

Im Falle der LP ist es recht einfach, das duale Problem zu konstruieren. Hierzu sei das LP-Problem

$$\mathbf{LP}: \quad \max \quad 2x_1 + 3x_2 + x_3$$

unter den Nebenbedingungen

$$\begin{array}{rcrcrcl} 2x_1 & + & x_2 & + & x_3 & \leq & 20 \\ x_1 & + & 2x_2 & & & \leq & 30 \end{array} \quad , \quad x_1, x_2, x_3 \geq 0$$

betrachtet. Das Problem **DP**,

[7] Das muss natürlich nicht so sein und kann auch nicht garantiert werden, denn wäre es so, wäre das duale Simplexverfahren ein polynomiales Verfahren zur Lösung von LP-Problemen. Die Praxis ist dennoch in vielen Fällen gutartig und viele Probleme in der Praxis lassen sich damit gut lösen.

$$\mathbf{DP}: \quad \min \quad 20y_1 + 30y_2$$

unter den Nebenbedingungen

$$\begin{array}{rcrcl} 2y_1 & + & y_2 & \geq & 2 \\ y_1 & + & 2y_2 & \geq & 3 \\ y_1 & & & \geq & 1 \end{array} \quad , \quad y_1 \geq 0 \quad , \quad y_2 \geq 0$$

ist mit dem ursprünglichen Problem **LP** assoziiert und wird das zu **LP** duale Problem genannt; das ursprüngliche Problem heißt *primales* Problem. Beide Probleme sind in folgender Weise miteinander verbunden:

1. Ist das **LP**-Problem ein Maximierungsproblem und sind alle Nebenbedingung $\leq$ Ungleichungen, so ist das **DP**-Problem ein Minimierungsproblem mit $\geq$ Ungleichungen.
2. Die Koeffizienten der Zielfunktion des **LP**-Problems werden (bei Beibehaltung der Ordnung) die Koeffizienten der rechten Seite der Nebenbedingungen **DP**-Problems.
3. Entsprechend werden die Koeffizienten der rechten Seite in **LP** (bei Beibehaltung der Ordnung) die Koeffizienten der Zielfunktion in **DP**.
4. Die linksseitigen Koeffizienten der Nebenbedingungen des Problems **LP** in horizontaler Richtung gelesen, entsprechen den linksseitigen Koeffizienten des Problems **DP** in vertikaler Lesweise. Die mit der linearen Algebra vertrauten Leser werden dies als transponierte Matrix erkennen.
5. Als Folge der Eigenschaft 4 führt also ein **LP**-Problem mit m Nebenbedingungen und n Variablen auf ein duales Problem **DP** mit n Nebenbedingungen und m Variablen. Die Variablen des dualen Problems, naheliegenderweise *duale Variablen* genannt, sind gerade die in Abschnitt 3.3.5.1 bereits aufgetretenden Schattenpreise.

Aus der engen Verknüpfung des primalen und dualen Problems folgt in der LP [siehe Abschnitt A.2.3] das wichtige Resultat, dass, sofern eine optimale Lösung existiert, die optimalen Werte der Zielfunktionen in beiden Problemen übereinstimmen[8]. Dieses Resultat findet im *starken Dualitätssatz* seinen Niederschlag. Konstruiert man das zu **DP** duale Problem, so erhält man wieder ein zu **LP** äquivalentes Problem. In der Literatur findet man je nach Vorliebe des Autors hinsichtlich Zeilen- und Spaltenvektoren die beiden folgenden, äquivalenten[9] Vorschriften zur Konstruktion des dualen Problems:

primales Problem		duales Problem 1		duales Problem 2
$\max \ \mathbf{c}^{\mathrm{T}}\mathbf{x}$		$\min \ \mathbf{b}^{\mathrm{T}}\mathbf{y}$		$\min \ \mathbf{y}^{\mathrm{T}}\mathbf{b}$
u.d.N. $\mathsf{A}\mathbf{x} \leq \mathbf{b}$	$\longleftrightarrow$	*u.d.N.* $\mathsf{A}^{\mathrm{T}}\mathbf{y} \geq \mathbf{c}$	$\Longleftrightarrow$	*u.d.N.* $\mathbf{y}^{\mathrm{T}}\mathsf{A} \geq \mathbf{c}^{\mathrm{T}}$
$\mathbf{x} \geq 0$		$\mathbf{y} \geq 0$		$\mathbf{y}^{\mathrm{T}} \geq 0$

. (A.2.1)

In der ersten Formulierung wird der Vektor $\mathbf{y}$ der dualen Variablen als Spaltenvektor angesehen und die Daten des primalen Problems werden transponiert, um das duale Problem im Spaltenvektorraum (der primale Vektorraum) zu formulieren. In der zweiten Formulierung sind die dualen Variablen und alle Nebenbedingungen Objekte im dualen Zeilenvektorraum; die ursprünglichen Daten bleiben unverändert.

[8] Zu beachten ist, dass diese Eigenschaften bei MILP-Problemen *nicht* gilt.

[9] Unter Verwendung der Resultate $\mathbf{u}^{\mathrm{T}}\mathbf{v} = \left(\mathbf{u}^{\mathrm{T}}\mathbf{v}\right)^{\mathrm{T}} = \mathbf{v}^{\mathrm{T}}\mathbf{u}$ und $\left(\mathbf{y}^{\mathrm{T}}\mathsf{A}\right)^{\mathrm{T}} = \mathsf{A}^{\mathrm{T}}\mathbf{y}$ aus der linearen Algebra wird ersichtlich, dass beide Darstellungen des dualen Problems äquivalent sind.

Das nächste Beispiel zeigt die primal-duale Formulierung für die in Abschnitt 4.2.1 gegebene Standardformulierung

primales Problem		duales Problem 1		duales Problem 2
$\max \quad \mathbf{c}^{\mathrm{T}}\mathbf{x}$		$\min \quad \mathbf{b}^{\mathrm{T}}\mathbf{y}$		$\min \quad \mathbf{y}^{\mathrm{T}}\mathbf{b}$
u.d.N. $\mathsf{A}\mathbf{x} = \mathbf{b}$	$\longleftrightarrow$	*u.d.N.* $\mathsf{A}^{\mathrm{T}}\mathbf{y} + \mathbf{s} = \mathbf{c}$	$\Longleftrightarrow$	*u.d.N.* $\mathbf{y}^{\mathrm{T}}\mathsf{A} + \mathbf{s}^{\mathrm{T}} = \mathbf{c}^{\mathrm{T}}$
$\mathbf{x} \geq 0$		$\mathbf{s} \geq 0$		$\mathbf{s}^{\mathrm{T}} \geq 0$

.

Die duale Variable $\mathbf{y}$ ist nun eine freie Variable, d. h. sie ist unbeschränkt im Vorzeichen. Das duale Problem enthält die Überschussvariable $\mathbf{s}$, wodurch auch das duale Problem in Standardform erscheint, also nur Gleichungsnebenbedingungen enthält.

Damit können duale LP-Probleme nach folgenden Regeln konstruiert:

primales Problem (max)			*duales Problem* (min)
A	Koeffizientenmatrix	$\leftrightarrow$	A^{T} transponierte Matrix
$\mathbf{b}$	Vektor der rechten Seite	$\leftrightarrow$	$\mathbf{c}$ Zielfunktionsvektor
$\mathbf{c}^{\mathrm{T}}$	Zielfunktionsvektor	$\leftrightarrow$	$\mathbf{b}^{\mathrm{T}}$ transponierter Vektor der rechten Seite
i^{th}	Gleichungsbedingung	$\leftrightarrow$	y_i y_i wird freie Variable
i^{th}	$\leq$ Ungleichung	$\leftrightarrow$	y_i y_i wird nichtnegative Variable
i^{th}	$\geq$ Ungleichung	$\leftrightarrow$	y_i y_i wird nichtpositive Variable
x_j	freie Variable	$\leftrightarrow$	j^{th} die duale Bedingung wird eine Gleichung
x_j	nichtnegative Variable	$\leftrightarrow$	j^{th} die duale Bedingung $\geq$ Ungleichung
x_j	nichtpositive variable	$\leftrightarrow$	j^{th} die duale Bedingung $\leq$ Ungleichung

.

A.2.2 Interpretation des dualen Problems

Zur Illustrierung und Vertiefung des Konzeptes der Dualität sei das duale Bootsverleihproblem [siehe (4.2.1) und (4.2.2)] betrachtet:

$$\min \quad 350y_1 + 200y_2 + 1400y_4$$

unter den Nebenbedingungen

$$\begin{array}{rcrcrcrcl} y_1 & + & y_2 & - & y_3 & + & 4y_4 & \geq & 800 \\ y_1 & & & + & y_3 & + & 3y_4 & \geq & 600 \end{array} \quad ; \quad y_1, y_2, y_3, y_4 \geq 0 \quad . \qquad \text{(A.2.2)}$$

Um die in (A.2.1) zusammengefassten Regeln anwenden zu können, wurde die dritte Ungleichung in (4.2.2) zu $-p + s \leq 0$ umgeformt, um somit nur $\leq$ Ungleichungen zu haben; da die rechte Seite hier Null ist, taucht die assoziierte duale Variable y_3 nicht in der Zielfunktion auf. Aus der Dimensionsbetrachtung der Ungleichungen (A.2.2) folgt, dass y_1, y_2 und y_3 die Einheiten *Geld/Boot* haben. Zur Ableitung der Einheit von y_4 muss beachtet werden, dass die Koeffizienten 4 und 3 des primalen Problems die in den Einheiten Stunde/Boot angegeben sind. Daher hat y_4 die Einheit Geld/Stunde. Die Werte 350, 200, 0 und 1400 sind schließlich in den Einheiten Boot, Boot, Boot und Stunden gemessen. Hieraus folgt der Schluss: das primale Problem bestand in der Maximierung des Erlöses, das Duale minimiert Kosten; die Zielfunktion des dualen Problems wird ebenfalls in Geldeinheiten gemessen. Welche Kosten werden aber hier minimiert? Die Kostenkoeffizienten sind durch die rechte Seiten der primalen Nebenbedingungen gegeben. Da nach

dem starken Dualitätssatz die Zielfunktionswerte der primalen und dualen Zielfunktion im optimalen Punkt identisch sind, folgt, dass bei Relaxierung der primalen Nebenbedingungen z. B. von 350 auf 351 die Kosten des dualen Problems sich um y_1 verringern, sich der Erlös des primalen Problems jedoch um denselben Betrag y_1 erhöht. Daher trägt die duale Variable y_i die Information, was uns die Nebenbedingung i tatsächlich kostet; sie entsprechen somit den schon früher eingeführten Schattenpreisen. Insbesondere impliziert eine nicht aktive Ungleichungsbedingung eine verschwindende duale Variable; die mit Ungleichungen verbundene duale Variable kann also nur von Null verschieden sein, wenn die Ungleichung als Gleichung erfüllt wird. Diese Eigenschaft ist auch als komplementäre Schlupfbedingung [engl.: *complementary slackness condition*] bekannt [siehe Abschnitt A.2.3], d. h.

$$y_i^{\mathrm{T}} s_i = 0 \quad ,$$

wobei y_i und s_i die dualen Variablen und Schlupfvariablen der Ungleichung i bezeichnen. Die hier in der LP auftretende komplementäre Schlupfbedingung ist ein Spezialfall; sie wird später in der nichtlinearen Programmierung allgemeiner auftreten und die dualen Variablen werden Lagrange-Multiplikatoren genannt werden.

Beziehen wir uns kurz auf Anhang A.1.1, so können wir auch die Beschränkungen des dualen Problems interpretieren. Die hinreichenden Bedingungen (A.1.7) für die Optimalität eines Maximierungsproblems lauten in Vektorschreibweise

$$\boldsymbol{\pi}^{\mathrm{T}} \mathsf{A} \geq \mathbf{c} \quad .$$

Die Identifikation $\mathbf{y} = \boldsymbol{\pi}$ zeigt, dass die Nebenbedingungen des dualen Problems gerade die hinreichenden Bedingungen für die Optimalität des primalen Problems sind.

Schließlich sei die Lösung des dualen Problems betrachtet: hier finden wir $y_1 = £600/\text{Boot}$, $y_2 = £200/\text{Boot}$, $y_3 = £0/\text{Boot}$ und $y_4 = £0/\text{Stunde}$, was auf einen Zielfunktionswert von $£250,000$ führt, der genau dem primalen Zielfunktionswert im Optimum entspricht.

A.2.3 Dualität und Komplementarität

Um die Verbindung zwischen dem primalen und dualen Problem tiefer zu verstehen, sei die Differenz der Zielfunktionswerte beider Probleme betrachtet. Das im Beispiel (A.2.1) konstruierte Paar erlaubt uns, einige wichtige Resultate aus der Dualitätstheorie abzuleiten.

Sind $\mathbf{x}$ und $\mathbf{y}$ zulässige Punkte des primalen und dualen Problems, so heißt die Größe

$$\Delta(\mathbf{x}, \mathbf{y}) = \mathbf{b}^{\mathrm{T}}\mathbf{y} - \mathbf{c}^{\mathrm{T}}\mathbf{x} = \mathbf{y}^{\mathrm{T}}\mathbf{b} - \mathbf{c}^{\mathrm{T}}\mathbf{x} \tag{A.2.3}$$

Dualitätslücke [engl.: *duality gap*]. Handelt es sich beim primalen Problem um ein Maximierungsproblem, so kann der schwache Dualitätssatz als

$$\Delta(\mathbf{x}, \mathbf{y}) \geq 0$$

formuliert werden. Dies wird ersichtlich, wenn man die zweite Formulierung des dualen Problems in Beispiel 2 betrachtet; seien hier $\mathbf{x} \geq \mathbf{0}$ und $\mathbf{y}^{\mathrm{T}} \geq \mathbf{0}$ zulässige Punkte des primalen und dualen Problems. Die Ungleichungsketten

$$\mathsf{A}\mathbf{x} \leq \mathbf{b} \quad \Longrightarrow \quad \mathbf{y}^{\mathrm{T}}\mathsf{A}\mathbf{x} \leq \mathbf{y}^{\mathrm{T}}\mathbf{b}$$

und

$$\mathbf{y}^{\mathrm{T}}\mathsf{A} \geq \mathbf{c}^{\mathrm{T}} \quad \Longrightarrow \quad \mathbf{y}^{\mathrm{T}}\mathsf{A}\mathbf{x} \geq \mathbf{c}^{\mathrm{T}}\mathbf{x}$$

führen direkt auf $\mathbf{c}^{\mathrm{T}}\mathbf{x} \leq \mathbf{y}^{\mathrm{T}}\mathbf{b}$, woraus $\Delta(\mathbf{x}, \mathbf{y}) \geq 0$ folgt.

Aus dem schwachen Dualitätssatz können wir folgern, dass das duale Problem eine obere Schranke für das primales Maximierungsproblem liefert. Umgekehrt ergibt sich aus dem primalen Maximierungsproblem eine untere Schranke für das duale Minimierungsproblem. Diese Eigenschaft erlaubt wiederum den Schluss: ist das primale Maximierungsproblem zulässig und unbeschränkt, so ist das duale Minimierungsproblem unzulässig; umgekehrt entspricht ein zulässiges aber unbeschränktes duales Minimierungsproblem einem unzulässigen primalen Problem.

Der starke Dualitätssatz [engl.: *strong duality theorem*] besagt, dass, sofern primales und duales Problem zulässig sind, die optimalen Lösungen $\mathbf{x}^*$ und $\mathbf{y}^*$ existieren und die Dualitätslücke verschwindet, d. h.

$$\Delta = \Delta(\mathbf{x}^*, \mathbf{y}^*) = 0 \quad .$$

Die Existenz der optimalen Lösung folgt aus der Kompaktheit des zulässigen Bereichs und der Tatsache, dass jede stetige Funktion auf einer kompakten Menge ihr Optimum annimmt. Für den Beweis der Aussage $\Delta(\mathbf{x}^*, \mathbf{y}^*) = 0$ sei z. B. auf Padberg (1996,[228], S.87) verwiesen.

Gilt umgekehrt $\Delta(\mathbf{x}', \mathbf{y}') = 0$ für einige zulässige Punkte $\mathbf{x}'$ und $\mathbf{y}'$, so sind diese Punkte auch optimale Lösungen, d. h. $\mathbf{x}^* = \mathbf{x}'$ und $\mathbf{y}^* = \mathbf{y}'$. Diese Aussage hat insbesondere wichtige Auswirkungen auf Innere-Punkte-Methoden, denn zulässige Punkte $(\mathbf{x}', \mathbf{y}')$ mit von Null verschiedenem $\Delta(\mathbf{x}', \mathbf{y}')$ sind Punkte im Innern des zulässigen Bereichs und der Wert $\Delta(\mathbf{x}', \mathbf{y}')$ ist ein Maß für den Abstand des Punktes $(\mathbf{x}', \mathbf{y}')$ von einer optimalen Lösung. Daher kann $\Delta(\mathbf{x}', \mathbf{y}')$ und einige daraus abgeleitete Größen als Abbruchkriterium in IPM verwendet werden.

Aus der Dualität folgt weiter eine Eigenschaft über den Komplementaritätsschlupf. Definieren wir die primalen und dualen Schlupfvariablen $\mathbf{s}$ und $\mathbf{w}$

$$\mathbf{s} = \mathbf{b} - \mathsf{A}\mathbf{x} \quad , \quad \mathbf{w}^{\mathrm{T}} = \mathbf{y}^{\mathrm{T}}\mathsf{A} - \mathbf{c}^{\mathrm{T}} \quad ,$$

so folgt für die zulässigen Punkte $\mathbf{x}$ und $\mathbf{y}$ die Bedingung

$$\mathbf{w}^{\mathrm{T}}\mathbf{x} + \mathbf{s}^{\mathrm{T}}\mathbf{y} = \mathbf{0} \quad \Longleftrightarrow \quad \Delta(\mathbf{x}, \mathbf{y}) = 0 \quad , \tag{A.2.4}$$

die die Komplementaritätsschlupfbedingung mit

$$\mathbf{w}^{\mathrm{T}}\mathbf{x} + \mathbf{s}^{\mathrm{T}}\mathbf{y} = \mathbf{0} \tag{A.2.5}$$

mit der Dualitätslücke verbindet.

(A.2.4) bietet die Interpretation, dass aus der komplementären Schlupfbedingung für ein Paar zulässiger Punkte $\mathbf{x}$ und $\mathbf{y}$ des primalen und dualen Problems die Optimalität dieser Punkte folgt.

Da soweit nur nichtnegative Variablen betrachtet wurden, folgt aus $\mathbf{w}^{\mathrm{T}}\mathbf{x} + \mathbf{s}^{\mathrm{T}}\mathbf{y} = \mathbf{0}$

$$\mathbf{w}^{\mathrm{T}}\mathbf{x} = \mathbf{0} \quad , \quad \mathbf{s}^{\mathrm{T}}\mathbf{y} = \mathbf{0} \quad ,$$

und weiter die komponentenweise Bedingung

$$w_j x_j = 0 \quad , \quad \forall j$$

und

$$s_i y_i = 0 \quad , \quad \forall i \quad ,$$

die wir bereits in Abschnitt A.2.2 antrafen.

Der Beweis von (A.2.4) folgt durch Einsetzen aller Terme

$$\mathbf{w}^{\mathrm{T}}\mathbf{x} + \mathbf{s}^{\mathrm{T}}\mathbf{y} = \left(\mathbf{y}^{\mathrm{T}}\mathsf{A} - \mathbf{c}^{\mathrm{T}}\right)\mathbf{x} + \left(\mathbf{b} - \mathsf{A}\mathbf{x}\right)^{\mathrm{T}}\mathbf{y} = \mathbf{y}^{\mathrm{T}}\mathbf{b} - \mathbf{c}^{\mathrm{T}}\mathbf{x} = \Delta(\mathbf{x}, \mathbf{y})$$

und führt auf die Komplementaritätslücke [engl.: *complementarity gap*], $g(\mathbf{x}, \mathbf{y})$,

$$g(\mathbf{x}, \mathbf{y}) = \mathbf{w}^{\mathrm{T}}\mathbf{x} + \mathbf{s}^{\mathrm{T}}\mathbf{y} \quad , \tag{A.2.6}$$

die für alle zulässigen Punkte mit der Dualitätslücke $\Delta(\mathbf{x}, \mathbf{y})$ identisch ist

$$g(\mathbf{x}, \mathbf{y}) = \Delta(\mathbf{x}, \mathbf{y}) \quad .$$

A.3 Innere-Punkte-Methoden — Eine detaillierte Beschreibung

Wie bereits in Abschnitt 4.1.2 kurz erwähnt wurde, initiierte die Arbeit von Karmarkar (1984,[171]) die Entwicklung einer Vielzahl von *Innere-Punkte-Methoden* [siehe z. B. Gonzaga (1992,[114]) und Lustig *et al.* (1992,[199])]. Insbesondere Mehrotras [209] so genannte *primal-duale Prädiktor-Korrektor-Verfahren* [engl.: *second-order predictor-corrector methods*] haben Einzug in kommerzielle LP-Softwarepakete gefunden, wie z. B. `Xpress-MP` oder das `CPLEX` [siehe hierzu Bixby (2002,[39])]. IPM eignen sich besonders zur Lösung von großen dünnbesetzten LP-Problemen oder solchen, die hochgradig entartet sind; hierbei kann erheblich Rechenzeit eingespart werden. Hinsichtlich der Lösungsstrategie lassen sich die Verfahren, wie z. B. von Freund & Mizuno (1996,[95]) vorgenommen, in

1. *Affine Skalierungsverfahren* [engl.: *affine scaling methods*[10]],
2. *Potential-Reduzierungsverfahren* [engl.: *potential reduction methods*[11]], und
3. *Zentrale Pfad-Verfahren* [engl.: *central trajectory methods*[12]].

einteilen. Bei der Anwendung auf LP-Probleme besteht die Grundidee von IPM darin, ausgehend von einem zulässigen inneren Punkt $\mathbf{x} \in S$, $\mathbf{x} > 0$ des zulässigen Bereichs, sich iterativ dem Optimum auf dem Rand bzw. einer Ecke zu nähern, wobei die iterierten Punkte stets innere Punkte bleiben und nie exakt auf dem Rand sein können. Die Ungleichungsbedingung $\mathbf{x} > 0$ im zweiten und dritten Verfahren wird durch einen Strafterm in der Zielfunktion sichergestellt.

[10] *Affine Skalierungs-Verfahren* basieren auf einer Reihe von strikt zulässigen Punkten des primalen Problems. Um diese Punkte herum wird der zulässige Bereich lokal durch das sogenannte *Dikin-Ellipsoid* approximiert. Dieser Ansatz führt auf ein konvexes Optimierungsproblem, das bis auf *eine* quadratische Nebenbedingung linear ist.

[11] Potential-Reduzierungsverfahren wurde in Karmarkars berühmter Arbeit von 1984 eingeführt. Die Zielfunktion ist die Potentialfunktion $q \ln\left(\mathbf{c}^{\mathrm{T}}\mathbf{x} - \mathbf{b}^{\mathrm{T}}\mathbf{y}\right) - \sum_{j=1}^{n} \ln x_j$, die aus dem Logarithmus der Dualitätslücke und einem logarithmischem Term besteht, der dafür sorgt, dass nie der Rand des zulässigen Bereichs erreicht wird. Hinzu kommen noch die linearen Nebenbedingungen für primale und duale Zulässigkeit.

[12] *Zentrale Trajektorien-Verfahren* sind primal-duale Methoden, die auf dem Konzept des *zentral Pfades* oder der *zentralen Trajektorie* beruhen. In ihrer primal-dualen Prädiktor-Korrektor Version gehören sie zu den effizientesten, und nach Freund & Mizuno (1996,[95]) auch zu den ästhetischsten Verfahren.

Um die wesentlichen Eigenschaften zentraler Pfad-Verfahren detaillierter zu erläutern, sei das logarithmische Barriere-Verfahren [engl.: *logarithmic barrier method*] anhand des folgenden, sowohl primal als auch dual formulierten Problems

$$\begin{array}{ccc} \text{primales Problem} & \longleftrightarrow & \text{duales Problem} \\ \\ \begin{array}{rl} \min & \mathbf{c}^{\mathrm{T}}\mathbf{x} \\ u.d.N. & \mathsf{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq 0 \end{array} & & \begin{array}{rl} \max & \mathbf{b}^{\mathrm{T}}\mathbf{y} \\ u.d.N. & \mathsf{A}^{\mathrm{T}}\mathbf{y} + \mathbf{w} = \mathbf{c} \\ & \mathbf{w} \geq 0 \end{array} \end{array} \tag{A.3.7}$$

betrachtet. Hierin bezeichnet $\mathbf{y}$ die freie (vorzeichenunbeschränkte) duale Variable, $\mathbf{w}$ ist die duale Schlupfvariable und mit $\mathbf{x}^*$, $\mathbf{y}^*$ und $\mathbf{w}^*$ sind die Lösungsvektoren bezeichnet. Ein zulässiger Punkt $\mathbf{x}$ ($\mathbf{w}$) des primalen (dualen) Problems heißt strikt zulässig, wenn $\mathbf{x} > 0$ ($\mathbf{w} > 0$) gilt. Das primale Problem wird nun mit Hilfe des Homotopieparameters μ auf eine Folge nichtlinearer Optimierungsprobleme

$$P^{(k)} : \min \left\{ \mathbf{c}^{\mathrm{T}}\mathbf{x} - \mu \sum_{j=1}^{n} \ln\ x_j \;\middle|\; \begin{array}{c} \mathsf{A}\mathbf{x} = \mathbf{b} \\ \mathbf{x} > 0 \end{array} \quad , \quad \mu = \mu^{(k)} \right\}$$

abgebildet, die ursprüngliche Nichtnegativitätsbedingung $\mathbf{x} \geq 0$ durch den logarithmischen Strafterms modelliert. Anstelle von $x_j > 0$ könnte in der Straffunktion auch die Ungleichung $\mathbf{A}_i\mathbf{x} \leq b_i$ durch $\ln(b_i - \mathbf{A}_i\mathbf{x})$ berücksichtigt werden.

In jedem Iterationsschritt k wird μ mit Hilfe der auf Seite 334 beschriebenen Heuristik neugewählt. Der Strafterm – und damit auch die Zielfunktion – wächst gegen unendlich. Wird der Homotopieparameter $\mu > 0$ passend reduziert, so nimmt der Beitrag des Strafterms ab und die Folge der inneren Punkte kann sich dem Rand nähern; daher der Name logarithmisches Barriere-Verfahren. Durch Wahl von $\mu^{(k)}$ wird somit eine Folge $P^{(k)}$ von Minimierungsproblemen konstruiert, sodass schließlich

$$\lim_{k\to\infty}\ \mu^{(k)} \sum_{j=1}^{n} \ln\ x_j = 0$$

und

$$\lim_{k\to\infty}\ argmin(P^{(k)}) = argmin(LP) = x^*$$

erfüllt sind, wobei die Funktion *argmin* den optimalen Lösungsvektor des Problems als Bildwert hat.

Statt des ursprünglichen Minimierungsproblems (A.3.7) sind nun also mehrere NLP-Probleme zu lösen. IPM gehören zur Klasse pfadverfolgender Methoden bzw. spezieller Homotopieverfahren und lassen sich auf allgemeine beschränkte NLP-Probleme anwenden. Wendet man die im Abschnitt 4.4.2 beschriebenen KKT-Bedingungen [Karush (1939,[174]) oder Kuhn & Tucker (1951,[188])] – dies sind die notwendigen und auch hinreichenden Bedingungen für die Existenz lokaler Optima – an, so erhält man ein System nichtlinearer Gleichungen, das z. B. wie nachstehend ausgeführt mit Hilfe des Newton-Verfahrens gelöst werden kann. Hierbei kann ausgenutzt werden, dass die Probleme $P^{(k)}$ bzw. die Systeme nichtlinearer Gleichungen, die ja nur Teilprobleme sind, nicht exakt gelöst werden müssen; meist reichen ein oder zwei Newton-Iterationen in jedem Iterationsschritt k zur Bestimmung eines neuen inneren Punktes.

A.3.1 Primal-duale Innere-Punkte-Methoden

Bis hierin wurde nur das primale Problem in die Betrachtung einbezogen. Um duale und schließlich primal-duale[13] Varianten von IPM zu erhalten (die Motivation dafür wird weiter unten nachgeliefert), werden zunächst wie üblich in der nichtlinearen Optimierung die KKT-Bedingungen von der Lagrange-Funktion

$$L = L(\mathbf{x}, \mathbf{y}) = \mathbf{c}^{\mathrm{T}}\mathbf{x} - \mu \sum_{j=1}^{n} \ln\ x_j - \mathbf{y}^{\mathrm{T}}(\mathsf{A}x - \mathbf{b}) \tag{A.3.8}$$

abgeleitet. Die Nebenbedingungen werden dabei mit den dualen Werten (Lagrange-Multiplikatoren) $\mathbf{y}^{\mathrm{T}}$ multipliziert und dann zur Zielfunktion addiert. Mit[14]

$$\mathbf{e} = [1, ..., 1]^{\mathrm{T}} \in \mathbb{R}^n\ ,\ \mathsf{X} = diag(x_1, ..., x_n)\ ,\ \mathsf{X}^{-1} = diag(x_1^{-1}, ..., x_n^{-1})$$

als Abkürzungen lauten die KKT-Bedingungen:

$$\frac{\partial L}{\partial \mathbf{x}} = \mathbf{c} - \mu \mathsf{X}^{-1}\mathbf{e} - \mathsf{A}^{\mathrm{T}}\mathbf{y} = 0 \tag{A.3.9}$$

$$\frac{\partial L}{\partial \mathbf{y}} = \mathsf{A}\mathbf{x} - \mathbf{b} = 0 \tag{A.3.10}$$

$$x \geq 0 \quad . \tag{A.3.11}$$

Das System (A.3.9-A.3.10) besteht zunächst aus der primalen Zulässigkeitsbedingung

$$\mathsf{A}\mathbf{x} = \mathbf{b} \quad , \tag{A.3.12}$$

zusätzlich aus der Zulässigkeitsbedingung des dualen Problems

$$\mathsf{A}^{\mathrm{T}}\mathbf{y} + \mathbf{w} = \mathbf{c} \tag{A.3.13}$$

und schließlich aus den *gestörten* komplementären Schlupfbedingungen

$$\mathsf{X}\mathsf{W}\mathbf{e} = \mu\mathbf{e} \quad , \quad \mathsf{W} := diag(w_1, ..., w_n) \quad . \tag{A.3.14}$$

Die aus der Lagrange-Funktion (A.3.8) abgeleiteten KKT-Bedingungen reproduzieren somit unsere ursprünglichen Gleichungen des primal-dual Paars (A.3.7) und liefern zusätzlich die nichtlineare Gleichung (A.3.14), die vom Homotopieparameter abhängt.

Dies erlaubt die folgende Interpretation. Das System der Gleichungen (A.3.12)-(A.3.13) entspricht dem KKT-System des ursprünglichen LP-Problems, in dem die komplementären Schlupfbedingungen durch μ gestört sind. Eine nichtnegative Lösung von (A.3.12)-(A.3.13) heißt *analytisches Zentrum* und hängt vom Barriereparameter μ ab. Die Menge der Lösungspunkte $[\mathbf{x}(\mu), \mathbf{y}(\mu), \mathbf{w}(\mu)]$ definiert eine Trajektorie von Zentren des primalen bzw. dualen Problems und wird *zentraler Pfad* oder *zentrale Trajektorie* genannt.

Die Punkte des zentralen Pfades und der Barriereparameter μ genügen der Beziehung

[13] Dieser Name findet seine Begründung darin, dass sowohl die primalen als auch die dualen Variablen auftreten.

[14] Mit $diag(x_1, \ldots, x_n)$ wird die Matrix bezeichnet, deren Diagonalelemente $x_1, \ldots, x_n$ sind und deren Nichtdiagonalelemente alle den Wert Null haben.

$$\mathbf{w}^{\mathrm{T}}\mathbf{x} = g(\mu) = \Delta(\mathbf{x},\mathbf{y}) = 2\mu\mathbf{e}^{\mathrm{T}}\mathbf{e} = 2n\mu \quad . \tag{A.3.15}$$

Die Gleichung (A.3.15) zeigt, dass in Abwesenheit des Barriereproblems die Dualitätslücke verschwindet. Nähert sich μ der Null, so nähern sich sowohl die Komplementaritätslücke als auch die Dualitätslücke der Null, was nach dem starken Dualitätssatz bedeutet, dass sich die Punkte der optimalen Lösung des LP-Problems (A.3.7) annähern.

Setzen wir (A.3.15) in (A.3.14) ein, so erhalten wir eine parameterfreie Darstellung des zentralen Pfades, d. h. der Menge aller Lösungspunkte des Gleichungssytems

$$\mathsf{A}\mathbf{x} = \mathbf{b} \quad , \quad \mathsf{A}^{\mathrm{T}}\mathbf{y} + \mathbf{w} = \mathbf{c} \quad , \quad \mathsf{X}\mathsf{W}\mathbf{e} = \frac{\mathbf{w}^{\mathrm{T}}\mathbf{x}}{2n}\mathbf{e} \quad , \quad \begin{matrix}\mathbf{x} > \mathbf{0}\\ \mathbf{w} > \mathbf{0}\end{matrix} \quad .$$

im Folgenden wird nun aber gezeigt, wie das Gleichungssytem (A.3.12)-(A.3.14) gelöst werden kann. Gleichungssysteme $\mathbf{f}(\mathbf{z}^*) = 0$ lassen sich mit Hilfe des Newton-Verfahrens und eines Anfangspunktes $\mathbf{z}$, der Jacobi-Matrix J und der Linearisierung

$$0 = \mathbf{f}(\mathbf{z}^*) = \mathbf{f}(\mathbf{z}+\boldsymbol{\Delta}\mathbf{z}) = \mathbf{f}(z) + \mathsf{J}(\mathbf{z})\boldsymbol{\Delta}\mathbf{z} \quad , \quad \mathsf{J}(\mathbf{z}) = \frac{\partial \mathbf{f}}{\partial \mathbf{z}}(\mathbf{z})$$

lösen und ergeben schließlich

$$\Delta\mathbf{z} = -\mathsf{J}^{-1}\mathbf{f}(\mathbf{z})$$

und

$$\mathbf{z}^* = \mathbf{z} + \boldsymbol{\Delta}\mathbf{z} \quad . \tag{A.3.16}$$

In (A.3.16) berechnet sich der Punkt $\mathbf{z}^*$, der in der nächsten Iteration verwendet wird, aus dem aktuellen Punkt $\mathbf{z}$ und der vollen Schrittweite $\boldsymbol{\Delta}\mathbf{z}$. Tatsächlich wird man jedoch zur Sicherung der Konvergenz einen Relaxationsparameter δ, $0 < \delta \leq 1$, verwenden, so dass sich der neue Punkt zu

$$\mathbf{z}^* = \mathbf{z} + \delta\boldsymbol{\Delta}\mathbf{z}$$

ergibt. Wendet man das Newton-Verfahren auf das nichtlineare Gleichungssystem (A.3.12)-(A.3.13), so ergibt sich das folgende System linearer Gleichungen

$$\begin{pmatrix} 0 & \mathsf{A} & \mathbb{1} \\ \mathsf{A}^{\mathrm{T}} & 0 & 0 \\ \mathsf{W} & 0 & \mathsf{X} \end{pmatrix} \begin{pmatrix} \boldsymbol{\Delta}\mathbf{x} \\ \boldsymbol{\Delta}\mathbf{y} \\ \boldsymbol{\Delta}\mathbf{w} \end{pmatrix} = \begin{pmatrix} \mathbf{c} - \mathsf{A}^{\mathrm{T}}\mathbf{y} - \mathbf{w} \\ \mathbf{b} - \mathsf{A}\mathbf{x} \\ \mu\mathbf{e} - \mathsf{X}\mathsf{W}\mathbf{e} \end{pmatrix} \quad ; \tag{A.3.17}$$

hierin bezeichnet $\mathbb{1}$ wieder die Einheitsmatrix passender Dimension. Die dritte Gleichung in (A.3.17) kann eliminiert werden und führt auf

$$\boldsymbol{\Delta}\mathbf{w} = \mathsf{X}^{-1}\left(\mu\mathbf{e} - \mathsf{X}\mathsf{W}\mathbf{e} - \mathsf{W}\boldsymbol{\Delta}\mathbf{x}\right) \tag{A.3.18}$$

und das reduzierte System

$$\begin{pmatrix} -\mathsf{X}^{-1}\mathsf{W} & \mathsf{A} \\ \mathsf{A}^{\mathrm{T}} & 0 \end{pmatrix} \begin{pmatrix} \boldsymbol{\Delta}\mathbf{x} \\ \boldsymbol{\Delta}\mathbf{y} \end{pmatrix} = \begin{pmatrix} \mathbf{c} - \mathsf{A}^{\mathrm{T}}\mathbf{y} - \mu\mathsf{X}^{-1}\left(\mu\mathbf{e} - \mathsf{X}\mathsf{W}\mathbf{e}\right) \\ \mathbf{b} - \mathsf{A}\mathbf{x} \end{pmatrix} \quad . \tag{A.3.19}$$

Weiter ist es möglich, $\boldsymbol{\Delta}\mathbf{x}$ zu eliminieren, woraus

$$\mathsf{A}^{\mathrm{T}}\mathsf{W}^{-1}\mathsf{X}\mathsf{A}\boldsymbol{\Delta}\mathbf{y} = \mathbf{b} - \mathsf{A}\mathbf{x} + \mathsf{A}^{\mathrm{T}}\mathsf{W}^{-1}\mathsf{X}\left[\mathbf{c} - \mathsf{A}^{\mathrm{T}}\mathbf{y} - \mu\mathsf{X}^{-1}\left(\mu\mathbf{e} - \mathsf{X}\mathsf{W}\mathbf{e}\right) - \mathsf{A}\boldsymbol{\Delta}\mathbf{y}\right] \tag{A.3.20}$$

und

$$\boldsymbol{\Delta}\mathbf{x} = -\mathsf{W}^{-1}\mathsf{X}\left[\mathbf{c} - \mathsf{A}^{\mathrm{T}}\mathbf{y} - \mu\mathsf{X}^{-1}\left(\mu\mathbf{e} - \mathsf{X}\mathsf{W}\mathbf{e}\right) - \mathsf{A}\boldsymbol{\Delta}\mathbf{y}\right] \tag{A.3.21}$$

folgen. Der Hauptrechenaufwand in den Iterationen von IPM besteht in der Berechnung der Lösung von (A.3.19) oder (A.3.20). Da die Matrix $\mathsf{A}^{\mathrm{T}}\mathsf{W}^{-1}\mathsf{X}\mathsf{A}$ in (A.3.20) positiv definit ist, könnte man versucht sein, das Cholesky-Verfahren einzusetzen. Dies hat allerdings den Nachteil, dass dabei in einem gewissen Maße die Struktur der Dünnbesetztheit zerstört würde. Bei der Lösung von (A.3.19) tritt dieser Nachteil zwar nicht auf, aber die Operationen der linearen Algebra sind weniger effizient. Dennoch ist dieser Zugang für die meisten Probleme das effizienteste Verfahren.

Sobald (A.3.20) gelöst ist, kann $\mathbf{\Delta x}$ aus (A.3.21) berechnet werden und schließlich folgt $\mathbf{\Delta w}$ aus (A.3.18). Die maximalen Schrittlängen werden so berechnet, dass die Nichtnegativität der Variablen gewährleistet ist. In praktischen Realisierungen werden die primalen und dualen Schrittlängen getrennt berechnet:

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \delta_P \mathbf{\Delta x} \quad , \quad \begin{pmatrix} \mathbf{y} \\ \mathbf{w} \end{pmatrix}^{(k+1)} = \begin{pmatrix} \mathbf{y} \\ \mathbf{w} \end{pmatrix}^{(k)} + \delta_D \begin{pmatrix} \mathbf{\Delta y} \\ \mathbf{\Delta w} \end{pmatrix} \quad .$$

Die Dämpfungsfaktoren δ_P und δ_D folgen dann aus den Heuristiken

$$\delta_P = \min\left\{1, (1-\varepsilon)\delta_P^{'}\right\} \quad , \quad \delta_P = \min\left\{1, (1-\varepsilon)\delta_D^{'}\right\}$$

mit

$$\delta_P^{'} := \min_i \left\{ \frac{x_i}{-\Delta x_i} \middle| \Delta x_i < 0 \right\} \quad , \quad \delta_D^{'} := \min_i \left\{ \frac{s_i}{-\Delta w_i} \middle| \Delta w_i < 0 \right\} \quad .$$

Diese Schrittlängen werden geringfügig um den Faktor $1-\varepsilon$ gekürzt, um zu verhindern, dass eine Variable den Rand trifft; hierbei ist ε eine kleine positive Zahl, z. B. $\varepsilon \approx 10^{-4}$.

A.3.2 Prädiktor-Korrektor-Schritt

In diesem Abschnitt bleibt noch der Prädiktor-Korrektor-Schritt zu erklären, der in den meisten IPM eingebaut ist. Prädiktor-Korrektor-Methoden gehören zu den Verfahren höherer Ordnung zur Lösung nichtlinearer Gleichungssyteme. Wie bereits bemerkt, stellt die Lösung von (A.3.19) bis (A.3.20) und die erforderliche Matrixinversion bzw. Faktorisierung der Matrix $\mathsf{A}^{\mathrm{T}}\mathsf{W}^{-1}\mathsf{X}\mathsf{A}$ den teuersten Rechenschritt der IPM dar. Nach der Faktorisierung der Matrix $\mathsf{A}^{\mathrm{T}}\mathsf{W}^{-1}\mathsf{X}\mathsf{A}$ können $\mathbf{\Delta y}$ und $\mathbf{\Delta x}$ berechnet werden, wenn die rechte Seite bekannt ist. Die Idee der Prädiktor-Korrektor-Methoden besteht darin, die Faktorisierung für verschiedene rechte Seiten wieder zu verwenden, um so bessere Suchrichtungen zu finden. Mehrotras [209] Prädiktor-Korrektor-Verfahren zweiter Ordnung löst die Newtongleichung im ersten Schritt für $\mu = 0$, woraus die *affine Skalierungs-(Prädiktor)* Suchrichtung Δ_α folgt. Führt man einen Schritt der Länge δ in diese Richtung, so kann die Komplementaritätslücke ausgenützt werden, um einen neuen Wert für den Homotopieparameter μ_e zu berechnen. Danach werden die Komponenten höherer Ordnung der Prädiktor-Korrektor-Richtung berechnet. Dabei wird gefordert, dass in der nächsten Iteration eine perfekte Zentrierung vorliegt, d. h.

$$(\mathsf{X} + \Delta\mathsf{X})(\mathbf{w} + \Delta\mathbf{w}) = \mu_e \mathbf{e}$$

bzw. die äquivalente Gleichung

$$\mathsf{W}\Delta\mathbf{x} + \mathsf{X}\Delta\mathbf{w} = -\mathsf{X}\mathbf{w} + \mu_e\mathbf{e} - \Delta\mathsf{X}\mathbf{\Delta w} \quad . \tag{A.3.22}$$

Statt nun den Term zweiter Ordnung auf der rechten Seite zu Null zu setzen, schlägt Mehrotra vor, $\Delta X \Delta \mathbf{w}$ aus der affinen-Skalierungsrichtung $\Delta X_\alpha \Delta \mathbf{w}_\alpha$ zu bestimmen. Daher ist es die Gleichung (A.3.22) mit einem angenommenen Barriereparameter μ_e, aus der die Prädiktor-Korrektor-Richtung bestimmt wird.

A.3.3 Zur Berechnung zulässiger Startpunkte

Gemäß ihrer Definition operieren IPM im Inneren des zulässigen Bereichs und benötigen daher strikt positive Startwerte für die Vektoren $\mathbf{x}$ und $\mathbf{w}$. Wie kann man derartige Startpunkte bestimmen? Dies ist in der Tat eine der schwierigen Aufgaben innerhalb dieses Forschungsgebietes. Die im Anhang A.1.2 beschriebene Zwei-Phasen-Methode, die beim Simplexverfahren angewendet wird, kann hier nämlich nicht helfen. Bei den Startheuristiken für IPM versucht man, wie z. B. in Andersen *et al.* (1996,[14]) beschrieben, einen Startpunkt so nahe wie möglich am zentralen Pfad und so nahe wie möglich an primaler und dualer Zulässigkeit zu generieren, d. h. man beginnt mit Startpunkten $\mathbf{x}^{(0)}$ und $\mathbf{w}^{(0)}$ richtigen Vorzeichens, verzichtet aber auf primale und duale Zulässigkeit. Problematisch bei derartigen Verfahren kann der Nachweis der primalen oder dualen Unzulässigkeit sein. Der Verzicht auf Zulässigkeit ist typisch für die Lösung nichtlinearer Probleme mit Hilfe von Newton-Verfahren. Die Zulässigkeit wird erst mit oder nahe der Optimalität erreicht[15]. Der Test auf primale und duale Zulässigkeit ist daher Bestandteil des Abbruchkriteriums [siehe Seite 335].

A.3.4 Die Berechnung des Homotopieparameters

Die Effizienz und Konvergenz von IPM hängt wesentlich von der Berechnung des Homotopieparameters μ ab. Da es sich bei den IPM um pfadfolgende Verfahren handelt, möchte man so nahe wie möglich am zentralen Pfad bleiben. Daher sind kleine Änderungen von μ empfohlen (*Verfahren mit kleiner Schrittweite*). Dies kann jedoch recht langsame Konvergenz zur Folge haben. Bei der gegenteiligen Heuristik, *Verfahren mit großer Schrittweite*, kann dagegen leicht der zentrale Pfad verlassen werden oder benötigt mehr Newtonschritte in jedem Iterationsschritt.

Die Beziehung (A.3.15), die auf dem zentralen Pfad gilt, lässt vermuten, dass der Barriereparameter mit der Komplementaritäts- oder der Dualitätslücke verknüpft sein sollte. Da auf die Forderung primaler und dualer Zulässigkeit verzichtet wurde, können beide Lücken voneinander verschieden sein.

Häufig wird $\mu^{(k)}$ nach der in Lustig *et al.* (1991,[198]) gegebenen Heuristik

$$\mu = \frac{\mathbf{b}^{\mathrm{T}}\mathbf{y} - \mathbf{c}^{\mathrm{T}}\mathbf{x}}{\varphi(n)}$$

berechnet, wobei n die Anzahl der Variablen und $\varphi(n)$ die Hilfsfunktion

$$\varphi(n) = \begin{cases} n & , n \leq 5000 \\ n^3, & n \geq 5000 \end{cases}$$

[15] Typischerweise erzeugt ein Innere-Punkte-Verfahren während der Iterationen die folgenden Informationen: die Anzahl der Iterationen, Größen, die die primale und duale Verletzung messen, primale und duale Zielfunktionswerte bzw. die Dualitätslücke sowie den Wert des Barriereparameters.

bezeichnet. In Mehrotras Prädiktor-Korrektor-Verfahren [209] wird der Barriereparameter gemäß

$$\mu = \left(\frac{g_a}{g}\right)^2 \frac{g_a}{n}$$

berechnet, wobei g_a die mit dieser Methode geschätzte Komplementaritätslücke bezeichnet. Ein anderer von Bock & Zillober (1995,[42]) verwendeter Ansatz, der besonders bei parallelen IPM effizient ist, beruht auf Extrapolationstechniken, die im Umfeld der Theorie von Differentialgleichungen verwendet werden.

A.3.5 Abbruchkriterium

Für einen gegeben Homotopieparameter μ bzw. μ^k iteriert das Verfahren solange, bis die relative primale Zulässigkeit

$$\Delta^P(\mathbf{x}) := \frac{\left\|\mathsf{A}\mathbf{x}^{(k)} - \mathbf{b}\right\|}{\max\left\{1, \left\|\mathsf{A}\mathbf{x}^{(0)} - \mathbf{b}\right\|\right\}} < \varepsilon^P$$

und die relative duale Zulässigkeit

$$\Delta^D(\mathbf{y}) := \frac{\left\|\mathsf{A}^{\mathrm{T}}\mathbf{y}^{(k)} + \mathbf{w}^{(k)} - \mathbf{c}\right\|}{\max\left\{1, \left\|\mathsf{A}^{\mathrm{T}}\mathbf{y}^{(0)} + \mathbf{w}^{(0)} - \mathbf{c}\right\|\right\}} < \varepsilon^D$$

mit hinreichender Genauigkeit erreicht wurden, und die Dualitätslücke $\Delta^R(\mathbf{x}, \mathbf{y})$

$$\Delta^R(\mathbf{x}, \mathbf{y}) := \frac{\mathbf{c}^{\mathrm{T}}\mathbf{x} - \mathbf{b}^{\mathrm{T}}\mathbf{y}}{1 + \mid \mathbf{b}^{\mathrm{T}}\mathbf{y} \mid} < \varepsilon^G$$

unterhalb des Wertes ε^G fällt. Typische Werte sind dabei $\varepsilon^P = \varepsilon^D = 10^{-10}$ und $\varepsilon^G = 10^{-8}$. Es wurde bereits in Abschnitt A.2.3 erwähnt, dass die Größe der Dualitätslücke ein Maß für die Entfernung von einem optimalen Punkt darstellt. Daher ist es nicht überraschend, dass $\Delta^R(\mathbf{x}, \mathbf{y})$ in diesem Abbruchkriterium auftritt.

A.3.6 Identifizierung einer Basis und Cross-Over

IPM generieren eine Approximation an die optimale Lösung, aber für LP-Probleme keine optimale Basis und keine Zerlegung der Variablen in Basis- und Nichtbasisvariablen. Da die berechnete Lösung strikt im Inneren des zulässigen Bereiches liegt, wird man wesentlich mehr Variablen finden, die nicht an ihren Schranken fixiert sind, als in Lösungen, die mit dem Simplexverfahren erzeugt wurden. Basislösungen sind jedoch sehr wesentlich in der Sensitivitätsanalyse und bei der Lösung der LP-Unterprobleme in B&B-Verfahren, da hier eine zur Verfügung stehende Basislösung einen Warmstart ermöglicht.

Daher ist es in vielen Situationen sinnvoll, eine optimale Innere-Punkt-Lösung mit Hilfe von Basisidentifizierungsverfahren [engl.: *basis identifications procedures*] in eine Basislösung zu konvertieren. Derartige Verfahren berechnen aus jeder zulässigen Lösung mit zugehörigem Zielfunktionswert eines LP-Problems eine Basislösung mit gleichem Zielfunktionswert; siehe z. B. Andersen & Ye (1994,[15]) sowie Andersen (1996,[12]).

Kommerzielle Implementierung von IPM verwenden ausgeklügelte *cross-over-Verfahren*, die gesteuert durch ein Abbruchkriterium von IPM auf das Simplexverfahren umschalten, wobei die mit Hilfe eines Basisidentifizierungsverfahren gewonnene Basis als gute, zulässige Startlösung verwendet wird. Das Simplexverfahren verbessert diesen Startwert schnell und berechnet eine optimale Basislösung.

A.3.7 Innere-Punkte-Methoden versus Simplexverfahren

Wie eine Reihe ausführlicher Testrechnungen z. B. Arbel (1994,[18]) zeigen, sind die besten Simplexverfahren und IPM in ihrer Güte vergleichbar. Es hängt vom einzelnen Problem ab, welches Verfahren effizienter ist. Dennoch lassen sich einige Charakteristika und Erfahrungswerte wie in Nemhauser (1994,[221]) zusammenstellen:

Das Simplexverfahren benötigt viele, aber numerisch nicht sehr aufwendige, etwa linear mit der Anzahl der Nebenbedingungen anwachsende Iterationsschritte.

Während für das Simplexverfahren eine weitestgehend vollständige Theorie und ein klares Verständnis der Komplexität dieses Verfahrens entwickelt wurden, sind Fragen nach dem Komplexitätsverhalten im schlechtesten Fall bei IPM noch eher schwierig zu beantworten. Eine theoretische Schranke für die Anzahl der Iterationen

$$\mathcal{O}\left(\sqrt{n}\ln\frac{1}{\varepsilon}\right)$$

ist noch nicht in guter Übereinstimmung mit dem in der Praxis beobachteten Verhalten

$$\mathcal{O}\left(\ln n\right) \quad \text{oder} \quad \mathcal{O}\left(n^4\right) \quad . \tag{A.3.23}$$

IPM benötigen meist etwa 20 bis 50 Iterationen; diese Zahl wächst nur leicht mit der Problemgröße gemäß (A.3.23). Jede Iteration benötigt die Lösung eines $n \times n$ Systems nichtlinearer Gleichungen; dieser Schritt kostet sehr viel Rechenzeit. Die Linearisierung dieses nichtlinearen Gleichungssystems führt auf ein lineares Gleichungssystem, dessen Lösung die Invertierung einer $n \times n$-Matrix erfordert. Für IPM ist es daher sehr wichtig, dass die Matrix dünnbesetzt ist. Gerade für große LP-Probleme, die auf dünnbesetzte Matrizen führen, scheinen IPM Vorteile gegenüber dem Simplexverfahren zu haben.

Für große Systeme zeigen Hybridverfahren, die in einer ersten Phase mit Hilfe von IPM eine nahezu optimale Lösung bestimmen, die dann in der zweiten Phase als Startlösung für das Simplexverfahren verwendet und schließlich zu optimalen Basis verbessert wird, gute Laufzeiteigenschaften.

Im Zusammenhang mit B&B-Verfahren der gemischt-ganzzahligen Optimierung haben IPM den Nachteil, dass Warmstarts kaum effizient möglich sind.

A.4 Gemischt-ganzzahlige nichtlineare Optimierung

A.4.1 Ein Äußere-Approximation-Verfahren für konvexe Probleme

Verfahren, die auf Äußere-Approximation aufbauen, beschreiben den zulässigen Bereich eines beschränkten Optimierungsproblems als Schnitt eines Ensembles von einfach strukturierteren Mengen, z. B. Polyeder. Ein solches Verfahren soll hier auf spezielle Optimierungsprobleme angewendet werden, in denen sämtliche auftretenden Funktionen

stetig-differenzierbar und konvex sind und ganzzahlige Variablen nur in linearen Termen auftreten, d. h.

$$\mathbf{P} : \min_{\mathbf{x},\mathbf{y}} \left\{ \mathbf{c}^{\mathrm{T}}\mathbf{y} + f(\mathbf{x}) \,|\mathbf{g}(\mathbf{x}) + B\mathbf{y} \leq 0 \right\} \tag{A.4.24}$$

mit $\mathbf{x} \in X \subseteq \mathbb{R}^n$, $\mathbf{y} \in U \subseteq \mathbb{R}^m_+$ und Polyeder $X := \{\mathbf{x} \in \mathbb{R}^n \mid A_1\mathbf{x} \leq \mathbf{a}_1\}$, $U := \{\mathbf{y} \in Y \mid A_2\mathbf{y} \leq \mathbf{a}_2\ ,\ \ \mathbf{y} \text{ diskret}\}$ und passend gewählten Matrizen A_1, A_2, $\mathbf{a}_1$ und $\mathbf{a}_2$. In den meisten Fällen wird der Fall $Y = \{0,1\}^m$ vorliegen, d. h. die ganzzahligen Variablen sind in einem m-dimensionalen Hyperwürfel enthalten. Zu beachten ist, dass zwar die Funktionen $f(\mathbf{x})$ und $\mathbf{g}(\mathbf{x})$ konvex sind, nicht jedoch das Problem $\mathbf{P}$ simultan in den Variablen $\mathbf{x}$ und $\mathbf{y}$.

Gegebenenfalls können ganzzahlige Variablen, die in nichtlinearen Termen auftreten, durch zusätzliche Gleichungen und kontinuierliche Variablen ersetzt werden. Gleichungsbedingungen, die z. B. in der Form $A\mathbf{y} + \mathbf{h}(\mathbf{x}) = 0$ vorliegen, können, wie in Abschnitt A.4.9.1 beschrieben, durch Ungleichungen relaxiert werden.

Mit den Hilfsgrößen $f^L := \min\{f(\mathbf{x}), \mathbf{x} \in X\}$ und $f^U := \max\{f(\mathbf{x}), \mathbf{x} \in X\}$ kann (A.4.24) in eine speziellere Form geschrieben werden, in der die Zielfunktion linear ist:

$$\mathbf{P}_0 : \quad \min_{\mathbf{x}\in X, \mathbf{y}\in U} \left\{ \mathbf{c}^{\mathrm{T}}\mathbf{y} + \mu \ \mid\ f(\mathbf{x}) - \mu \leq 0\ ,\ \mathbf{g}(\mathbf{x}) + B\mathbf{y} \leq 0\ ,\ \mu \in \left[f^L, f^U\right] \right\} \quad . \tag{A.4.25}$$

Der zulässige Bereich des Problems $\mathbf{P}_0$ sei mit $Feas(\mathbf{P}_0)$ bezeichnet. Ferner sei für die weitere Betrachtungen angenommen, dass die Nebenbedingungen bestimmten *Regularitätsbedingungen* genügen. Ein Beispiel dafür sind die *Slaterbedingungen*

$$\exists \mathbf{x} \in X \mid \mathbf{g}(\mathbf{x}) + B\mathbf{y} < 0 \quad \forall \mathbf{y} \in U \cap V$$

mit

$$V := \{\mathbf{y} \mid \mathbf{g}(\mathbf{x}) + B\mathbf{y} \leq 0 \quad \text{für mindestens ein } \mathbf{x} \in X\} \quad . \tag{A.4.26}$$

Die Menge V ist die Menge derjenigen $\mathbf{y}$-Variablen, die einen nichtleeren kontinuierlichen Bereich erzeugen. Desweiteren sei der für festes $\mathbf{y}$ resultierende zulässige Bereich des kontinuierlichen Problems

$$F(\mathbf{y}) := \left\{ \mathbf{x} \in X\ ,\ \ \mu \in \left[f^L, f^U\right] \ \mid\ f(\mathbf{x}) - \mu \leq 0\ ,\ \mathbf{g}(\mathbf{x}) + B\mathbf{y} \leq 0 \right\}\ ,\ \mathbf{y} \in U \cap V \tag{A.4.27}$$

definiert, d. h. $F(\mathbf{y})$ ist die Menge der zulässigen Punkte des Problems $\mathbf{P}_0$ (A.4.25) bei festgehaltenem $\mathbf{y} \in U \cap V$. Wegen $\mathbf{y} \in U \cap V$ und der Definition von V ist $F(\mathbf{y})$ eine nichtleere Menge. Für alle $\mathbf{y} \in U \cap V$ ist $F(\mathbf{y})$ eine abgeschlossene konvexe Menge, da f und $\mathbf{g}$ konvexe Funktionen sind und $\mathbf{X}$ als konvexes Polyeder kompakt ist.

A.4.2 Natürliche polyedrische Repräsentation

Die natürliche polyedrische Darstellung von $F(\mathbf{y})$ ist der Schnitt eines Ensembles von Halbebenen. Wegen der angenommenen Linearität des Problems in den ganzzahligen Variablen führen verschiedene $\mathbf{y}$ lediglich zu verschiedenen Positionen des zulässigen Bereichs $F(\mathbf{y})$ innerhalb der Menge X. Damit folgt durch den Schnitt über alle $\mathbf{y} \in U \cap V$ die polyedrische Darstellung von $Feas(\mathbf{P}_0)$:

$$\begin{aligned} 0 &\geq f(\mathbf{x}) - \mu \geq f(\mathbf{x}^i) + \boldsymbol{\nabla} f(\mathbf{x}^i)^{\mathrm{T}}(\mathbf{x} - \mathbf{x}^i) - \mu \\ 0 &\geq \mathbf{g}(\mathbf{x}) + B\mathbf{y} \geq \mathbf{g}(\mathbf{x}^i) + \boldsymbol{\nabla}\mathbf{g}(\mathbf{x}^i)^{\mathrm{T}}(\mathbf{x} - \mathbf{x}^i) + B\mathbf{y} \quad , \quad \forall \mathbf{x}^i \in X \\ \mathbf{x} &\in X\ ,\ \ \mu \in [f^L, f^U]\ ,\ \mathbf{y} \in U \quad . \end{aligned} \tag{A.4.28}$$

Die so definierten unendlich vielen, durch $\mathbf{x}^i$ induzierten Halbräume liefern eine Approximation von f und $\mathbf{g}$ durch das punktweise Maximum aus der simultanen Betrachtung aller Linearisierungen (hier wird die Annahme der Konvexität benötigt). Der Halbraum $H_{ij} = 0$ ist durch

$$\{\mathbf{x} \mid \mathbf{g}_j(\mathbf{x}^i) + \boldsymbol{\nabla}\mathbf{g}_j(\mathbf{x}^i)^{\mathrm{T}}(\mathbf{x} - \mathbf{x}^i) + B\mathbf{y} = 0\}$$

charakterisiert. Betrachtet man als Beispiel einen zweidimensionalen zulässigen Bereich, der von einer Kurve $\mathbf{g}_1$ begrenzt wird, und liegt ein Punkt $\mathbf{x}_1$ auf der Kurve $\mathbf{g}_1$, so ist H_{11} gerade die Tangente an die Kurve; im allgemeineren Fall also die Tangential-Hyperebene.

A.4.3 Äußere Approximation und MILP-Masterprobleme

Verfahren, die auf Äußere-Approximation aufbauen, beschreiben den zulässigen Bereich eines beschränkten Optimierungsproblems als Schnitt eines Ensembles von einfach strukturierteren Mengen, z. B. Polyeder. Das ursprüngliche Problem $\mathbf{P}_0$ lässt sich unter Verwendung der oben diskutierten polyedrischen Darstellung $Feas(\mathbf{P}_0)$ nun in folgender Weise durch das „einfachere" Problem $\mathbf{P}_1$

$$\begin{aligned} & \mathbf{P}_1 : \min \mathbf{c}^{\mathrm{T}}\mathbf{y} + \mu & & & & \\ u.d.N. \quad & f(\mathbf{x}^i) + \boldsymbol{\nabla} f(\mathbf{x}^i)^{\mathrm{T}}(\mathbf{x} - \mathbf{x}^i) - \mu & \leq & 0 & , & \forall \mathbf{x}^i \in X \\ & \mathbf{g}(\mathbf{x}^i) + \boldsymbol{\nabla}\mathbf{g}(\mathbf{x}^i)^{\mathrm{T}}(\mathbf{x} - \mathbf{x}^i) + B\mathbf{y} & \leq & 0 & , & \forall \mathbf{x}^i \in X \\ & \mathbf{x} & \in & X \ , \ \mathbf{y} \in U \ , \ \mu \in \left[f^L, f^U\right] & & \end{aligned} \tag{A.4.29}$$

ersetzen. Genauso kann gezeigt werden, dass die in (A.4.26) definierte Menge V durch

$$V := \{\mathbf{y} \mid \ \mathbf{g}(\mathbf{x}^i) + \boldsymbol{\nabla}\mathbf{g}(\mathbf{x}^i)^{\mathrm{T}}(\mathbf{x} - \mathbf{x}^i) + B\mathbf{y} \leq 0 \quad , \quad \forall \mathbf{x}^i \in X \text{ für mindestens ein } \mathbf{x} \in X \tag{A.4.30}$$

dargestellt werden kann. In beiden Fällen ist das ursprüngliche Problem in ein MILP-Problem zurückgeführt worden, allerdings auf eines mit unendlich vielen Nebenbedingungen.

Theorem 7 $\mathbf{P}_0$ *und* $\mathbf{P}_1$ *sind äquivalent.*

Theorem 8 $U \cap V$ *ist diskret und endlich.*

A.4.4 Formulierung des Masterproblems

Zwar ist das ursprüngliche Problem $\mathbf{P}_0$ in ein gemischt-ganzzahliges lineares Problem überführt worden, aber die unendlich vielen Nebenbedingungen bereiten natürlich erhebliche Schwierigkeiten. Es lässt sich nun unter Ausnutzung der Eigenschaft, dass $U \cap V$ diskret und endlich ist und vermöge einer Projektion (Projektion von Penthargy) des ursprünglichen Problems $\mathbf{P}$ auf einen diskreten Raum nur *endlich* viele Punkte $\mathbf{x}^i \in X$ für die äußere Approximation benötigt werden. Die Projektion von $\mathbf{P}$ auf $\mathbf{y}$ ist durch

$$z : \min_{\mathbf{y}^i \in U \cap V} \left\{ \inf_{\mathbf{x} \in X} \left\{ \mathbf{c}^{\mathrm{T}}\mathbf{y}^i + f(\mathbf{x}) \mid \mathbf{g}(\mathbf{x}) + B\mathbf{y}^i \right\} \right\} =: \min_{\mathbf{y}^i \in U \cap V} z(\mathbf{y}^i) \tag{A.4.31}$$

gegeben. Für gegebenes $\mathbf{y}^i \in U \cap V$ entspricht das Infimum des inneren Problems dem optimalen Wert des Problems $\mathbf{P}$ für festes $\mathbf{y}^i$. Weiter kann gezeigt werden, dass für jedes $\mathbf{y}^i \in U \cap V$ das Infimum auch angenommen und dem optimalen Wert $z(\mathbf{y}^i)$ des nichtlinearen kontinuierlichen Minimierungsproblems $\mathbf{S}(\mathbf{y}^i)$

$$\begin{aligned} & & \mathbf{S}(\mathbf{y}^i) \quad &: \quad z(\mathbf{y}^i) = \mathbf{c}^{\mathrm{T}}\mathbf{y}^i + \min f(\mathbf{x}) \\ u.d.N. \quad & \mathbf{g}(\mathbf{x}) + B\mathbf{y}^i \quad &\leq \quad 0 \quad , \quad \mathbf{x} \in X \end{aligned} \tag{A.4.32}$$

entspricht. Hierbei handelt es sich wegen der Konvexität von $f(\mathbf{x})$ und $\mathbf{g}(\mathbf{x})$ um ein konvexes Problem in $\mathbf{x}$. Dieser Punkt ist wichtig, da zwar Verfahren zur Bestimmung des globalen Minimums bei konvexen Problemen existieren, nicht aber zur Lösungen von nichtkonvexen Problemen.

Sei $\mathbf{y}^i$ ein zulässiger Punkt für P, sodass $S(\mathbf{y}^i)$ zulässig ist, d. h. $\mathbf{y}^i \in U \cap V$, dann sei mit $\mathbf{x}^i$ der minimierende Punkt des Teilproblems $S(\mathbf{y}^i)$ bezeichnet.

Die MILP-Lösung von $\mathbf{P}_1$ hat als y-Teil $\mathbf{y}^i$ eine Ecke der konvexen Hülle aller zulässigen ganzzahligen Lösungen $Conv(U \cap V)$ und als kontinuierlichen Teil einen Randpunkt $[\mathbf{x} = \mathbf{x}^i, \mu = f(\mathbf{x}^i)]$ von (A.4.32) für den linearen Träger an $f(\mathbf{x})$, der zu $\mathbf{y} = \mathbf{y}^i$ gehört. Dieses $\mathbf{x}^i$ wurde als Optimallösung von $S(\mathbf{y}^i)$ bestimmt (Projektion von $\mathbf{P}$ auf $\mathbf{x}$-Raum).

Es gibt nur endlich viele $\mathbf{y}^i \in U \cap V$ und nur endlich viele Optimallösungen $\mathbf{x}^i$ von Problemen $S(\mathbf{y}^i)$. Wählt man dasjenige Paar $[\mathbf{y}^i, \mathbf{x}^i(\mathbf{y}^i)]$ mit minimalem $z(\mathbf{y}^i)$ aus, so ist $[\mathbf{x}^*, \mathbf{y}^*]$ mit

$$\mathbf{x}^* = \mathbf{x}^i \quad , \quad \mathbf{y}^* = \mathbf{y}^i$$

Lösung von $\mathbf{P}$. In der Praxis durchführbar ist dieser Weg allerdings nicht, da es in der Regel eine exponentiell anwachsende Anzahl von Punkten $\mathbf{y}^i \in U \cap V$ und damit zu lösende NLP-Probleme gibt. Daher wird ein anderer Weg verfolgt:

Wegen der durchgeführten Projektion und deren Eigenschaften sind es genau die Punkte $\mathbf{x}^i$, die in den Nebenbedingungen (A.4.29) ausreichen, um die äußere Approximation zu erzeugen. Damit ergibt sich das folgende *Masterproblem*:

$$\begin{aligned} & & \mathbf{M} \quad &: \quad \min \mathbf{c}^{\mathrm{T}}\mathbf{y} + \mu \\ u.d.N. \quad & f(\mathbf{x}^i) + \boldsymbol{\nabla} f(\mathbf{x}^i)^{\mathrm{T}}(\mathbf{x} - \mathbf{x}^i) - \mu \quad &\leq \quad 0 \quad , \quad \forall i \in T \\ & \mathbf{g}(\mathbf{x}^i) + \boldsymbol{\nabla}\mathbf{g}(\mathbf{x}^i)^{\mathrm{T}}(\mathbf{x} - \mathbf{x}^i) + B\mathbf{y} \quad &\leq \quad 0 \quad , \quad \forall i \in T \\ & & \mathbf{x} \quad &\in \quad X \ , \ \mathbf{y} \in U \ , \ \mu \in \left[f^L, f^U\right] \end{aligned} \tag{A.4.33}$$

mit

$$T := \left\{ i \mid \mathbf{x}^i \text{ ist Optimallösung von } S(\mathbf{y}^i) \ , \ \mathbf{y}^i \in U \cap V \right\} \quad .$$

Das Masterproblem sieht dem Problem $\mathbf{P}$ sehr ähnlich, insbesondere handelt es sich wieder um ein MILP-Problem, aber es besitzt nur endlich viele Nebenbedingungen. Da der zulässige Bereich des ursprünglichen Problems $\mathbf{P}$ als nichtleer und kompakt angenommen wurde, existiert eine optimale Lösung sowohl für das Problem $\mathbf{P}$ als auch für $\mathbf{M}$. Das folgende Theorem stellt die Äquivalenz zwischen dem ursprünglichen Problem $\mathbf{P}$ und dem Masterproblem $\mathbf{M}$ her.

Theorem 9 $(\mathbf{x}^*, \mathbf{y}^*)$ *ist optimal für* $\mathbf{P}$ *genau dann, wenn* $(\mathbf{x}^*, \mathbf{y}^*)$ *optimal ist für das obige Masterproblem* $\mathbf{M}$ *mit* $\mu = f(\mathbf{x}^*)$.

Nun steht man aber auch wieder bzw. noch vor dem Problem, dass für jedes $\mathbf{y}^i \in U \cap V$ das zugehörige Problem $S(\mathbf{y}^i)$ zu lösen ist. Um dies zu vermeiden, wird die folgende Strategie, diese ist die eigentliche Idee des Verfahrens, verfolgt, die geeignete Relaxierungen $\mathbf{M}^k$ von $\mathbf{M}$ betrachtet und folgende Optimierungsprobleme $\mathbf{M}^k$

$$\mathbf{M}^k: \quad z^k = \min \mathbf{c}^{\mathrm{T}}\mathbf{y} + \mu \quad , \quad (\mathbf{x}, \mathbf{y}) \in \Omega^k \ , \ \mathbf{x} \in X \ , \ \mathbf{y} \in U \ , \ \mu \in \left[f^L, f^U\right]$$

mit

$$\Omega^k := \left\{ \begin{array}{rl} (\mathbf{x}, \mathbf{y}) \mid f(\mathbf{x}^i) + \boldsymbol{\nabla} f(\mathbf{x}^i)^{\mathrm{T}}(\mathbf{x} - \mathbf{x}^i) - \mu \leq 0 & , \quad \forall i \in T^k \\ \mathbf{g}(\mathbf{x}^i) + \boldsymbol{\nabla}\mathbf{g}(\mathbf{x}^i)^{\mathrm{T}}(\mathbf{x} - \mathbf{x}^i) + B\mathbf{y} \leq 0 & , \quad \forall i \in T^k \end{array} \right\} \tag{A.4.34}$$

und

$$T^k := \left\{ i \mid \mathbf{x}^i \text{ ist Optimallösung von } S(\mathbf{y}^i) \ , \ \mathbf{y}^i \in U \cap V \ , \ i = 1, 2, \ldots, k \right\} \tag{A.4.35}$$

zu lösen sucht. Hierzu ist Folgendes zu bemerken:

1. In Iteration k wird das relaxierte Problem $\mathbf{M}^k$ gelöst und fast alle Nebenbedingungen in $\mathbf{M}$ werden ignoriert, nämlich all diejenigen, die durch $i \in T \backslash T^k$ erzeugt würden.
2. Falls die Lösung $(\mathbf{x}, \mathbf{y}^{k+1})$ von M^k nicht bestimmte Abbruchkriterien erfüllt, wird das Unterproblem $S(\mathbf{y}^{k+1})$ gelöst, um $\mathbf{x}^{k+1}$ für weitere äußere Approximationen zu bestimmen.
3. Es wird das neue relaxierte Masterproblem M^{k+1} durch die Halbräume bezüglich
$$\mathbf{x}^{k+1} : T^{k+1} := T^k \cup \{k+1\} \to \Omega^{k+1} \quad ,$$
konstruiert, um Ω^{k+1} zu erzeugen.

A.4.5 Schranken

Sei G der zulässige Bereich des Problems $\mathbf{P}_0$

$$G := Feas(\mathbf{P}_0) = \{(\mathbf{x}, \mathbf{y}) \mid f(\mathbf{x}) - \mu \leq 0 \ , \ \mathbf{g}(\mathbf{x}) + B\mathbf{y} \leq 0 \ , \ \mathbf{x} \in X \ , \ \mathbf{y} \in U\}$$

und Γ^k der zulässige Bereich des Masterproblems in der k-ten Iteration, d. h.

$$\Gamma^k := Feas(M^k) = (X \times U) \cap \Omega^k \quad .$$

Es lässt sich zeigen, dass $\Gamma^k \supseteq G$ für $k \in \mathbb{N}$. Damit bietet es sich nun an, das Problem $\mathbf{P}_0$ durch eine Folge von Problemen $\mathbf{M}^k$

$$\mathbf{M}^k: \quad \min\left\{\mathbf{c}^{\mathrm{T}}\mathbf{y} + \mu \mid (\mathbf{x}, \mathbf{y}) \in \Gamma^k \ , \ \mu \in \left[f^L, f^U\right]\right\} \ , \ k = 1, 2, \ldots \tag{A.4.36}$$

zu lösen, wobei die folgende Inklusion $G \subseteq \Gamma^k \subseteq \Gamma^{k-1} \subseteq \ldots \subseteq \Gamma^1$ gilt. Dann gilt als Konsequenz der Relaxierung

$$\begin{array}{rl} & \min\left\{\mathbf{c}^{\mathrm{T}}\mathbf{y} + \mu \mid (\mathbf{x}, \mathbf{y}) \in G \ , \ \mu \in \left[f^L, f^U\right]\right\} \\ \geq & z^k := \min\left\{\mathbf{c}^{\mathrm{T}}\mathbf{y} + \mu \mid (\mathbf{x}, \mathbf{y}) \in \Gamma^k \ , \ \mu \in \left[f^L, f^U\right]\right\} \\ & \vdots \\ \geq & z^1 := \min\left\{\mathbf{c}^{\mathrm{T}}\mathbf{y} + \mu \mid (\mathbf{x}, \mathbf{y}) \in \Gamma^1 \ , \ \mu \in \left[f^L, f^U\right]\right\} \quad . \end{array}$$

Da das Teilproblem $S(\mathbf{y}^i)$ einen „kleineren" zulässigen Bereich als $\mathbf{P}$ hat, gilt auch

$$\begin{aligned} z &= \min\left\{\mathbf{c}^{\mathrm{T}}\mathbf{y} + f(\mathbf{x}) \mid (\mathbf{x},\mathbf{y}) \in G\right\} \\ &\leq z(\mathbf{y}^i) = \min\left\{\mathbf{c}^{\mathrm{T}}\mathbf{y}^i + f(\mathbf{x}) \mid \mathbf{x} \in X\ ,\ \mathbf{g}(\mathbf{x}) + B\mathbf{y}^i \leq 0\right\} \quad . \end{aligned}$$

Dies erlaubt die folgende Interpretation:

1. Die optimalen Zielfunktionswerte z^k der relaxierten Masterprobleme bilden eine monoton ansteigende Folge von unteren Schranken für das ursprüngliche MINLP-Problem $\mathbf{P}$.
2. Für jedes $\mathbf{y}^i \in U \cap V$ ist $z(\mathbf{y}^i)$ eine obere Schranke. Die Ungleichungskette $z^k - 1 \leq \mathbf{c}^{\mathrm{T}}\mathbf{y} + \mu \leq z^*$, wobei z^* die gegenwärtig beste obere Schranke ist, ersetzt die ursprüngliche Nebenbedingungen $f^L \leq \mu \leq f^U$.

A.4.6 Unzulässige Teilprobleme

Damit $\mathbf{y} \in U$ brauchbar ist, muss $S(\mathbf{y})$ zulässig sein, d. h. es muss $\mathbf{y} \in U \cap V$ gelten, wobei gemäß (A.4.30) V die Darstellung

$$V := \{\mathbf{y} \mid\ \mathbf{g}(\mathbf{x}^i) + \boldsymbol{\nabla}\mathbf{g}(\mathbf{x}^i)^{\mathrm{T}}(\mathbf{x} - \mathbf{x}^i) + B\mathbf{y} \leq 0 \quad \forall \mathbf{x}^i \in X \text{ für mindestens ein } \mathbf{x} \in X\}$$

hat. Die Iterationen produzieren sukzessive Teilmengen V^k der Nebenbedingungen, die V beschreiben. Die Mengen V^k sind durch

$$V^k := \{\mathbf{y} \mid\ \mathbf{g}(\mathbf{x}^i) + \boldsymbol{\nabla}\mathbf{g}(\mathbf{x}^i)^{\mathrm{T}}(\mathbf{x} - \mathbf{x}^i) + B\mathbf{y} \leq 0 \quad \forall i \in T^k \text{ für mindestens ein } \mathbf{x} \in X\}$$

gegeben und genügen den offensichtlichen Inklusionen

$$V \subseteq V^k \subseteq V^{k-1} \subseteq \ldots \subseteq V^1 \quad .$$

Daraus folgt nun allerdings, dass ein $\mathbf{y} \in V^k$ als Lösung des Masterproblems M^k nicht unbedingt der Bedingung $\mathbf{y} \in V$ genügt, d. h. es kann vorkommen, dass $\mathbf{y} \in V^k \backslash V$, was bedeutet, dass $S(\mathbf{y})$ keine zulässigen Punkte besitzt. Ein solches $\mathbf{y}$ sollte in weiteren Iterationen nicht wieder vorkommen. In der Tat kann ein solches $\mathbf{y} \in V^k \backslash V$ eliminiert werden, indem ein geeigneter Halbraum zu V^k addiert wird, der dann zu einer restriktiveren Menge V^{k+1} führt. Eine natürliche Möglichkeit zu testen, ob für $\mathbf{y}^i \in V^k$ auch $\mathbf{y}^i \in V$ gilt, ist es, das Problem $S(\mathbf{y}^i)$ zu lösen. Führt $\mathbf{y}^i \in V^k \backslash V$ zu einem unzulässigen Problem $S(\mathbf{y}^i)$, so liefert das NLP-Lösungsverfahren (Phase I) ein $\mathbf{x}^i \in X$ mit minimaler Verletzung der Nebenbedingungen von $S(\mathbf{y}^i)$. Damit lässt sich V^{k+1} aus V^k durch

$$V^{k+1} := V^k \cap \left\{\mathbf{x} \mid f(\mathbf{x}^i) + \boldsymbol{\nabla} f(\mathbf{x}^i)^{\mathrm{T}}(\mathbf{x} - \mathbf{x}^i) - \mu\ ,\ \mathbf{g}(\mathbf{x}^i) + \boldsymbol{\nabla}\mathbf{g}(\mathbf{x}^i)^{\mathrm{T}}(\mathbf{x} - \mathbf{x}^i) + B\mathbf{y}^i \leq 0\right\} \tag{A.4.37}$$

definieren. Die neuen Nebenbedingungen schneiden $(\mathbf{y}^i, \mathbf{x}^i)$, aber auch andere unzulässige Bereiche ab. Außerdem ist durch (A.4.37) nicht garantiert, dass $\mathbf{y}^i \notin V$ nicht noch einmal gewählt wird, d. h. der Fall

$$0 < \mathbf{g}(\mathbf{x}^i) + B\mathbf{y}^i \geq \mathbf{g}(\mathbf{x}^i) + \boldsymbol{\nabla}\mathbf{g}(\mathbf{x}^i)^{\mathrm{T}}(\mathbf{x} - \mathbf{x}^i) + B\mathbf{y} \leq 0$$

ist möglich. Die einzig sinnvolle Abhilfe besteht darin, einen Integer-Cut der Form (A.4.38) hinzuzufügen, der $\mathbf{y}^i$ aus U abschneidet. Für den Fall, dass $\mathbf{y}$ eine Kombination von Binärvariablen repräsentiert, bietet sich ein Binär-Cut der Form (5.9.5) an.

A.4.7 Zusammenfassung des Äußere-Approximation-Verfahrens

Seien alle bisher gemachten Annahmen erfüllt, insbesondere habe **P** ein endliches Optimum, d. h. der zulässige Bereich ist beschränkt. Definiere für $\mathbf{x}^i \in X$ den Integer-Cut

$$C(\mathbf{x}^i) := \left\{ \begin{array}{c} (\mathbf{x},\mathbf{y}) \mid f(\mathbf{x}^i) + \nabla f(\mathbf{x}^i)^{\mathrm{T}}(\mathbf{x}-\mathbf{x}^i) - \mu \leq 0 \\ \mathbf{g}(\mathbf{x}^i) + \nabla \mathbf{g}(\mathbf{x}^i)^{\mathrm{T}}(\mathbf{x}-\mathbf{x}^i) + B\mathbf{y} \leq 0 \quad , \quad \mu \in \mathbb{R} \end{array} \right\} \quad . \tag{A.4.38}$$

1. Sei $\Omega^0 := \mathbb{R}^n \times \mathbb{R}^m$, $z^0 = -\infty$ die Initialisierung der monoton wachsenden Folge z^i der unteren Schranken, $z^* = +\infty$ die Initialisierung der monoton wachsenden Folge oberer Schranken und $i := 1$ die Initialisierung des Iterationszählers. Wähle $\mathbf{y}^1 \in U$ oder, wenn möglich, $\mathbf{y}^1 \in U \cap V$.

2. Löse das durch $\mathbf{y}^i$ parametrisierte nichtlineare Teilproblem $S(\mathbf{y}^i)$, d. h.

$$\mathbf{S}(\mathbf{y}^i) : \mathbf{c}^{\mathrm{T}}\mathbf{y}^i + \min f(\mathbf{x}) \quad , \quad u.d.N. \quad \mathbf{g}(\mathbf{x}) + B\mathbf{y}^i \leq 0 \quad , \quad \mathbf{x} \in X \quad .$$

   Nun können natürlich zwei Fälle auftreten: (a) Falls $S(\mathbf{y}^i)$ eine endliche Optimallösung hat, nenne diese $[\mathbf{x}^i, z(\mathbf{y}^i)]$, wobei $z(\mathbf{y}^i)$ nun eine neue zulässige obere Schranke des optimalen Wertes des Problems **P** ist, und führe das folgende Update durch: Falls $z(\mathbf{y}^i) \leq z^*$, so setze:

$$z^* := z(\mathbf{y}^i) \quad , \quad \mathbf{y}^* := \mathbf{y}^i \quad , \quad \mathbf{x}^* := \mathbf{x}^i \quad , \quad \Omega^i := \Omega^{i-1} \cap C(\mathbf{x}^i)$$

   und gehe zu Schritt 3. (b) Falls $S(\mathbf{y}^i)$ unzulässig ist, d. h. $\mathbf{y}^i \notin V$, so ist das aus $S(\mathbf{y}^i)$ resultierende $\mathbf{x}^i$ die „Lösung mit kleinster Verletzung der Nebenbedingungen von $S(\mathbf{y}^i)$". Wie weiter unten beschrieben, wird daraus ein Integer-Cut berechnet und zu M^i hinzugefügt, der $\mathbf{y}^i$ abschneidet. Setze $\Omega^i := \Omega^{i-1} \cap C(\mathbf{x}^i)$ und gehe zu Schritt 3.

3. Löse das Masterproblem M^i, d. h.

$$\begin{array}{rcl} \mathbf{M}^i & : & \min \mathbf{c}^{\mathrm{T}}\mathbf{y}^i + \mu \\ u.d.N. \quad (\mathbf{x},\mathbf{y}) & \in & \Omega^i \quad , \quad z^{i-1} \leq \mathbf{c}^{\mathrm{T}}\mathbf{y} + \mu \leq z^* \quad , \quad \begin{array}{l} \mathbf{x} \in X \\ \mathbf{y} \in U \end{array} \quad , \quad \mu \in \mathbb{R} \\ \mathbf{y} & \in & U \cap \{\text{Menge der Integer-Cuts}\} \quad . \end{array}$$

   Hier können die beiden folgenden Fälle auftreten: (a) Falls $\mathbf{M}^i$ keinen MILP-zulässigen Punkt hat, wird das Verfahren terminiert. $(\mathbf{x}^*, \mathbf{y}^*)$ mit Zielfunktionswert z^* ist Lösung des MINLP-Problems **P**. Dies entspricht der optimalen Lösung des durch $\mathbf{y}^*$ parametrisierten, in Schritt 2a) definierten NLP. (b) Liefert dagegen $\mathbf{M}^i$ eine Lösung $(\mathbf{x}, \mathbf{y}, \mu)$ mit z^i, so ist z^i ein weiteres Folgenglied in der monoton wachsenden Folge unterer Schranken des optimalen Wertes des MINLP-Problems **P** und $\mathbf{y}$ eine weitere Kombination ganzzahliger Variablen, die im weiteren Verlauf des Verfahrens getestet werden muss. Daher wird

$$\mathbf{y}^i = \mathbf{y} \quad , \quad i := i + 1$$

   gesetzt und mit Schritt 2 fortgesetzt.

Bemerkung: Das Verfahren löst also abwechselnd kontinuierliche nichtlineare Teilprobleme (NLP-Probleme) $S(\mathbf{y}^i)$ und MILP-Masterprobleme $\mathbf{M}^i$. Wären f und $\mathbf{g}(\mathbf{x})$ lineare Funktionen, so würde das relaxierte Masterproblem $\mathbf{M}^1$ in der ersten Iteration mit dem ursprünglichen Problem identisch sein und das Verfahren in maximal zwei Iterationen terminieren. Im nichtlinearen Fall erhält man in einer endlichen Anzahl von Schritten die optimale Lösung von $\mathbf{P}$. Der Abbruch des Verfahrens kann durch mindestens zwei Kriterien gesteuert werden. Durch die bereits erwähnten Schrankeneigenschaften und durch die Tatsache, dass die Menge U nur endliche viele Elemente enthält. In der i-ten Iteration des relaxierten Masterproblems tritt explizit die Schrankenbedingung

$$z^{i-1} \leq \mathbf{c}^{\mathrm{T}}\mathbf{y} + \mu \leq z^*$$

auf. Andererseits impliziert der Fall 3a) (keine weitere zulässige MILP-Lösung) die Bedingung

$$\mathbf{c}^{\mathrm{T}}\mathbf{y} + \mu \geq z^* \quad ,$$

d. h. die Schranken kreuzen sich, womit gezeigt ist, dass $(\mathbf{x}^*, \mathbf{y}^*)$ Lösung ist mit z^*.

A.4.8 Optimalität

Ersetze „$z^{i-1} \leq \mathbf{c}^{\mathrm{T}}\mathbf{y} + \mu < z^*$" durch „$z^{i-1} \leq \mathbf{c}^{\mathrm{T}}\mathbf{y} + \mu \leq z^*$" und (3a) durch: „STOP, wenn $\mathbf{y}^*$ sich wiederholt". Setze

$$\begin{aligned} z^i &= \mathbf{c}^{\mathrm{T}}\mathbf{y}^* + \mu^* = z^* \quad , \quad \mu^* = f(\mathbf{y}^*) \\ z^{i-1} &= \mathbf{c}^{\mathrm{T}}\mathbf{y}^* + \mu^* = z^* \quad . \end{aligned}$$

Damit lässt sich das relaxierte Masterproblem $\mathbf{M}^i$ lediglich durch die letzte Äußere-Approximation, nämlich die durch $\mathbf{x}^*$ erzeugte, ausdrücken als

$$\begin{aligned} & & z^i &= \min\{\mathbf{c}^{\mathrm{T}}\mathbf{y} + \mu\} \\ \textit{u.d.N.} \quad & f(\mathbf{x}^*) + \boldsymbol{\nabla} f(\mathbf{x}^*)^{\mathrm{T}}(\mathbf{x} - \mathbf{x}^*) - \mu &\leq 0 \quad , \\ & \mathbf{g}(\mathbf{x}^i) + \boldsymbol{\nabla}\mathbf{g}(\mathbf{x}^*)^{\mathrm{T}}(\mathbf{x} - \mathbf{x}^*) + B\mathbf{y} &\leq 0 \\ & & z^i &= z^* \quad , \\ & & \mathbf{x} &\in X \ , \ \mathbf{y} \in U \cap \{\text{Integer Cuts}\}, \ \mu \in [f^L, f^U] \end{aligned} \tag{A.4.39}$$

und das Verfahren terminiert mit

$$\mathbf{y}^t = \mathbf{y}^* \quad , \quad \forall t \geq i+1 \quad .$$

Theorem 10 *Falls* $(\mathbf{x}^*, \mathbf{y}^*)$ *optimal ist für ein* i*, d. h.* $(\mathbf{x}^{i+1}, \mathbf{y}^{i+1})$*, dann ist* $(\mathbf{x}^t, \mathbf{y}^t)$ *optimal für* P*.*

Theorem 11 *Seien alle Annahmen und Eigenschaften erfüllt, insbesondere der letzte Satz und die Schranken-Eigenschaften. Ist* U *eine endliche, diskrete Menge, so terminiert das Äußere-Approximation-Verfahren in endlich vielen Schritten.*

A.4.9 Erweiterung des Äußere-Approximation-Verfahrens

A.4.9.1 Einbeziehung von Gleichungen

Die bisherige Überlegungen und das Ausgangsproblem **P** beinhalteten nur Ungleichungsbedingungen, nicht jedoch Gleichungen. Hinsichtlich (kontinuierlich) nichtlinearer Optimierungsprobleme bereiten Gleichungsbedingungen keine Probleme, wohl aber bei MINLP, da diese letztlich auf Linearisierung in der Outer Approximation oder verwandten Verfahren basieren. Ohne Probleme lassen sich lineare Gleichungen der Form $\mathbf{l}(\mathbf{x},\mathbf{y}) = 0$ dem Problem hinzufügen, da diese sich hinsichtlich der Linearisierungspunkte $\mathbf{x}^i$ invariant verhalten und Algorithmen zur Lösung beliebiger linearer Probleme verfügbar sind.

Die Berücksichtigung nichtlinearer Gleichungen

$$\mathbf{h}(\mathbf{x},\mathbf{y}) = 0 \quad , \quad \mathbf{h} : \mathbb{R}^{n_c+n_d} \rightarrow \mathbb{R}^{n_h}$$

bereitet jedoch zwei verschiedenartige Schwierigkeiten:

- es ist nicht möglich, die linearisierten Gleichungen in k Linearisierungspunkten zu erzwingen, und
- die nichtlinearen Gleichungen zerstören meist die konvexe Struktur des Problems.

Nachfolgend wird eine Variante des OA-Algorithmus beschrieben, die Gleichungsbeschränkungen behandeln kann (OA/ER=Outer Approximation/Equality Relations). Kocis & Grossmann (1988,[182]) schlugen eine Relaxierungsstrategie vor, in der in den Master-Problemen die nichtlinearen Gleichungen durch Ungleichungen

$$T^i \nabla \mathbf{h} \left(\mathbf{x}^i, \mathbf{y}^i\right)^{\mathrm{T}} \left[\begin{array}{c} \mathbf{x} - \mathbf{x}^i \\ \mathbf{y} - \mathbf{y}^i \end{array} \right] \leq 0 \tag{A.4.40}$$

ersetzen werden, wobei die Diagonalmatrix

$$T^i := \left\{ t^i_{jj} \right\} \in \mathcal{M}(n_h, n_h)$$

die Vorzeichen der mit der Gleichung $h_i(\mathbf{x},\mathbf{y}) = 0$ assoziierten Lagrange-Multiplikatoren enthält und bezüglich der Ungleichungen die Funktion einer Richtungsmatrix übernimmt und $\mathbf{x}^k$ und $\mathbf{y}^k$ Linearisierungspunkte sind. Dabei liegt folgende Vorgehensweise zugrunden: Im i-ten Iterationsschritt wurde der Vektor $\mathbf{y}$ auf den Wert $\mathbf{y}^i$ fixiert. Die Lösung des nichtlinearen kontinuierlichen Minimierungsproblems $\mathbf{S}(\mathbf{y}^i)$ unter Einbeziehung der nichtlinearen Gleichungen $\mathbf{h}(\mathbf{x},\mathbf{y}) = 0$, d. h. des nichtlinearen kontinuierlichen Minimierungsproblems $\mathbf{NLP1}(\mathbf{y}^i)$

$$\begin{array}{rlll} & \mathbf{NLP1}(\mathbf{y}^i) & : & z(\mathbf{y}^i) = \mathbf{c}^{\mathrm{T}}\mathbf{y}^i + \min f(\mathbf{x}) \\ u.d.N. & \mathbf{g}(\mathbf{x}) + B\mathbf{y}^i & \leq & 0 \quad , \quad \mathbf{h}(\mathbf{x},\mathbf{y}) = 0 \quad , \quad \mathbf{x} \in X \quad , \end{array} \tag{A.4.41}$$

liefert dann den neuen Vektor $\mathbf{x}^i$ und den Vektor der Lagrange-Multiplikatoren $\boldsymbol{\lambda}$.

Die Relaxierung besteht nun darin, das Problem $\mathbf{NLP1}(\mathbf{y}^i)$ durch

$$\begin{array}{rlll} & \mathbf{NLP2}(\mathbf{y}^i) & : & z(\mathbf{y}^i) = \mathbf{c}^{\mathrm{T}}\mathbf{y}^i + \min f(\mathbf{x}) \\ u.d.N. & \mathbf{g}(\mathbf{x}) + B\mathbf{y}^i & \leq & 0 \quad , \quad T^k\mathbf{h}(\mathbf{x},\mathbf{y}) \leq 0 \quad , \quad \mathbf{x} \in X \end{array} \tag{A.4.42}$$

zu ersetzen. Mit Hilfe der KKT- und unter bestimmten Voraussetzungen wie Konvexität von $\mathbf{h}(\mathbf{x},\mathbf{y})$ und $T^i\mathbf{h}(\mathbf{x},\mathbf{y})$ kann die Äquivalenz von **NLP1** und **NLP2** gezeigt werden. Linearisierung von **NLP2** liefert dann (A.4.40).

Relaxieren diese Gleichungen die Ungleichungen $\mathbf{h}(\mathbf{x},\mathbf{y}) \leq 0$ für alle $\mathbf{y}$, und ist $\mathbf{h}(\mathbf{x},\mathbf{y})$ konvex, so stellt obige Relaxierung ein strenges Verfahren dar. Andernfalls werden durch die Tangentialebenen an den Linearisierungspunkte unzulässige Bedingungen generiert, die einen Teil des zulässigen Bereichs abschneiden. Dieser Fall kann dann aber mit den Erweiterung des OA-Algorithmus für nichtkonvexe Probleme behandelt werden.

Hierzu sei als Beispiel das folgende Problem

$$\begin{array}{rrcl} & \min z & = & -y + 2x_1 + x_2 \\ u.d.N. & x_1 - 2e^{-x_2} & = & 0 \quad , \quad -x_1 + x_2 + y \leq 0 \quad , \quad 0.5 \leq x_1 \leq 1.4 \quad , \quad y \in \{0,1\} \end{array}$$

aus Duran & Grossmann (1986,[75]) betrachtet. Die beiden zu $y = 0$ und $y = 1$ gehörigen NLP-Probleme haben die Lösungen:

$$\begin{array}{rrcll} 1. & y & = & 0: \quad z = 2.558 & , \quad \mathbf{x}^{\mathrm{T}} = (0.853, 0.853) \quad , \quad \lambda = -1.619 < 0 \\ 2. & y & = & 1: \quad z = 2.124 & , \quad \mathbf{x}^{\mathrm{T}} = (1.375, 0.375) \quad . \end{array}$$

Offensichtlich ist die zweite Lösung die optimale. Um die Wirkungsweise des OA/ER zu demonstrieren, sei angenommen, dass das Verfahren mit $y = 0$ begonnen wurde. Die Lösung von **NLP1** ist gerade wieder die erste Lösung. Die Richtungsmatrix ist damit $T^1 = sign(\lambda^1) = -1$. Die Relaxierung der Gleichungsbedingung folgt daraus zu

$$T^1 \cdot h(x) \leq 0 \quad \Rightarrow \quad -1 \cdot \left(x_1 - 2e^{-x_2}\right) \leq 0 \quad \Leftrightarrow \quad 2e^{-x_2} - x_1 \leq 0 \quad .$$

Wegen der Konvexität der Funktion $2e^{-x_2} - x_1$ liefert ihre Linearisierung im Punkt $\mathbf{x}^1 = (0.853, 0.853)^{\mathrm{T}}$ eine zulässige und strenge Approximierung im Masterproblem, das damit die Bestimmung der optimalen Lösung ermöglicht.

A.4.9.2 Behandlung nichtkonvexer MINLP-Probleme

Strikt betrachtet, sollten derartige Probleme mit Methoden der Globalen Optimierung behandelt werden, d. h. man versucht, dass Problem mit strukturierten Methoden zu lösen. Hier können z. B. die in Abschnitt 4.6 beschriebenen konvexen Unterschätzer verwendet werden, mit denen ein vorliegendes Problem in eine Folge konvexer MINLP-Problem überführt wird. Empfehlenswert ist der Einsatz deterministischer, globaler, bis auf Rundungsfehlern exakter Optimierungsverfahren, die beispielsweise in Solvern wie αBB, BARON, LINDOGLOBAL oder GLOMIQO implementiert sind.

Steht aber keine Globale Optimierungssoftware zur Verfügung oder können in vertretbarer Zeit keine global optimalen Lösungen bestimmt werden, so bleibt nichts anderes übrig, als unstrukturierte Verfahren anzuwenden, die mit Hilfe heuristischer Strategien versuchen, die durch die nichtkonvexe Strukturen verursachten Effekte soweit wie möglich zu reduzieren. Werden nichtkonvexe nichtlineare Problemen mit Hilfe lokaler NLP-Verfahren gelöst, so treten bei Outer-Approximations-Verfahren insbesondere die Schwierigkeiten auf, dass

- lokale Algorithmus zur Lösung von NLP-Problemen nicht notwendigerweise das globale Optimum liefern,

- die MILP-Masterprobleme keine rigorosen Schranken liefern.

Mit Hilfe des von Viswanathan & Grossmann (1990,[286]) entwickelten Augmented-Penalty-Verfahrens lassen sich dennoch viele Probleme approximativ im Sinne der Outer Approximation lösen. Dieses Verfahren beruht auf der Überlegung, dass es bei nichtkonvexen Problemen geschehen kann, dass durch die Tangentialebenen in den Linearisierungspunkten Bereiche des zulässigen Gebietes abgeschnitten werden. Bedenkt man, dass eine Ebene

$$ax + by + cz = 0$$

durch den Koordinatenursprung parallel zum Normalenvektor

$$\mathbf{n} = (a, b, c)^{\mathrm{T}} / \sqrt{a^2 + b^2 + c^2}$$

verschoben werden kann und die verschobene Ebene

$$ax + by + cz = d$$

vom Koordinatenursprung $(0, 0, 0)^{\mathrm{T}}$ den Abstand d hat, besteht die Grundidee des erweiterten Verfahrens (Outer Approximation & Augmented Penalty & Equality Relaxation) darin, mit Hilfe von Schlupfvariablen die Wahrscheinlichkeit von unzulässigen Reduzierungen des zulässigen Bereichs zu verringern. Das APER-Masterproblem im k-ten Iterationsschritt lautet damit:

$$\begin{aligned} (\mathbf{M} - \mathbf{APER})^k \quad : \qquad z^k &= \min \left\{ \mu + \sum_{i=1}^{k} w_p^i p^i + w_q^i q^i \right\} \\ (\mathbf{x}, \mathbf{y}) \quad &\in \quad \Omega^k \ , \ \mathbf{x} \in X \ , \ \mathbf{y} \in U \ , \ \mu \in [f^L, f^U] \end{aligned} \tag{A.4.43}$$

mit

$$\Omega^k := \left\{ \begin{array}{c} (\mathbf{x}, \mathbf{y}) \mid f(\mathbf{x}^i) + \boldsymbol{\nabla} f(\mathbf{x}^i)^{\mathrm{T}} (\mathbf{x} - \mathbf{x}^i) \leq \mu \quad , \quad \forall i \in O^k \\ T^i \nabla \mathbf{h} \left(\mathbf{x}^i, \mathbf{y}^i \right)^{\mathrm{T}} \left[\begin{array}{c} \mathbf{x} - \mathbf{x}^i \\ \mathbf{y} - \mathbf{y}^i \end{array} \right] \leq p^i \\ \mathbf{g}(\mathbf{x}^i) + \boldsymbol{\nabla} \mathbf{g}(\mathbf{x}^i)^{\mathrm{T}} (\mathbf{x} - \mathbf{x}^i) + B\mathbf{y} \leq q^i \quad , \quad \forall i \in O^k \end{array} \right\} \quad , \quad \forall \mathbf{x}^i \ \in O^k$$

und

$$O^k := \left\{ i \mid \mathbf{x}^i \text{ ist Optimallösung von } S(\mathbf{y}^i) \ , \ \mathbf{y}^i \in U \cap V \ , \ i = 1, 2, \ldots, k \right\} \quad , \tag{A.4.44}$$

wobei $w_p^i \geq 0$ und $w_q^i \geq 0$ hinreichend große Gewichte sein sollten, etwa 1000 mal so groß wie die Lagrange-Multiplikatoren. Sind die im Master-Problem auftretenden Funktionen konvex, so liefert M-APER rigorose Schranken mit verschwindenden Schlupfvariablen.

B Glossar

Die im Buch verwendeten technischen Begriffe aus dem Umfeld der mathematischen Optimierung sind hier der Übersichtlichkeit wegen noch einmal zusammengefasst; Verweise sind in Fettschrift hervorgehoben.

Abbruchkriterium [engl.: *termination criterion*]: Mathematisches Kriterium, das bei Erfüllung zum Abbruch oder zur Terminierung eines iterativen **Algorithmus** führt; Beispiel: das Unterschreiten einer Fehlerschranke.

Algorithmus [engl.: *algorithm*]: Ein systematisches deterministisches Verfahren zur Lösung eines bestimmten Problems, das durch eine endliche Anzahl von Verfahrensschritten beschrieben werden kann, wobei jedem Verfahrensschritt eindeutig der nächste Schritt folgt.

Akzeptanzwert [engl.: *cut off*]: In einem B&B-Verfahren der Zielwert, den ein weiterer ganzzahlig zulässiger Punkt erreichen muss, damit er weiter fortgeführt wird.

Äußere-Approximation [engl.: *outer approximation*]: **Algorithmus** zur Lösung von **MINLP**-Problemen, der auf einer äquivalenten Darstellung des zulässigen Bereichs durch einen Schnittmenge von Hyperebenen beruht. Die Hyperebenen werden durch Linearisierung bzw. Taylorreihenentwicklung der Nebenbedingungen gewonnen.

Basis (zulässige Basislösung) [engl.: *basic feasible solution*]: In einem LP-Problem mit den Nebenbedingungen $\mathsf{A}\mathbf{x} = \mathbf{b}$ und $\mathbf{x} \geq 0$ die Menge der m linear unabhängigen Spaltenvektoren der $m \times n$ Matrix A, die eine reguläre Matrix $\mathcal{B}$ bilden. Der Vektor $\mathbf{x}_B = \mathcal{B}^{-1}\mathbf{b}$ heißt Basislösung; manchmal wird auch dieser Vektor Basis genannt. Gilt $\mathbf{x}_B \geq 0$, so heißt $\mathbf{x}_B$ *zulässige Basislösung.*

Basisvariablen [engl.: *basic variables*]: Diejenigen Variablen eines linearen Optimierungsproblems, deren Werte, in nichtentarteten Fällen, nicht an den unteren oder oberen **Schranken** fixiert sind und eindeutig aus der Lösung des linearen Gleichungssystems $\mathcal{B}\mathbf{x}_B = \mathbf{b}$ folgen.

Branch & Bound: Ein zur Gruppe der Verzweigungsverfahren gehörendes, implizites Enumerationsverfahren zur Lösung ganzzahliger und gemischt-ganzzahliger Optimierungsprobleme. Das Hauptproblem wird dabei mit Hilfe geeignet konstruierter Unterprobleme, die auch Zweige [engl.: *branches*) genannt werden, in Form eines Suchbaumes abgearbeitet, wobei die einzelnen Zweige hinsichtlich der Zielfunktion bewertet und zur Generierung von Schranken für den Wert der **Zielfunktion** ausgewertet werden.

Branch & Cut: Ein implizites Enumerationsverfahren zur Lösung ganzzahliger und gemischt-ganzzahliger Optimierungsprobleme, das kombiniert mit Branch&Bound-Verfahren zusätzliche Ungleichungen (Schnittebenen, [engl.: cuts]) erzeugt, die unerwünschte nicht-ganzzahlige Punkte oder Gebiete zwischen LP-Relaxierung und konvexer Hüelle abtrennen.

J. Kallrath, *Gemischt-ganzzahlige Optimierung: Modellierung in der Praxis*,
DOI 10.1007/978-3-658-00690-7, © Springer Fachmedien Wiesbaden 2013

Dualität [engl.: *duality*]: Ein sehr nützliches Konzept in der Optimierungstheorie, welches das ursprüngliche (primale) Optimierungsproblem mit seinem dualen Problem verknüpft.

Dualitätslücke [engl.: *duality gap*]: Für zulässige Punkte eines Optimierungsproblems die Differenz der Zielfunktionswerte des primalen und dualen Problems. Bei LP-Problemen ist die Dualitätslücke identisch Null.

Duales Optimierungsproblem [engl.: *dual optimization problem*]: Ein aus dem primalen Problem mit Hilfe der Lagrange-Funktion abgeleitetes Optimierungsproblem. Das duale Problem eines LP-Problems ergibt sich aus dem Austausch der Zielfunktion und der rechten Seite der Nebenbedingungen und der Transposition der Systemmatrix.

Dualwerte [engl.: *dual values*]: Die dualen Werte sind die Werte der Variablen des dualen Optimierungsproblems. Der Begriff wird auch synonym zu den Schattenpreisen verwendet.

Ganzzahligkeitslücke [engl.: *integrality gap*]: Die Differenz zwischen dem optimalen Zielfunktionswert eines gemischt-ganzzahligen Optimierungsproblems und seiner Relaxierung.

Gemischt-ganzzahlige lineare Optimierung [engl.: *Mixed Integer Linear Programming; MILP*]: Eine Erweiterung der linearen Optimierung, bei der nun einige Variablen binär, ganzzahlig, halb-stetig oder partiell-ganzzahlig sind.

Gemischt-ganzzahlige nichtlineare Optimierung [engl.: *Mixed Integer Nonlinear Programming; MINLP*]: Eine Erweiterung der nichtlinearen Optimierung, bei der nun einige Variablen binär, ganzzahlig, halb-stetig oder partiell-ganzzahlig sind.

Globales Optimum [engl.: *global optimum*]: Ein zulässiger Punkt $\mathbf{x}^*$ eines Optimierungsproblems, für den die Zielfunktion einen Wert annimmt, der besser ist als die Zielfunktionswerte $f(\mathbf{x})$ aller anderen zulässigen Punkte; in einem Maximierungsproblem gilt $f(\mathbf{x}^*) \geq f(\mathbf{x})$ für alle $\mathbf{x} \neq \mathbf{x}^*$.

Graph [engl.: *graph*]: Ein aus Knoten und Kanten bestehendes mathematisches Objekt, das sehr nützlich bei der Beschreibung von Netzwerk-Fluss-Problemen ist. Die Eigenschaften solcher Graphen werden in der Graphentheorie untersucht.

Kante [engl.: *arc, edge*]: In einem **Graphen** die Verbindungselemente (Linien), die jeweils zwei **Knoten** des Graphen verbinden. In graphentheoretisch modellierten Problemen stellen die Kanten häufig Straßen, Rohrleitungen oder andere Wege dar, entlang denen extensive Größen fließen können.

Knoten [engl.: *node*]: Ein Objekt eines **Graphen**. Knoten stellen meist Anlagen, Depots oder andere Bilanzpunkte in einem Graphen dar und sind durch **Kanten** verbunden.

Konvexe Menge [engl.: *convex set*]: Eine Menge eines mehrdimensionales Raumes, in der die Verbindungslinie zweier Punkte der Menge komplett innerhalb der Menge enthalten ist.

Kuhn-Tucker-Bedingungen [engl.: *Kuhn-Tucker conditions*]: die notwendigen und hinreichenden Bedingungen für die Existenz eines lokalen Optimums eines kontinuierlichen nichtlinearen Optimierungsproblems. Richtig müsste man sie Karush-Kuhn-Tucker-Bedingungen nennen.

Linearkombination [engl.: *linear combination*]: Die lineare Kombination der Vektoren $\mathbf{v}_1, ..., \mathbf{v}_n$ ist der Vektor $\sum_i a_i \mathbf{v}_i$ mit reell- oder komplexwertigen Koeffizienten a_i. Die *triviale* **Linearkombination** erhält man, indem man alle Vektoren mit Null multipliziert und sie dann addiert, d. h. $a_i = 0$ für alle i.

Lineare Funktion [engl.: *linear function*]: Eine Funktion $f(\mathbf{x})$ eines Vektors $\mathbf{x}$ mit konstantem Gradienten $\boldsymbol{\nabla} f(\mathbf{x}) = \mathbf{c}$. In diesem Fall ist $f(\mathbf{x})$ von der Form $f(\mathbf{x}) = \mathbf{c}^{\mathrm{T}}\mathbf{x} + \alpha$, wobei α ein beliebiger fester Skalar ist.

Lineare Unabhängigkeit [engl.: *linear independence*]: Eine Menge von Vektoren heißt linear unabhängig, wenn es keine nichttriviale Linearkombination des Nullvektors gibt.

Lineare Programmierung [engl.: *linear programming; LP*]: Teildiziplin der mathematischen Optimierung, die sich mit kontinuierlichen linearen Optimierungsproblemen und ihrer Lösung beschäftigt; Nebenbedingungen und Zielfunktion sind hier **lineare Funktionen**.

Lokales Optimum [engl.: *local optimum*]: Ein **zulässiger Punkt** $\mathbf{x}^*$ eines Optimierungsproblems, für den die **Zielfunktion** einen Wert annimmt, der besser ist als die Zielfunktionswerte $f(\mathbf{x})$ aller anderen zulässigen Punkte in einer lokalen Umgebung $U_{\mathbf{x}^*}$ um $\mathbf{x}^*$; in einem Maximierungsproblem gilt $f(\mathbf{x}^*) \geq f(\mathbf{x})$ für alle $\mathbf{x} \in U_{\mathbf{x}^*}$.

Matrix [engl.: matrix]: Darstellung eines linearen Operators. Bei Wahl einer geeigneten Basis ergibt sich daraus ein rechteckiges Koeffiziententableau. Für Matrizen lassen sich Addition, Subtraktion, Multiplikation sowie unter bestimmten Voraussetzungen die Konstruktion der inversen Matrix definieren.

Mengen mit Ordnungsstruktur [engl.: *special ordered sets*]: Mengen von **Variablen**, in denen nur eine bestimme Anzahl von Variablen von Null verschieden sein können und in denen hinsichtlich der Indizierung eine Ordnungsstruktur vorliegt. Diese Mengen erlauben spezielle und effiziente Verzweigungsregeln in Branch&Bound-Verfahren.

Minimalverbesserung [engl.: *addcut*]: In einem B&B-Verfahren die Differenz, um die der Zielfunktionswert eines neuen ganzzahlig zulässigen Punkt den Wert eines bereits existierenden ganzzahlig zulässigen Punktes übertreffen muss, damit er akzeptiert wird.

Modell [engl.: *model*]: Eine mathematische Darstellung eines Problems mit Hilfe mathematischer Symbole und Terme. In einem mathematischen Optimierungsmodell sind die wesentlichen Bestandteile die **Variablen**, die **Nebenbedingungen** und die **Zielfunktion**.

Nebenbedingungen [engl.: *constraints*]: In der mathematischen Optimierung algebraisch formulierte Gleichungen und Ungleichungen, die die **Variablen** und die Daten eines Optimierungsmodells verknüpfen.

Netzwerk [engl.: *network*]: Eine Darstellung eines Problems durch eine Menge von durch gerichtete oder ungerichtete **Kanten** verbundene **Knoten**, die in einem **Graphen** zusammengefasst sind.

Nichtbasisvariable [engl.: *non-basic variables*]: Die Variablen in einem LP-Problem, die gleich ihrer oberen oder unteren Schranke gesetzt sind.

Nichtlineare Funktion [engl.: *nonlinear function*]: Eine Funktion $f(\mathbf{x})$ eines Vektors $\mathbf{x}$ mit nichtkonstantem Gradienten $\nabla f(\mathbf{x})$.

Nichtlineare Optimierung [engl.: *nonlinear Programming; NLP*]: Teildiziplin der mathematischen Optimierung, die sich mit kontinuierlichen nichtlinearen Optimierungsproblemen und ihrer Lösung beschäftigt; Nebenbedingungen und Zielfunktion sind hier **nichtlineare Funktionen**.

$\mathcal{NP}$-Vollständigkeit [engl.: *$\mathcal{NP}$ completeness*]: Charakterisierung eines Problems oder einer Klasse von Problemen hinsichtlich der Komplexität der Lösbarkeit. Der Rechenaufwand zur Lösung von Problemen in der Klasse der **$\mathcal{NP}$-vollständigen Probleme** wächst meist exponentiell mit der Anzahl der Variablen oder einer anderen, die Problemgröße charakterisierenden Maßzahl. Gelänge es, für eines der Probleme in dieser Komplexitätsklasse einen polynomialen Algorithmus zu konstruieren, so könnten damit alle übrigen ebenfalls in polynomialer Zeit gelöst werden.

Optimierung [engl.: *optimization*]: Der Vorgang, hinsichtlich eines vorgegebenen Zieles bzw. einer Zielfunktion die beste Lösung zu finden und nachzuweisen, dass die optimale Lösungen bestimmt wurde.

Optimum (optimale Lösung) [engl.: *optimal solution*]: Ein zulässiger Punkt eines Optimierungsproblems, der bezüglich einer bestimmten Zielfunktion nicht schlechter ist als jeder andere zulässige Punkt.

Parametrische Programmierung [engl.: *parametric programming*]: Verfahren im Rahmen der Sensitivitätsanalyse.

Pivot [engl.: *pivot element*]: Ein spezifisches Element einer Matrix, das bei der Transformation linearer Gleichungen wichtig ist. Im Zusammenhang mit der Lösung linearer Gleichungssysteme wird das Pivot-Element hinsichtlich der numerischen Stabilität gewählt. In der LP wird das Pivot-Element bei der Wahl einer neuen Basisvariable durch das *Pricing* und der *minimum-ratio-Regel* bestimmt. Im **Simplexverfahren** wird der Übergang von einer Basis-Lösung zu einer anderen manchmal auch als Pivot-Schritt bezeichnet.

Polylithische Modellierungs- und Lösungsansätze [engl.: *polylithic modeling and solution approaches*]: Modellierungs- und Lösungsverfahren zur Lösung schwieriger Optimierungsprobleme, in denen verschiedene Modelle und ihre Lösungen algorithmisch verknüpft sind, um das gegebene Ursprungsmodell zu lösen.

Post-optimale Analyse [engl.: *postoptimal analysis*]: Untersuchung der Lösung, Ableitung weiterer interessierender Größen und Durchführung der Sensitivitätsanalyse.

Presolve: Verfahren zur Eliminierung redundanter Nebenbedingungen, Fixierung von Variablen oder in der gemischt-ganzzahligen Optimierung Verschärfung von Nebenbedingungen, bevor der eigentliche Lösungsalgorithmus zur Anwendung kommt.

Ranging: Im Rahmen der **Sensitivitätsanalyse** die Untersuchung der Grenzen von Koeffizientenänderung der Eingangsdaten eines linearen Optimierungsproblems, in denen sich die optimale Lösung nicht fundamental ändert.

Reduzierte Kosten [engl.: *reduced costs*]: In einem linearen Minimierungsproblem (Maximierungsproblem) die Kosten (der Gewinn), die sich ergeben, wenn eine Nichtbasisvariable von ihren Schranken gelöst wird.

Relaxierung [engl.: *relaxation*]: In der mathematischen Optimierung die Generierung eines Hilfsoptimierungsproblems, in dem einige **Nebenbedingungen** des ursprünglichen Problems vernachlässigt werden oder der zulässige Wertebereich einiger Variablen vergrößert wird.

Schattenpreis [engl.: *shadow price*]: Die differentielle Änderung des Zielfunktionswertes der optimalen Lösung bei differentieller Änderung des Wertes der rechten Seite einer Nebenbedingung. Die Schattenpreise werden auch die **dualen Werte** genannt.

Schlupfvariablen [engl.: *slack variables*]: Nichtnegative Hilfsvariablen, die mit negativem Koeffizienten in $\leq$ Ungleichungen eingeführt werden, um die Ungleichungen in Gleichungen zu überführen.

Schnittebenen [engl.: *cutting-planes*]: Zusätzliche zulässige Ungleichungen, die zu einem **MILP**-Modell hinzugefügt werden, um den Abstand der LP-Relaxierung zur konvexen Hülle zu verkleinern.

Schranken [engl.: *bounds*]**:** Schranken sind spezielle Nebenbedingungen, die nur eine Variable enthalten und deren Koeffizient gleich eins ist.

Sensitivitätsanalyse [engl.: *sensitivity analysis*]: Untersuchung der optimalen Lösung hinsichtlich differentieller Änderungen der Eingangsdaten.

Sequentielle Lineare Programmierung [engl.: *successive linear programming; SLP*]: Algorithmus zur Lösung nichtlinearer kontinuierlicher Optimierungsprobleme mit einer nicht zu großen Anzahl von nichtlinearen Termen in den Nebenbedingungen und in der Zielfunktion.

Simplexverfahren: Algorithmus zur Lösung von LP-Problemen, der die Eckpunkte des zulässigen Bereichs untersucht.

Skalierung [engl.: *scaling*]: Transformation der Daten und möglichen Variablenwerte eines Problems in einen numerisch sinnvollen Wertebereich; dies ist allerdings nicht immer möglich. In der linearen Optimierung speziell die Reduzierung der Variabilität der Koeffizienten der Systemmatrix durch eine Reihe von Zeilen- und Spaltenoperationen.

Solver: In der Gemeinschaft der mathematischen Optimierer, insbesondere unter den Verwendern algebraischer Modellierungssprachen, eine häufig verwendete Kurzbezeichnung für den Lösungsalgorithmus, der als Software zur Lösung eines Optimierungsproblems aufgerufen wird, bzw. für das Programm, in welches der Lösungsalgorithmus einer bestimmten Firma implementiert wurde.

Stochastische Optimierung [engl.: *stochastic programming*]: Eine Unterdisziplin der mathematischen Optimierung, die sich mit der Lösung von Optimierungsproblemen beschäftigt, in denen einige der Eingangsdaten mit stochastischen Unsicherheiten unterliegen.

Traveling salesman problem: Ein sehr schwieriges Optimierungsproblem, das darin besteht, den kostenminimalen geschlossenen Reiseweg zu finden, der eine vorgegebene Menge von Städten genau einmal besucht.

Überschussvariablen [engl.: *surplus variables*]: Nichtnegative Hilfsvariablen, die mit negativem Koeffizienten in $\geq$ Ungleichungen eingeführt werden, um die Ungleichungen in Gleichungen zu überführen.

Unbeschränktes Problem [engl.: *unbounded problem*]: Ein Optimierungsproblem, in dem keine optimale Lösung existiert, da die Zielfunktion in einem Maximierungsproblem beliebig große positive oder in einem Minimierungsproblem beliebig große negative Werte annehmen kann.

Unimodularität [engl.: *unimodularity*]: Eigenschaft einer Matrix. Eine quadratische Matrix heißt unimodular, wenn ihre Determinante den Wert +1 hat. In einem LP-Problem mit Nebenbedingungen $\mathsf{A}\mathbf{x} = \mathbf{b}$ heißt die Matrix A unimodular, wenn sämtliche Untermatrizen Determinanten mit den Werten +1, 0 oder −1 haben. Ist A unimodular und sind alle Komponenten des Vektors $\mathbf{b}$ ganzzahlig, so sind alle zulässigen Basislösung des LP-Problems ebenfalls ganzzahlig.

Unzulässiges Problem [engl.: *infeasible problem*]: Ein Optimierungsproblem, das keinen **zulässigen Punkt** besitzt.

Variable [engl.: *variable*]: Ein mathematisches Objekt, das in einem Optimierungsmodell eine Entscheidung oder eine unbekannte Größe bezeichnet, die es zu bestimmen gilt. Die Variablen werden auch *Unbekannte*, in der linearen Optimierungswelt auch häufig *Spalten* [engl.: *columns*) genannt.

Vektor [engl.: *vector*]: In der linearen Algebra ein mathematisches Objekt mit bestimmten Transformationseigenschaften; in MINLP-Problemen der Form (1.8.1) sind alle Variablen in den Vektoren $\mathbf{x}$ und $\mathbf{y}$ zusammengefasst.

Zielfunktion [engl.: *objective function*]: Mathematischer Term eines Optimierungsproblems, der maximiert oder minimiert werden soll.

Ziel-Programmierung [engl.: *goal programming*]: Ein Verfahren zur Lösung multikriterieller Optimierungsprobleme.

Zulässiger Bereich [engl.: *feasible region*]: In einem Optimierungsproblem die Menge aller **zulässigen Punkte**.

Zulässiges Problem [engl.: *feasible problem*]: Ein Optimierungsproblem, das zumindest einen **zulässigen Punkt** besitzt.

Zulässiger Punkt [engl.: *feasible point*]: Ein Punkt bzw. Vektor, der allen **Nebenbedingungen** eines Problems genügt.

Literaturverzeichnis

[1] E. H. L. Aarts und J. H. M. Korst. *Simulated Annealing and Boltzmann Machines*. Wiley, Chichester, UK, 1989.

[2] E. H. L. Aarts, J. H. M. Korst, und P. J. M. van Laarhoven. Simulated Annealing. In E. H. L. Aarts und J. K. Lenstra, editors, *Local Search in Combinatorial Optimization*, Seite 91–120. Wiley, Chichester, UK, 1997.

[3] J. Abadie. The GRG Method for Nonlinear Programming. In H. J. Greenberg, editor, *Design and Implementation of Optimization Software*, Seite 335–363. Sijthoff and Noordhoff, The Netherlands, 1978.

[4] J. Abadie und J. Carpenter. Generalization of the Wolfe Reduced Gradient Method to the Case of Nonlinear Constraints. In R. Fletcher, editor, *Optimization*, Seite 37–47. Academic Press, New York, 1969.

[5] C. Adjiman, S. Dallwig, C. Floudas, und A. Neumaier. A Global Optimization Method, αBB, for General Twice-differentiable Constrained NLPs - I. Theoretical Advances. *Computers and Chemical Engineering*, 22:1137–1158, 1998.

[6] C. Adjiman, S. Dallwig, C. Floudas, und A. Neumaier. A Global Optimization Method, αBB, for General Twice-differentiable Constrained NLPs - II. Implementation and Computational Results. *Computers and Chemical Engineering*, 22:1159–1179, 1998.

[7] C. Adjiman und C. Floudas. The αBB Global Optimization Algorithm for Nonconvex Problems: An Overview. In A. Migdalas, P. Pardalos, und P. Värbrand, editors, *From Local to Global Optimization*, Seite 155–186. Kapitel 8, Kluwer Academic Publishers, Dordrecht, The Netherlands, 2001.

[8] C. S. Adjiman und C. A. Floudas. Rigorous Convex Underestimators for General Twice-Differentiable Problems. *Journal of Global Optimization*, 9:23–40, 1996.

[9] C. S. J. Adjiman. *Global Optimization Techniques for Process Systems Engineering*. PhD Dissertation, Dept. of Chemical Engineering, Princeton University, Princeton, NJ, 1999.

[10] F. A. Al-Khayyal. Jointly Constrained Bilinear Programs and Related Problems: An Overview. *Computers Math. Applic.*, 19:53–62, 1990.

[11] F. A. Al-Khayyal und J. E. Falk. Jointly Constrained Biconvex Programming. *Maths Ops Res*, 8:273–286, 1983.

[12] E. D. Andersen. On Exploiting Problem Structure in a Basis Identifications Procedure for Linear Programming. Department Publication 6, Department of Management Sciences, Odense Universitet, Odense, Dänemark, 1996.

[13] E. D. Andersen und K. D. Andersen. Presolving in Linear Programming. *Mathematical Programming*, 71:221–245, 1995.

[14] E. D. Andersen, J. Gonzio, C. Meszaros, und X. Xu. Implementation of Interior Point Methods for Large Scale Linear Programming. Department Publication 1, Department of Management Sciences, Odense Universitet, Odense, Dänemark, 1996.

[15] E. D. Andersen und Y. Ye. Combining Interior-point and Pivoting Algorithms for Linear Programming. Technical report, Department of Management Sciences, The University of Iowa, Ames, Iowa, 1994.

[16] E. D. Andersen und Y. Ye. On a Homogeneous Algorithm for the Monotone Complementary Problem. Technical report, Department of Management Sciences, The University of Iowa, Ames, Iowa, 1995.

[17] J. ao Marques-Silva. Practical Applications of Boolean Satisfiability. In *In Workshop on Discrete Event Systems (WODES*, Seite 74–80, Goteborg, Sweden, 2008. IEEE Press.

[18] A. Arbel. *Exploring Interior-Point Linear Programming Algorithms and Software.* MIT Press, London, 1994.

[19] N. Ascheuer, M. Fischetti, und M. Grötschel. Solving the Asymmetric Travelling Salesman Problem with time windows by branch-and-cut. *Math. Program. Series A*, 90:475–506, 2001.

[20] R. W. Ashford und R. C. Daniel. LP-MODEL XPRESS-LP's Model Builder. *Institute of Mathematics and its Application Journal of Mathematics in Management*, 1:163–176, 1987.

[21] R. W. Ashford und R. C. Daniel. Some Lessons in Solving Practical Integer Problems. *J. Opl. Res. Soc.*, 43(5):425–433, 1992.

[22] E. Balas. An Additive Algorithm for Solving Linear Programs with Zero-One Variables. *Operations Research*, 13:517–546, 1965.

[23] C. Barnhart, E. L. Johnson, G. L. Nemhauser, M. W. P. Savelsbergh, und P. H. Vance. Branch-and-Price: Column Generation for Solving Integer Programs. *Operations Research*, 46:316–329, 1998.

[24] M. Bazaraa, H. D. Sheraldi, und C. M. Shetly. *Nonlinear Programming.* Wiley, Chichester, UK, 2nd edition, 1993.

[25] M. S. Bazaraa und H. D. Sherali. On the Choice of Step Sizes in Subgradient Optimization. *European Journal of Operational Research*, 7:380–388, 1981.

[26] E. M. L. Beale. Integer Programming. In D. A. H. Jacobs, editor, *The State of the Art of Numerical Analysis*, Seite 409–448. Academic Press, London, 1977.

[27] E. M. L. Beale und R. C. Daniel. Chains of Linked Ordered Sets. Technical report, Scicon, Wavendon Tower, Wavendon, Milton Keynes, MK17 8LX, 1980.

[28] E. M. L. Beale und J. J. H. Forrest. Global Optimization Using Special Ordered Sets. *Mathematical Programming*, 10:52–69, 1976.

[29] E. M. L. Beale und J. J. H. Forrest. Global Optimization Using Special Ordered Sets. *Mathematical Programming*, 10:52–69, 1976.

[30] E. M. L. Beale und J. A. Tomlin. Special Facilities in a General Mathematical Programming System for Non-convex Problems Using Ordered Sets of Variables. In J. Lawrence, editor, *OR '69: Proceedings of the 5th International Conference on Operational Research*, Seite 447–454. Tavistock, London, 1970.

[31] R. Bellman. *Dynamic Programming.* Princeton University Press, Princeton, NJ, 1957.

[32] G. Belov und G. Scheithauer. A branch-and-cut-and-price algorithm for one-dimensional stock cutting and two-dimensional two-stage cutting. *European Journal of Operational Research*, 171:85–106, 2006.

[33] A. Ben-Tal und A. Nemirovski. Robust Solutions of Linear Programming Problems Contaminated with Uncertain Data. *Mathematical Programming*, 88:411–424, 2000.

[34] J. F. Benders. Partitioning Procedures for Solving Mixed-variables Programming Problems. *Numerische Mathematik*, 4:238–252, 1962.

[35] T. Benoist und E. Bourreau. Fast Gloabl Filtering for Eternity II. *Constraint Programming Letters*, 3:35–50, 2008.

[36] A. Biere, A. Cimatti, E. M. Clarke, und Y. Zhu. Symbolic Model Checking without BDDs. In R. Cleaveland, editor, *TACAS '99 Proceedings of the 5th International Conference on Tools and Algorithms for Construction and Analysis of Systems*, Seite 193–207, London, UK, 1999. Springer.

[37] J. R. Birge und F. V. Louveaux. *Introduction to Stochastic Programming.* Operations Research and Financial Engineering. Springer, New York, 2000.

[38] J. Bisschop. *AIMMS: Optimization Modeling.* Paragon Decision Technology B.V., Haarlem, The Netherlands, 1999.

[39] R. E. Bixby. Solving Real-World Linear Programs: A Decade and More of Progress. *Operations Research*, 50:3–15, 2002.

[40] C. Bliek, P. Spellucci, L. Vicente, A. Neumaier, L. Granvilliers, E. Monfroy, F. Benhamouand, E. Huens, P. V. Hentenryck, D. Sam-Haroud, und B. Faltings. Algorithms for Solving Nonlinear Constrained and Optimization Problems: The State of the Art. Report of the European Community funded project COCONUT, Mathematisches Institut der Universität Wien, http://www.mat.univie.ac.at/ neum/glopt/coconut/StArt.html, 2001.

[41] H. G. Bock. Randwertproblemmethoden zur Parameteridentifizierung in Systemen nichtlinearer Differentialgleichungen. Preprint 142, Universität Heidelberg, SFB 123, Institut für Angewandte Mathematik, 69120 Heidelberg, 1987.

[42] H. G. Bock und C. Zillober. Interior Point Methods and Extrapolation. In P. Kleinschmidt, editor, *Symposium Operations Research (SOR'95): Program and Abstract of the Annual Conference of the DGOR, GMÖOR and ÖGOR*, Seite 39, Passau, Germany, 1995. University of Passau.

[43] I. M. Bomze und W. Grossmann. *Optimierung – Theorie und Algorithmen.* Wissenschaftsverlag, Mannheim, 1993.

[44] A. L. Brearley, G. Mitra, und H. P. Williams. Analysis of Mathematical Programming Problems Prior to Applying the Simplex Algorithm. *Mathematical Programming*, 8:54–83, 1975.

[45] A. Brooke, D. Kendrick, und A. Meeraus. *GAMS - A User's Guide (Release 2.25)*. Boyd & Fraser Publishing Company, Danvers, Massachusetts, 1992.

[46] P. Brucker, A. Drexl, R. Möhring, K. Neumann, und E. Pesch. Resource-Constrained Project Scheduling: Notation, Classification, Models and Methods. *European Journal of Operational Research*, 112:3–41, 1999.

[47] S. Bunte und N. Kliewer. An Overview on Vehicle Scheduling Models in Public Transport. *Public Transport*, 1:299–317, 2009.

[48] R. E. Burkard. *Methoden der Ganzzahligen Optimierung*. Springer, Wien, New York, 1972.

[49] C. C. Carœ und R. Schultz. Dual Decomposition in Stochastic Integer Programming. *Operations Research Letters*, 24:37–45, 1999.

[50] E. Castillo, A. J. Conejo, P. P. R. Garcia, und N. Alguacil. *Building and Solving Mathematical Programming Models in Engineering and Science*. John Wiley & Sons, New York, US, 2002.

[51] V. Chvátal. *Linear Programming*. W. H. Freeman and Company, New York, USA, 1983.

[52] L. Collatz und W. Wetterling. *Optimierungsaufgaben*. Springer, Berlin, Germany, 2nd edition, 1971.

[53] H. E. Crowder, E. L. Johnson, und M. W. Padberg. Solving Large Scale 0-1 Linear Programming Problems. *Operations Research*, 31:803–834, 1983.

[54] K. Cunningham und L. Schrage. The LINGO Algebraic Modeling Language. In J. Kallrath, editor, *Modeling Languages in Mathematical Optimization*, Seite 159–171. Kluwer Academic Publishers, Norwell, MA, USA, 2004.

[55] A. R. Curtis und J. K. Reid. On the Automatic Scaling of Matrices for Gaussian Elimination. *Journal of the Institute of Mathematics and its Applications*, 10:118–124, 1972.

[56] R. J. Dakin. A Tree Search Algorithm for Mixed Integer Programming Problems. *Computer Journal*, 8:250–255, 1965.

[57] S. Dallwig, A. Neumaier, und H. Schichl. GLOPT - A Program for Constrained Global Optimization. In I. M. Bomze, T. Csendes, R. Horst, und P. Pardalos, editors, *Developments in Global Optimization*, Seite 19–36. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1997.

[58] E. Danna, E. Rothberg, und C. Le Pape. Exploring Relation Induced Neighborhoods to improve MIP Solutions. *Mathematical Programming*, 102:71–90, 2005.

[59] B. Dantzig und P. Wolfe. Decomposition Principle for Linear Programs. *Operations Research*, 8:101–111, 1960.

[60] B. Dantzig und P. Wolfe. The decomposition algorithm for linear programming. *Operations Research*, 8:101–111, 1960.

[61] G. B. Dantzig. Application of the Simplex Method to a Transportation Problem. In T. C. Koopmans, editor, *Activity Analysis of Production and Allocation*, Seite 359–373. Wiley, New York, 1951.

[62] G. B. Dantzig. Discrete Variable Extremum Problems. *Operations Research*, 5:266–277, 1957.

[63] G. B. Dantzig. *Linear Programming and Extensions*. Princeton University Press, Princeton, New Jersey, 1963.

[64] G. B. Dantzig. Linear Programming. *Operations Research*, 50:42–47, 2002.

[65] M. Davis und H. Putnam. A Computing Procedure for Quantification Theory. *J. ACM*, 7(3):201–215, July 1960.

[66] G. M. De Beuckelaer. *It's Broken, Let's Fix It - The Zeitgeist and Modern Enterprise*. Springer, Berlin, Deutschland, 2001.

[67] I. R. de Farias Jr., E. L. Johnson, und G. L. Nemhauser. A Generalized Assignment Problem with Special Ordered Sets: a Polyhedral Approach. *Mathematical Programming Ser. A*, 89:187–203, 2000.

[68] I. R. de Farias Jr., M. Zhao, und H. Zhao. A Special Ordered Set Approach for Optimizing a Discontinuous Separable Piecewise Linear Function. *Operations Research Letters*, 36:234–238, 2008.

[69] G. Desaulniers, J. Desrosiers, A. Erdmann, M. M. Solomon, und F. Soumis. VRP with Pickup and Delivery. In P. Toth und D. Vigo, editors, *The Vehicle Routing Problem*, Seite 225–242. Society for Industrial and Applied Mathematics, Philadelphia, 2001.

[70] M. Desrochers, J. Desrosiers, und M. M. Solomon. A New Optimization Algorithm for the Vehicle Routing Problem with time Windows. *Operations Research*, 40(2):342–354, 1992, March-April.

[71] J. Desrosiers, Y. Dumas, M. M. Solomon, und F. Soumis. Time Constrained Routing and Scheduling. In M. E. Ball, T. L. Magnanti, C. Monma, und G. L. Nemhauser, editors, *Handbook in Operations Research and Management Science*, Seite 35–140. Society for Industrial and Applied Mathematics, Philadelphia, 1995.

[72] J. V. Dinter, S. Rebennack, J. Kallrath, P. Denholm, und A. Newman. The Unit Commitment Model: A Tight Formulation for Benders' Decomposition with a Case Study. *Annals of Operations Research*, submitted, 2011.

[73] A. S. Drud. CONOPT - A Large-Scale GRG Code. *ORSA Journal of Computing*, 6(2):207–218, 1994.

[74] U. Dudley. *Mathematical Cranks*. MAA, San Francisco, CA, 1992.

[75] M. A. Duran und I. E. Grossmann. An Outer-Approximation Algorithm for a Class of Mixed-Integer NonLinear Programms. *Mathematical Programming*, 36:307–339, 1986.

[76] R. G. Dyson und A. S. Gregory. The Cutting Stock Problem in the Glass Industry. *Operational Research Quarterly*, 25:41–54, 1974.

[77] R. W. Eglese. Simulated Annealing: A Tool for Operational Research. *European Journal of Operational Research*, 46:271–281, 1990.

[78] T. Eichner, A. Pfingsten, und A. Wagener. Strategisches Abstimmungsverhalten bei Verwendung der Hare-Regel. *zfbv*, 48:466–473, 1996.

[79] A. Eisenblätter. *Frequency Assignment in GSM Networks - Models, Heuristics, and Lower Bounds*. Cuvillier, Göttingen, 2001.

[80] S. Engell, A. Märkert, G. Sand, R. Schultz, und C. Schulz. Online Scheduling of Multiproduct Batch Plants under Uncertainty. In *Online Optimization of Large Scale Systems*, Seite 649–676. Springer, Berlin, Germany, 2001.

[81] A. A. Farley. Planning the Cutting of Photographic Color Paper Rolls for Kodak (Australasis) Pty. Ltd. *Interfaces*, 21(1):96–106, 1991.

[82] A. V. Fiacco und G. P. McCormick. *NonLinear Programming. Sequential Unconstrained Minimization Techniques*. John Wiley and Sons, New York, 1968.

[83] M. Fieldhouse. The Pooling Problem. In T. Ciriani und R. C. Leachman, editors, *Optimization in Industry: Mathematical Programming and Modeling Techniques in Practice*, Seite 223–230. John Wiley and Sons, Chichester, 1993.

[84] M. Fischetti und F. Glover. The Feasibility Pump. *Mathematical Programming*, 104:91–104, 2005.

[85] M. Fischetti und A. Lodi. Local Branching. *Mathematical Programming*, 98:23–47, 2003.

[86] M. L. Fisher. An Applications Oriented Guide to Lagrangian Relaxation. *Interfaces*, 15:10–21, 1985.

[87] R. Fletcher. *Practical Methods of Optimization*. Wiley, Chichester, UK, 2nd edition, 1987.

[88] C. A. Floudas. *Nonlinear and Mixed-Integer Optimization : Fundamentals and Applications*. Oxford University Press, Oxford, England, 1995.

[89] C. A. Floudas. *Deterministic Global Optimization: Theory, Methods and Applications*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 2000.

[90] C. A. Floudas und C. E. Gounaris. A review of recent advances in global optimization. *Journal of Global Optimization*, 45:3–38, 2009.

[91] C. A. Floudas und P. M. Pardalos, editors. *Frontiers in Global Optimization*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 2004.

[92] C. A. Floudas, P. M. Pardalos, C. S. Adjiman, W. R. Esposito, Z. H. Gümüs, S. T. Harding, J. L. Klepeis, C. A. Meyer, und C. A. Schweiger. *Handbook of Test Problems of Local and Global Optimization*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 2000.

[93] J. J. H. Forrest, J. P. H. Hirst, und J. A. Tomlin. Practical Solution of Large Mixed Integer Programming Problems with UMPIRE. *Management Science*, 20:736–773, 1974.

[94] R. Fourer, D. M. Gay, und B. W. Kernighan. *AMPL: A Modeling Language for Mathematical Programming*. Duxbury Press, Brooks/Cole Publishing Company, Monterey, CA, 1993.

[95] R. M. Freund und S. Mizuno. Interior Point Methods: Current Status and Future Directions. *Optima (Mathematical Programming Society Newsletter)*, 51:1–9, 1996.

[96] K. R. Frisch. The logarithmic potential method for convex Programming. Technical report, University Institute of Economics, Oslo, Norway, 1955.

[97] D. Fulkerson und P. Wolfe. An Algorithm for Scaling Matrices. *SIAM Review*, 4:142–147, 1962.

[98] M. R. Garey und D. S. Johnson. *Computers and Intractability - A Guide to the Theory of NP Completeness*. W. H. Freeman and Company, New York, USA, 22nd edition, 2000.

[99] A. M. Geoffrion. Generalized Benders Decomposition. *Journal of Optimization Theory and Applications*, 10:237–260, 1972.

[100] A. M. Geoffrion. Lagrangian Relaxation and its Uses in Integer Programming. *Mathematical Programming Study*, 2:82–114, 1974.

[101] V. Ghildyal und N. V. Sahinidis. Solving Global Optimization Problems with BARON. In A. Migdalas, P. Pardalos, und P. Värbrand, editors, *From Local to Global Optimization*, Seite 205–230. Kapitel 10, Kluwer Academic Publishers, Dordrecht, The Netherlands, 2001.

[102] P. E. Gill, W. Murray, und M. A. Saunders. SNOPT: An SQP Algorithm for Large-scale Constrained Optimization. Numerical analysis report 97-2, Department of Mathematics, University of California, San Diego, San Diego, La Jolla, CA, 1997.

[103] P. E. Gill, W. Murray, M. A. Saunders, J. A. Tomlin, und M. H. Wright. On the Projected Newton Barrier Methods for Linear Programming and an Equivalence to Karmarkar's Projective Method. *Math. Programming*, 36:183–209, 1986.

[104] P. E. Gill, W. Murray, und M. H. Wright. *Practical Optimization*. Academic Press, London, 1981.

[105] P. C. Gilmore und R. E. Gomory. A Linear Programming Approach to the Cutting Stock Problem. *Operations Research*, 9:849–859, 1961.

[106] P. C. Gilmore und R. E. Gomory. A Linear Programming Approach to the Cutting Stock Problem. *Operations Research*, 9:849–859, 1961.

[107] P. C. Gilmore und R. E. Gomory. A Linear Programming Approach to the Cutting Stock Problem, Part II. *Operations Research*, 11:863–888, 1963.

[108] F. Glover. A New Foundation for a Simplified Primal Integer Programming Algorithm. *Operations Research*, 16:727–740, 1968.

[109] F. Glover und M. Laguna. *Tabu Search*. Kluwer Academic Publisher, Dordrecht, The Netherlands, 1997.

[110] R. Gollmer, M. P. Nowak, W. Römisch, und R. Schultz. Unit Commitment in Power Generation - A Basic Model and Some Extensions. *Annals of Operations Research*, 96:167–189, 2000.

[111] R. E. Gomory. Outline of an Algorithm for Integer Solutions to Linear Programming. *Bulletin of the American Mathematical Society*, 64:275–278, 1958.

[112] R. E. Gomory. Outline of an Algorithm for Integer Solutions to Linear Programs. *Bull. Amer. Math. Soc.*, 64:275–278, 1958.

[113] R. E. Gomory. Early Integer Programming. *Operations Research*, 50:78–81, 2002.

[114] C. L. Gonzaga. Path-Following Methods for Linear Programming. *SIAM Review*, 34:167–224, 1992.

[115] N. I. M. Gould und J. K. Reid. New Crash Procedures for Large Systems of Linear Constraints. *Mathematical Programming*, 45:475–503, 1989.

[116] F. Granot und P. L. Hammer. On the Use of Boolean Functions in 0-1 Programming. *Methods of Operations Research*, 12:154–184, 1972.

[117] H. J. Greenberg. How to Analyse the Results of Linear Programs - Part 1: Preliminaries. *Interfaces*, 23(4):56–67, 1993.

[118] H. J. Greenberg. How to Analyse the Results of Linear Programs - Part 2: Price Interpretation. *Interfaces*, 23(5):97–114, 1993.

[119] H. J. Greenberg. How to Analyse the Results of Linear Programs - Part 3: Infeasibility. *Interfaces*, 23(6):120–139, 1993.

[120] H. J. Greenberg. How to Analyse the Results of Linear Programs - Part 4: Forcing Substructures. *Interfaces*, 24(1):121–130, 1994.

[121] M. Grötschel und L. Lovasz. Combinatorial Optimization. In R. L. Graham, editor, *Handbook on Combinatorics*, Seite 1541–1597. North-Holland, Amsterdam, 1982.

[122] C. Guéret, S. Heipcke, C. Prins, und M. Sevaux. *Applications of Optimization with Xpress-MP*. Dash Optimization, Blisworth, UK, 2002.

[123] O. Günlük und Y. Pochet. Mixing Mixed-Integer Inequalities. *Math. Program. Series A*, 90:429–457, 2001.

[124] O. K. Gupta und V. Ravindran. Branch and Bound Experiments in Convex Non-Linear Integer Programming. *Management Science*, 31:1533–1546, 1985.

[125] M. T. Hajian, R. Rodošek, und E. B. Richards. Introduction of a New Class of Variables to Discrete and Integer Programming Problems. *Annals of Operations Research*, 86:39–51, 1999.

[126] H. W. Hamacher. *Mathematische Lösungsverfahren für planare Standortprobleme*. Vieweg, Braunschweig/Wiesbaden, 1995.

[127] S. T. Harding und C. A. Floudas. Locating Heterogeneous and Reactive Azeotropes. *Industrial and Engineering Chemistry Research*, 39:1576–1595, 2000.

[128] I. Harjunkoski. *Application of MINLP Methods on a Scheduling Problem in the Paper Converting Industry*. PhD Dissertation, Abo Akademi University, Abo, Finland, 1997.

[129] I. Harjunkoski, V. Jain, und I. E. Grossmann. Hybrid Mixed-integer/Constrained Logic Programming Strategies for Solving Scheduling and Combinatorial Optimization Problems. *Computers and Chemical Engineering*, 24:337–343, 2000.

[130] P. M. J. Harris. Pivot Selection Methods of the Devex LP Code. *Mathematical Programming*, 5:1–28, 1973.

[131] S. Heipcke. *Mosel: Modeling and Optimization*. Dash Optimization, Blisworth, England, 2002.

[132] M. H. Held, P.Wolfe, und H. D. Crowder. Validation of Subgradient Optimization. *Mathematical Programming*, 6:62–88, 1974.

[133] D. Hertz. The Extreme Eigenvalues and Stability of Real Symmetric Interval Matrices. *IEEE Transactions on Automatic Control*, 37:532–535, 1992.

[134] S. Hölldobler, V. H. Nguyen, J. Stocklina, und P. Steinke. A short overview on modern parallel sat-solvers. In *Proceedings of the International Conference on Advanced Computer Science and Information Systems*, Seite 201–206, 2011.

[135] J. Hooker. *Logic-Based Methods for Optimization: Combining Optimization and Constraint Satisfaction*. Wiley, Chichester, UK, 2000.

[136] R. Horst und P. M. Pardalos, editors. *Handbook of Global Optimization*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1995.

[137] R. Horst, P. M. Pardalos, und N. V. Thoai. *Introduction to Global Optimization*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1996.

[138] R. Horst, P. M. Pardalos, und N. V. Thoai. *Introduction to Global Optimization*. Kluwer Academic Publishers, 2nd edition, 2000.

[139] R. Horst und H. Tuy. *Global Optimization: Deterministic Approaches*. Springer, New York, 3rd edition, 1996.

[140] W. H. Hummeltenberg. Implementations of Special Ordered Sets in MP Software. *European Journal of Operational Research*, 17:1–15, 1984.

[141] T. Hürlimann. The LPL Modeling Language. In J. Kallrath, editor, *Modeling Languages in Mathematical Optimization*, Seite 173–183. Kluwer Academic Publishers, Norwell, MA, USA, 2004.

[142] M. G. Ierapetriou und C. A. Floudas. Effective Continuous-Time Formulation for Short-Term Scheduling. 1. Multipurpose Batch Processes. *Industrial and Engineering Chemistry Reserach*, 37:4341–4359, 1998.

[143] M. G. Ierapetriou und C. A. Floudas. Effective Continuous-Time Formulation for Short-Term Scheduling. 2. Continuous and Semicontinuous Processes. *Industrial and Engineering Chemistry Reserach*, 37:4360–4374, 1998.

[144] M. G. Ierapetriou, T. S. Hene, und C. A. Floudas. Continuous Time Formulation for Short-Term Scheduling with Multiple Intermediate Due Dates. *Industrial and Engineering Chemistry Research*, 38:3446–3461, 1999.

[145] J. P. Ignizio. *Goal Programming and Extensions*. Heath, Lexington, Massachusetts, USA, 1976.

[146] V. Jain und I. E. Grossmann. Algorithms for Hybrid MILP/CP Models for a Class of Optimization Problems. *IFORMS Journal on Computing*, 13:258–276, 2001.

[147] L. Jaulin, M. Kieffer, O. Didrit, und E. Walter. *Applied Interval Analysis*. Springer, London, England, 2001.

[148] R. G. Jeroslow und J. K. Lowe. Modelling with Integer Variables. *Mathematical Programming Study*, 22:167–184, 1984.

[149] R. G. Jeroslow und J. K. Lowe. Experimental Results with the New Techniques for Integer Programming Formulations. *Journal of the Operational Research Society*, 36:393–403, 1985.

[150] S. Jetzke und J. Kallrath. Optimaltät ist nicht genug für die Logistik. *OR News*, 44:6–10, 2012.

[151] E. L. Johnson, M. M. Kostreva, und U. H. Suhl. Solving 0-1 integer Programming Problems Arising from Large Scale Planning Models. *Operations Research*, 33:803–819, 1985.

[152] R. E. Johnston und E. Sadinlija. A New Model for Complete Solutions to One-Dimensional Cutting Stock Problems. *European Journal of Operational Research*, 153:176–183, 2004.

[153] L. R. F. (Jr) und D. R. Fulkerson. Maximum Flow Through a Network. *Canadian Journal of Mathematics*, 8:399–404, 1956.

[154] J. Kallrath. Mixed-Integer Nonlinear Programming Applications. In T. A. Ciriani, S. Gliozzi, E. L. Johnson, und R. Tadei, editors, *Operational Research in Industry*, Seite 42–76. Macmillan, Houndmills, Basingstoke, UK, 1999.

[155] J. Kallrath. Combined Strategic, Design and Operative Planning - Two Success Stories in MILP and MINLP. In V. Bulatov und V. Baturin, editors, *Proceedings of 12th Baikal International Conference: Optimization Methods and Their Applications*, Seite 123–128, Irkutsk, Russia, 2001. Institute of System Dynamics and Control Theory.

[156] J. Kallrath. Combined Strategic and Operational Planning - An MILP Success Story in Chemical Industry. *OR Spectrum*, 24(3):315–341, 2002.

[157] J. Kallrath. Planning and Scheduling in the Process Industry. *OR Spectrum*, 24(3):219–250, 2002.

[158] J. Kallrath. Exact Computation of Global Minima of a Nonconvex Portfolio Optimization Problem. In C. A. Floudas und P. M. Pardalos, editors, *Frontiers in Global Optimization*, Seite 237–254. Kluwer Academic Publishers, 2004.

[159] J. Kallrath. Modeling Difficult Optimization Problems. In C. A. Floudas und P. M. Pardalos, editors, *Encyclopedia of Optimization*, Seite 2284–2297. Springer Verlag, New York, 2008.

[160] J. Kallrath. Combined Strategic Design *and* Operative Planning in the Process Industry. *Computers and Chemical Engineering*, 33:1983–1993, 2009.

[161] J. Kallrath. Cutting Circles and Polygons from Area-Minimizing Rectangles. *Journal of Global Optimization*, 43:299–328, 2009.

[162] J. Kallrath. Polylithic Modeling and Solution Approaches Using Algebraic Modeling Systems. *Optimization Letters*, 5:453–466, 2011. 10.1007/s11590-011-0320-4.

[163] J. Kallrath und T. I. Maindl. *Real Optimization with SAP-APO*. Springer, Heidelberg, Germany, 2006.

[164] J. Kallrath und S. Rebennack. Cutting Ellipses from Area-Minimizing Rectangles. *Journal of Global Optimization*, in preparation, 2013.

[165] J. Kallrath, S. Rebennack, J. Kallrath, und R. Kusche. Non-Standard Cutting Stock-Problems in the Paper Industry. *Management Science*, submitted, 2013.

[166] J. Kallrath und A. Schreieck. Discrete Optimization and Real World Problems. In B. Hertzberger und G. Serazzi, editors, *High-Performance Computing and Networking*, number 919 in Lecture Notes in Computer Science, Seite 351–359, Berlin-Heidelberg-New York, 1995. Springer.

[167] J. Kallrath und J. M. Wilson. *Business Optimisation Using Mathematical Programming*. Macmillan, Houndmills, Basingstoke, UK, 1997.

[168] E. Kalvelagen. Branch-and-Bound Methods for an MINLP Model with Semi-Continuous Variables, 2003, discontinued on http://www.gams.com.

[169] L. V. Kantorovich. O peremeschenii mass. *Doklady Akademii Nauk*, 37:227, 1942.

[170] L. V. Kantorovich und M. K. Gavurin. Primenenie matematicheskih metodov v voprosah analiza gruzopotokov. Problemy povysheniya efektivnosti transporta. *Izdatelstvo AN SSSR*, 1:110–138, 1949.

[171] N. Karmarkar. A New Polynomial Time Algorithm for Linear Programming. *Combinatorica*, 4:375–395, 1984.

[172] R. M. Karp. Reducibility among Combinatorial Problems. In R. E. Miller und J. W. Thatcher, editors, *Complexity of Computer Computations*, Seite 85–103. Plenum Press, New York, 1972.

[173] R. Karuppiah und I. E. Grossmann. A Lagrangean based branch-and-cut algorithm for global optimization of nonconvex mixed-integer nonlinear programs with decomposable structures. *Journal of Global Optimization*, 41:163–186, DOI: 10.1007/s10898–007–9203–8, 2008.

[174] W. Karush. *Minima of Functions of Several Variables with Inequalities as Side Constraints*. Master thesis, Department of Mathematics, University of Chicago, Chicago, 1939.

[175] R. B. Kearfott. *Rigorous Global Search: Continuous Problems*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1996.

[176] L. G. Khachian. A Polynomial Algorithm in Linear Programming. *Soviet Mathematics Doklady*, 20:191–194, 1979.

[177] G. W. Klau. *A Combinatorial Approach to Orthogonal Placement Problems*. Shaker, Aachen, 2001.

[178] V. Klee und G. J. Minty. How Good is the Simplex algorithm? In O. Shisha, editor, *Inequalities III*, Seite 159–175. Academic Press, New York, 1972.

[179] N. Kliewer, B. Amberg, und B. Amberg. Optimierungssysteme im ÖPNV: Mehrdepot-Umlauf- und Dienstplanung mit Zeitfenstern für geplante Fahrten. In H. R. Hansen, D. Karagiannis, und H.-G. Fill, editors, *Tagungsband 9. Internationale Tagung Wirtschaftsinformatik*, volume 247 of *books@ocg.at*, Seite 55–66. Österreichische Computer Gesellschaft, 2009.

[180] N. Kliewer, B. Amberg, und B. Amberg. Multiple Depot Vehicle and Crew Scheduling with Time Windows for Scheduled Trips. *Public Transport*, 3:213–244, 2012.

[181] A. Klose. *Standortplanung in distributiven Systemen. Modelle, Methoden, Anwendungen (Betriebswirtschaftliche Studien)*. Physica, Heidelberg, Deutschland, 2001.

[182] G. G. Kocis und I. E. Grossmann. Global Optimization of Nonconvex MINLP Problems in Process Synthesis. *Industrial and Engineering Chemistry Reserach*, 27:1407–1421, 1988.

[183] A. A. Kolokolov. Regular Partitions and Cuts in Integer Programming. In A. D. Koshunov, editor, *Discrete Analysis and Operations Research*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1996.

[184] A. A. Korbut und J. J. Finkelstein. *Diskrete Optimierung*. Akademie-Verlag, Berlin, Deutschland, Deutsche Übersetzung von H. Hollatz, 1971.

[185] B. Kristjansson und D. Lee. The MPL Modeling System. In J. Kallrath, editor, *Modeling Languages in Mathematical Optimization*, Seite 239–266. Kluwer Academic Publishers, Norwell, MA, USA, 2004.

[186] W. Küchlin und C. Sinz. Proving Consistency Assertions for Automotive Product Data Management. *Journal of Automated Reasoning*, 24:145–163, 2000.

[187] H. Kuhn. Nonlinear Programming: A Historical View. In R. Cottle und C. Lemke, editors, *Nonlinear Programming*, volume 9 of *SIAM-AMS Proceedings*, Seite 1–26, Providence, RI, 1976. American Mathematical Society.

[188] H. W. Kuhn und A. W. Tucker. Nonlinear Programming. In J. Neumann, editor, *Proceedings Second Berkeley Symposium on Mathematical Statistics and Probability*, Seite 481–492, Berkeley, CA, 1951. University of California.

[189] Y. C. L. und Y. H.-C. To Estimate Vapor Pressure Easily. Antoine Coefficients Relate Vapor Pressure to Temperature for Almost 700 Major Organic Compounds. *Hydrocarbon Processing*, 68:65–68, 1989.

[190] A. H. Land und A. G. Doig. An Automatic Method for Solving Discrete Programming Problems. *Econometrica*, 28:497–520, 1960.

[191] L. S. Lasdon und A. D. Waren. Generalized Reduced Gradient Method for Linearly and Nonlinearly Constrained Programming. In H. J. Greenberg, editor, *Design and Implementation of Optimization Software*, Seite 363–397. Sijthoff and Noordhoff, The Netherlands, 1978.

[192] L. S. Lasdon, A. D. Waren, A. Jain, und M. Ratner. Design and Testing of a Generalized Reduced Gradient Code for Nonlinear Programming. *ACM Trans. Math. Software*, 4:34–50, 1978.

[193] L. Liberti und N. Maculan, editors. *Global Optimization: From Theory to Implementation*, volume 84 of *Nonconvex Optimization and Its Applications*. Springer, 2006. 223–232.

[194] X. Lin und C. A. Floudas. Design, Synthesis and Scheduling of Multipurpose Batch Plants via an Effective Continuous-Time Formulation. *Computers and Chemical Engineering*, 25:665–674, 2001.

[195] X. Lin, C. A. Floudas, und J. Kallrath. Global Solution Approaches for Nonconvex MINLP Problems in Product Portfolio Optimization. *Journal of Global Optimization*, 32:417–431, 2005.

[196] X. Lin, C. A. Floudas, S. Modi, und N. M. Juhasz. Continuous-Time Production Scheduling of a Multiproduct Batch Plant. *Industrial and Engineering Chemistry Research*, im Druck, 2002.

[197] M. E. Lübbecke und J. Desrosiers. Selected topics in column generation. *Oper. Res.*, 53(6):1007–1023, 2005.

[198] I. J. Lustig, R. E. Marsten, und D. F. Shanno. Computational experience with a primal-dual interior point method for Linear Programming. *Linear Algebra Applications*, 152:191–222, 1991.

[199] I. J. Lustig, R. E. Marsten, und D. F. Shanno. On Implementing Mehrotra's Predictor-Corrector Interior-Point Method for Linear Programming. *SIAM Journal of Optimisation*, 2:435–449, 1992.

[200] I. Lynce und J. ao Marques-Silva. SAT 2006. In A. Biere und C. P. Gomes, editors, *SAT in Bioinformatics: Making the Case with Haplotype Inference*, LNCS 4121, Seite 136–141. Springer, Heidelberg, Germany, 2006.

[201] R. A. Main. Large Recursion Models: Practical Aspects of Recursion Techniques. In T. Ciriani und R. C. Leachman, editors, *Optimization in Industry: Mathematical Modeling Techniques in Practice*, Seite 241–249. John Wiley and Sons, Chichester, 1993.

[202] I. Maros und G. Mitra. Finding Better Starting Bases for the Simplex Method. In P. Kleinschmidt, editor, *Operations Research Proceedings 1995*, Berlin, 1996. Springer Verlag.

[203] S. Martello und P. Toth. *Knapsack Problems: Algorithms and Computer Implementations*. John Wiley and Sons, Chichester, 1990.

[204] R. K. Martin. *Large Scale Linear and Integer Optimization – A Unified Approach*. Kluwer, Dortrecht, The Netherlands, 1999.

[205] G. Mayer. *Strategische Logistikplanung von Hub&Spoke-Systemen*. Gabler, Wiesbaden, 2001.

[206] G. P. McCormick. Computation of Global Solutions to Factorable Nonconvex Programs: Part I - Convex Underestimations Problems. *Mathematical Programming*, 10:147–175, 1976.

[207] K. I. M. McKinnon und H. P. Williams. Constructing integer Programming models by the predicate calculus. *Annals of Operations Research*, 21:227–246, 1989.

[208] P. McMullen. The Maximum Number of Faces of Convex Polytopes. *Mathematika*, 17:179–184, 1970.

[209] S. Mehrotra. On the Implementation of a Primal-Dual Interior Point Method. *SIAM Journal on Optimization*, 2(4):575–601, 1992.

[210] H. Meyr, J. Rohde, H. Stadtler, und C. Sürie. Architecture of Selected APS. In H. Stadtler und C. Kilger, editors, *Supply Chain Analysis*, Seite 241–249. Springer, Berlin, Deutschland, 2000.

[211] Z. Michaelewicz und D. B. Fogel. *How to Solve It: Modern Heuristics*. Springer, Berlin, 2000.

[212] C. E. Miller. The Simplex Method for Local Separable Programming. In R. L. Graves und P. L. Wolfe, editors, *Recent Advances in Mathematical Programming*, Seite 311–317. McGraw-Hill, London, 1963.

[213] R. Misener und C. Floudas. GloMIQO: Global Mixed-integer Quadratic Optimizer. *Journal of Global Optimization*, Seite 1–48, 2012. 10.1007/s10898-012-9874-7.

[214] R. Misener und C. A. Floudas. Piecewise-Linear Approximations of Multidimensional Functions. *Journal of Optimization Theory and Applications*, 145:120–147, 2010.

[215] G. Mitchell. *The Practice of Operational Research*. John Wiley and Sons, Chichester, 1993.

[216] G. Mitra, K. Darby-Dowman, C. Lucas, und J. W. Smith. Maritime Scheduling Using Discrete Optimization and Artificial Intelligence Techniques. In A. Sciomachen, editor, *Optimization in Industry 3: Mathematical Programming Techniques in Practice*, Seite 1–17. John Wiley and Sons, Chichester, 1995.

[217] J. Muñoz, G. Gutierrez, und A. Sanchis. Evolutionary techniques in a constraint satisfaction problem: Puzzle Eternity II. In *Proceedings 2009 IEEE Congress on Evolutionary Computation*, Seite 2985–2991, May 2009.

[218] B. A. Murtagh und M. A. Saunders. Large-scale Linearly Constrained Optimization. *Mathematical Programming*, 14:41–72, 1978.

[219] B. A. Murtagh und M. A. Saunders. A Projected Lagrangian Algorithm and its Implementation for Sparse Nonlinear Constraints. *Mathematical Programming Study (Algorithm for Constrained Minimization of Smooth Nonlinear Function)*, 16:84–117, 1982.

[220] J. A. Nelder und R. Mead. A Simplex Method for Function Minimization. *Comp. J.*, 7:308–313, 1965.

[221] G. L. Nemhauser. The Age of Optimization: Solving Large-Scale Real World-Problems. *Operations Research*, 42:5–13, 1994.

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

[223] A. Neumaier. NOP - A Compact Input Format for Nonlinear Optimization Problems. In I. M. Bomze, T. Csendes, R. Horst, und P. Pardalos, editors, *Developments in Global Optimization*, Seite 1–18. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1997.

[224] K. Neumann und M. Morlock. *Operations Research*. Carl Hanser, München, Wien, 1993.

[225] K. Neumann, C. Schwindt, und J. Zimmermann. *Project Scheduling with Time Windows and Scarce Resources*. Springer, Berlin, Deutschland, 2002.

[226] S. Nickel. *Discretization of Planar Location Problems*. Shaker, Aachen, Deutschland, 1995.

[227] R. K. Oliver und M. D. Webber. Supply-chain Management: Logistics Catches up with Strategy. In M. Christopher, editor, *Logistics – The Strategic Issues*, Seite 63–75. Springer, Berlin, Deutschland (reprint of OUTLOOK 1982), 1992.

[228] M. Padberg. *Linear Optimization and Extensions*. Springer, Berlin - Heidelberg, 1996.

[229] M. W. Padberg und G. Rinaldi. Optimization of a 532-city Traveling Salesman Problem by Branch and Cut. *Operations Research Letters*, 6:1–6, 1987.

[230] C. H. Papadimitriou und K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*. Prentice Hall, Englewood Cliffs, NJ, 1982.

[231] G. Parija, R. Gadidov, und W. Wilhelm. A Facet Generation Procedure for Solving 0/1 Integer Programs. *Operations Research*, 5:789–791, 1999.

[232] G. D. Pillo, S. Lucidi, und F. Rinaldi. An approach to constrained Global Optimization based on Exact Penalty Functions. *Journal of Global Optimization*, Seite to appear, DOI 10.1007/s10898–010–9582–0, 2010.

[233] Y. Pochet und L. A. Wolsey. *Production Planning by Mixed Integer Programming*. Springer, New York, 2006.

[234] B. T. Poljak. A General Method of Solving Extremum Problems. *Soviet Mathematics Doklady*, 8:593–597, Translation of Doklady Akademii Nauk SSSR 174, 1967, 1967.

[235] G. Polya. *Vom Lernen und Lösen mathematischer Aufgaben. Einsicht und Entdeckung. Lernen und Lehren*. Birkhäuser Verlag, Basel, 1979.

[236] K. R. Popper. *The Logic of Scientific Discovery*. Hutchinson, London, 10th edition, 1980.

[237] W. H. Press, B. P. Flannery, S. A. Teukolsky, und W. T. Vetterling. *Numerical Recipes - The Art of Scientific Computing*. Cambridge University Press, Cambridge, UK, 2nd edition, 1992.

[238] H. Ratschek und J. Rokne. Experiments using Interval Analysis for Problem Solving a Circuit Design Problem. *Journal of Global Optimization*, 3:501–518, 1993.

[239] H. Ratschek und J. Rokne. Interval Methods. In R. Horst und P. M. Pardalos, editors, *Handbook of Global Optimization*, Seite 751–828. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1995.

[240] A. Ravindran, D. T. Phillips, und J. J. Solberg. *Operations Research. Principles and Practice*. John Wiley & Sons, New York, 1987.

[241] S. Rebennack und J. Kallrath. Continuous Piecewise Linear δ-Approximations for MINLP Problems. I. Minimal Breakpoint Systems for Univariate Functions. *Mathematical Programming*, submitted, 2013.

[242] S. Rebennack und J. Kallrath. Continuous Piecewise Linear δ-Approximations for MINLP Problems. II. Bivariate und Multivariate Functions. *Mathematical Programming*, submitted, 2013.

[243] S. Rebennack, J. Kallrath, und P. M. Pardalos. Column Enumeration based Decomposition Techniques for a Class of Non-convex MINLP Problems. *Journal of Global Optimization*, 43:277–297, 2009.

[244] S. Rebennack, J. Kallrath, und P. M. Pardalos. Column Enumeration based Decomposition Techniques for a Class of Non-Convex MINLP Problems. *Journal of Global Optimization*, in press, 2009.

[245] S. Rebennack, A. Nahapetyan, und P. M. Pardalos. Bilinear Modeling Solution Approach for Fixed Charged Network Flow Problems. *Optimization Letters*, 3:347–355, 2009.

[246] S. Rebennack, M. Oswald, D. O. Theis, H. Seitz, G. Reinelt, und P. M. Pardalos. A Branch and Cut Solver for the Maximum Stable Set Problem. *Journal of Combinatorial Optimization*, Seite to appear, DOI:10.1007/s10878–008–9175–8, 2011.

[247] S. Rebennack, G. Reinelt, und P. M. Pardalos. A Tutorial on Branch&Cut Algorithms for the Maximum Stable Set Problem. *International Transactions in Operational Research*, Seite to appear, DOI:10.1111/j.1475–3995.2011.00805.x, 2011.

[248] S. M. Robinson. A Quadratically Convergent Algorithm for General Nonlinear for Programming Problems. *Mathematical Programming*, 3:145–156, 1972.

[249] J. Rohde, H. Meyr, und M. Wagner. Architecture of Selected APS. In H. Stadtler und C. Kilger, editors, *Supply Chain Management and Advanced Planning*, Seite 29–56. Springer, Berlin, Deutschland, 2000.

[250] J. Rohde, H. Meyr, und M. Wagner. Die Supply Chain Planning Matrix. *PPS-Management*, 5(1):10–15, 2000.

[251] C. Romero. *Handbook of Critical Issues in Goal Programming.* Pergamon Press, Oxford, 1991.

[252] H. M. Salkin und C. A. D. Kluyver. The Knapsack Problem: a Survey. *Naval Research Logistics Quarterly*, 24:127–144, 1975.

[253] G. Sand, S. Engell, A. Märkert, R. Schultz, und C. Schulz. Approximation of an Ideal Online Scheduler for a Multiproduct Batch Plant. *Computers and Chemical Engineering*, 24:361–367, 2000.

[254] M. W. P. Savelsbergh. Solving the Asymmetric Travelling Salesman Problem with Time Windows by Branch-and-Cut. *Math. Program. Series A*, 90:475–506, 1985.

[255] M. W. P. Savelsbergh. Preprocessing and Probing Techniques for Mixed Integer Programming Problems. *ORSA Journal on Computing*, 6:445–454, 1994.

[256] M. W. P. Savelsbergh. A Branch-and-Price Algorithm for the Generalized Assignment Problem. *Operations Research*, 6:831–841, 1997.

[257] M. W. P. Savelsbergh. Branch-and-Price: Integer Programming with Column Generation. In C. A. Floudas und P. Pardalos, editors, *Encyclopedia of Optimization*, Seite 218–221. Kluwer Academic Publishers, Dordrecht, Holland, 2001.

[258] M. W. P. Savelsbergh, G. C. Sismondi, und G. L. Nemhauser. Functional Description of MINTO and a Mixed INTeger Optimizer. *Operations Research Letters*, 8:119–124, 1994.

[259] G. Scheithauer und J. Terno. The Modified Integer Round-up Property of the one-dimensional Cutting Stock Problem. *European Journal of Operational Research*, 84:562–571, 1995.

[260] L. Schrage. *Optimization Modeling with LINDO*. Duxbury Press, Brooks/Cole Publishing Company, Monterey, CA, 1997.

[261] L. Schrage. LindoSystems: LindoAPI, 2004.

[262] L. Schrage. *Optimization Modeling with LINGO*. LINDO Systems, Inc., Chicago, IL, 2006.

[263] R. Schultz. On Structure and Stability in Stochastic Programs with Random Technology Matrix and Complete Integer Recourse. *Mathematical Programming*, 70:73–89, 1995.

[264] C. A. Schweiger und C. A. Floudas. Optimization Framework for the Synthesis of Chemical Reactor Networks. *Industrial and Engineering Chemistry Research*, 38:744–766, 1999.

[265] P. Spelluci. *Numerische Verfahren der nichtlinearen Optimierung*. Birkhäuser, Basel, 1993.

[266] W. Spendley, G. R. Hext, und F. R. Himsworth. Sequential Application of Simplex Designs in Optimisation and Evolutionary Operation. *Technometrics*, 4:441–461, 1962.

[267] J. Stoer. Foundations of Recursive Quadratic Programming Methods for Solving Nonlinear Programs. In K. Schittkowski, editor, *Computational Mathematical Programming*, number 15 in NATO ASI Series, Heidelberg, Germany, 1985. Springer.

[268] R. Subramanian, R. P. Scheff(Jr.), J. D. Quinlan, D. S. Wiper, und R. E. Marsten. Coldstart: Fleet assignment at Delta Air Lines. *Interfaces*, 24(1):104–120, 1994.

[269] M. Tawarmalani und N. V. Sahinidis. *Convexification and Global Optimization in Continuous and Mixed-Integer Nonlinear Programming: Theory, Algorithms, Software, and Applications*, volume 65 of *Nonconvex Optimization And Its Applications*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 2002.

[270] M. Tawarmalani und N. V. Sahinidis. Global Optimization of Mixed Integer Nonlinear Programs: A Theoretical and Computational Study improve MIP Solutions. *Mathematical Programming*, 99:563–591, 2004.

[271] C. Timpe. Solving Mixed Planning & Scheduling Problems with Mixed Branch & Bound and Constraint Programming. *OR Spectrum*, 24, im Druck, 2002.

[272] A. N. Tolstoi. Metody ustraneniya nerachionalnyh perevozok pri planirovanii. *Sochialisticheskii transport*, 9:28–51, 1939.

[273] J. A. Tomlin. On Scaling Linear Programming Problems. *Mathematical Programming*, 4:146–166, 1975.

[274] J. A. Tomlin und J. S. Welch. A Pathological Case in the Reduction of Linear Programs. *Operations Research Letters*, 2:53–57, 1983.

[275] J. A. Tomlin und J. S. Welch. Formal Optimisation of Some Reduced Linear Programming Problems. *Mathematical Programmng*, 27:232–240, 1983.

[276] C. D. M. und C. A. Floudas. Global Minimum Potential Energy Confirmations of Small Molecules. *Journal of Global Optimization*, 4:135–170, 1994.

[277] S. Vajda. *Mathematical Programming.* Adison-Wesley, Reading, Massachusetts, 1961.

[278] P. Van Hentenryck. *The OPL Optimization Programming Language.* MIT Press, Cambridge, MA, 1998.

[279] P. Van Hentenryck, L. Michel, und Y. Deville. *Numerica - A Modeling Language for Global Optimization.* MIT Press, Cambridge, MA, 1997.

[280] F. Vanderbeck. Exact Algorithm for Minimising the Number of Setups in the One-dimensional Cutting Stock Problem. *Operations Research*, 48(5):915–926, 2000.

[281] F. Vanderbeck und L. A. Wolsey. An Exact Algorithm for IP Column Generation. *Operations Research Letters*, 19:151–160, 1996.

[282] R. J. Vanderbei. *Linear Programming - Foundations and Extensions.* Kluwer, Dordrecht, The Netherlands, 1996.

[283] T. J. VanRoy und L. A. Wolsey. Solving Mixed Integer Programs by Automatic Reformulation. *Operations Research*, 35(1):45–57, 1987.

[284] J. P. Vielma, S. Ahmed, und G. Nemhauser. Mixed-Integer Models for Nonseparable Piecewise-Linear Optimization: Unifying Framework and Extensions. *Operations Research*, 53:303–315, 2009.

[285] J. P. Vielma und G. Nemhauser. Modeling disjunctive constraints with a logarithmic number of binary variables and constraints. *Math. Programming Ser. A*, 128:49–72, 2011.

[286] J. Viswanathan und I. E. Grossmann. A Combined Penalty Function and Outer-Approximation Method for MINLP Optimization. *Comp. Chem. Eng.*, 14(7):769–782, 1990.

[287] S. W. Wallace. Decision Making under Uncertainty: Is Sensitivity Analysis of any Use? *Operations Research*, 48(1):20–25, 2000.

[288] J. Werner. *Numerische Mathematik.* Vieweg, Wiesbaden, Deutschland, 1992.

[289] G. O. Wesolowsky. The Weber Problem: History and Perspectives. *Location Science*, 1:5–23, 1993.

[290] T. Westerlund, F. Petterson, und I. E. Grossmann. Optimization of Pump Configuration Problems as a MINLP problem. *Computers and Chemical Engineering*, 18(9):845–858, 1994.

[291] H. P. Williams. *Model Building in Mathematical Programming.* John Wiley and Sons, Chichester, 3rd edition, 1993.

[292] H. P. Williams. The Dual of a Logical Linear Programme. Research paper, Mathematical Sciences, University of Southampton, Southhampton, 1995.

[293] J. M. Wilson. Generating Cuts in Integer Programming with Families of Special Ordered Sets. *European Journal of Operational Research*, 46:101–108, 1990.

[294] T. Winter und U. T. Zimmermann. Real-Time Dispatch of Trams in Storage Yards. *Annals of Operations Research*, 96:287–315, 2000.

[295] P. Wolfe. The Reduced-Gradient Method. unpublished manuscript, RAND Corporation, 1962.

[296] L. A. Wolsey. Group-Theoretic Results in Mixed Integer Programming. *Operations Research*, 19:1691–1697, 1971.

[297] L. A. Wolsey. *Integer Programming*. Wiley, New York, US, 1998.

[298] S. Wright. *Primal-Dual Interior-Point Methods*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 1996.

[299] R. D. Young. A Simplified Primal (All-integer) Integer Programming Algorithm. *Operations Research*, 16:750–782, 1968.

[300] R. E. Young und W. J. Baumol. Integer Programming and Pricing. *Econometrica*, 28:520–550, 1960.

[301] J. Zhang, N. Kim, und L. Lasdon. An Improved Successive Linear Programming Algorithm. *Management Science*, 31:1312–1331, 1985.

Index

N

T